Lecture Notes in Computer Science 2877

Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Thomas Böhme Gerhard Heyer
Herwig Unger (Eds.)

Innovative Internet Community Systems

Third International Workshop, IICS 2003
Leipzig, Germany, June 19-21, 2003
Revised Papers

Springer

Volume Editors

Thomas Böhme
Technische Universität Ilmenau, Institut für Mathematik
Pf 100565, 98684 Ilmenau, Germany
E-mail: tboehme@mathematik.tu-ilmenau.de

Gerhard Heyer
Universität Leipzig, Institut für Informatik
Augustusplatz 10/11, 04109 Leipzig, Germany
E-mail: heyer@informatik.uni-leipzig.de

Herwig Unger
Universität Rostock, Fachbereich Informatik
Albert-Einstein-Str. 23, 18051 Rostock, Germany
E-mail: hunger@informatik.uni-rostock.de

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): C.2, H.3-5, D.2, I.2.11, K.4.3-4

ISSN 0302-9743
ISBN 3-540-20436-9 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
is a part of Springer Science+Business Media GmbH

http://www.springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 10966181 06/3142 5 4 3 2 1 0

Preface

The papers in this volume were presented at the workshop "Innovative Internet Community Systems 2003" held on June 19–21, 2003 in Leipzig. IICS 2003 was the third workshop in the IICS series. The purpose of these workshops is to bring together researchers in the area of system and information management for the Next Generation Internet (NGI).

Like the preceding two workshops, IICS 2001 and IICS 2002, this year's workshop was organized by the Gesellschaft für Informatik (GI) in Germany to support the exchange of experiences, results and technology in the area of focus. The 21 papers (2 invited, 19 other contributions) presented at the conference and in the present volume were selected from more than 30 submissions. Every submission was fully reviewed by 3 members of the program committee.

We wish to thank all those who made the meeting possible: the authors for submitting papers, the members of the program committee for their excellent work, and the two invited speakers. We wish to express our sincere appreciation to Regine Gabler (University of Leipzig) and Barbara Hamann (Technical University, Ilmenau) for their great efforts and perfect work concerning the administrative details associated with the workshop and the preparation of this volume. Finally, we wish to acknowledge the substantial help provided by our sponsors: the University of Leipzig, the Technical University, Ilmenau, and the TKK (Techniker Krankenkasse) Leipzig.

August 2003

Thomas Böhme
Gerhard Heyer
Herwig Unger

Organization

IICS 2003 was organized by the Gesellschaft für Informatik (GI) in Germany.

Steering Committee

T. Böhme
G. Heyer (Local Chair)
A. Mikler
H. Unger

Program Committee

A. Brandstädt
J. Brooke
M. Bui
N. Deo
M. Dietzfelbinger
K.-P. Fähnrich
T. Haupt
N. Kalyaniwalla
P. Kropf
M. Kunde
R. Liskowsky
S. Lukosch
A. Pears
U. Quasthoff
M.A.R. Dantas
F. Ramos
A. Ryjov
D. Tavangarian
D. Tutsch
T. Ungerer

Sponsoring Institutions

University of Leipzig
Technical University, Ilmenau
TKK (Techniker Krankenkasse)

Table of Contents

Evaluation of a Communication Environment for High-Speed Mobile Wireless LAN Clients 1
A. Bengsch, H. Kopp, A. Petry, D. Tavangarian

Small Worlds of Concepts and Other Principles of Semantic Search 10
S. Bordag, G. Heyer, U. Quasthoff

Communities in Graphs 20
M. Brinkmeier

Randomly Distributed Tasks in Bounded Time 36
A. Bui, M. Bui, D. Sohier

e-Learning and Communities, Supporting the Circulation of Knowledge Pieces 48
G. Eichler

Where to Start Browsing the Web? 65
D. Fogaras

Small-World Networks Revisited 80
T. Fuhrmann

Message-Oriented-Middleware in a Distributed Environment 93
S. Goel, H. Sharda, D. Taniar

Recursive Node Similarity in Networked Information Spaces 104
J.P. Grossman

Mobile Business Processes 114
V. Gruhn, M. Book

A Distributed Communication System for Integration with Logistics Software 134
V. Gruhn, M. Hülder, R. Ijioui, L. Schöpe

Fault Tolerant Geographical Addressing 144
D. Heutelbeck, R. Räth, C. Unger

Improving Web Server Performance by Distributing Web Applications . . . 156
M. Kakaiya, S. Goel, H. Sharda, D. Taniar

On How to Model Content Engineering in a Semantic Web Environment . 168
L. Maicher

Matching Algorithms in Alternative File-Sharing Systems to Find New Users Having New Content 180
J. Nützel

A Fuzzy Kernel-Based Method for Real-Time Network Intrusion Detection 189
M. Petrovskiy

Testing Protocol Robustness 201
A. Rollet, H. Fouchal

Semantic Geocast Using a Self-organizing Infrastructure 216
J. Roth

Linguistic Knowledge Services – Developing Web Services in Language Technology 229
F. Schmidt, C. Wolff

The Application of XML Templates for Task Preparation in an Electronic Memory Aid System 239
A. Voinikonis, K. Irmscher

Aggregation Transformation of XML Schemas to Object-Relational Databases 251
N.D. Widjaya, D. Taniar, J.W. Rahayu

Author Index 263

Evaluation of a Communication Environment for High-Speed Mobile Wireless LAN Clients

Andreas Bengsch, Heiko Kopp, Andr Petry, Djamshid Tavangarian

University of Rostock
Department of Computer Science
Chair of Computer Architecture
Albert-Einstein-Str. 21
Germany, 18059 Rostock
{andreas.bengsch,heiko.kopp,andre.petry,
djamshid.tavangarian}@informatik.uni-rostock.de

Abstract. The use of networks, especially wireless networks, has been constantly increasing for some years. The use of Wireless LAN was however so far limited to the installation of fixed Hot Spots, especially at airports or train stations. This article exemplifies the accomplished experiments with the use of Wireless LAN IEEE 802.11b Standard in mobile systems, e.g. in driving cars. The most important result is, that slight optimisations of Wireless LAN components can achieve an acceptable communication up to at least 90 km/h.

1 Introduction

Wireless LAN based on the IEEE Standard 802.11 is together with its supplement 802.11b one of the brightest areas of communication business [1]. Beside the use as PC-Card in notebook computers a wide range of user devices, such as PDAs use build in Wireless LAN features. The majority of today s wireless communication is based on IEEE 802.11b, standardized as supplement to the initial 802.11 [3] standard. Changes applied to the supplement reflect the used modulation technique as well as the maximum available bandwidth which increased to 11 MBit/s. However, the used frequency band is still the *Industrial, Scientific and Medical* (ISM) band at 2.4 GHz IEEE: .

When speaking about WLAN often words like client, hot-spot, access point and others are used without definition. A client inside a Wireless LAN can be defined as the consuming mobile device integrated into a wireless network, like a notebook, a desktop computer or a PDA. It contains a wireless device available e.g. as PC-Card, PCI-Interface-Card or even directly build-in. A client uses services provided by the network or provides services itself. The IEEE 802.11 standard describes a network topology referred to as ad-hoc network, where only clients are needed to build up the network. Major drawback of these kind of networks is the shared wireless medium decreasing the amount of clients that are able to communicate, because inside a ad-hoc network, all communication is done point-to-point from one client to another. Furthermore no communication is possible for hidden stations - where two clients

T. Bh me, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 1-9, 2003.

willing to communicate cannot do so, because they are out of range to each other while a third station lies between them and no routing is done using the third station.

To solve those major drawbacks, the wireless standard describes a second possible network topology - the infrastructure mode. Thereby the communication topology changes from point-to-point communication to star-based communication with a special wireless device, called access point and placed in the star's centre. Like a client the access point contains wireless hardware, but its task is to handle each communication occurring inside its spanned wireless cell. All clients willing to communicate inside an infrastructure network must communicate with the corresponding access point only.

Infrastructure mode cells can be combined to cover bigger areas as possible by single access points. A great advantage for clients is the facility of a seamless roaming between these cells. Two concerned access points handle the hand-over of packets from and two the client during the roaming using the *Inter Access Point Protocol* (IAAP). Recently congregations of infrastructure mode cells have been defined as so called hot spots. A hotspot can be referred to as an area that provides one or more wireless networks, independent of the amount of used access points. Hotspots can be categorised according their relation to a client as shown in figure 1.

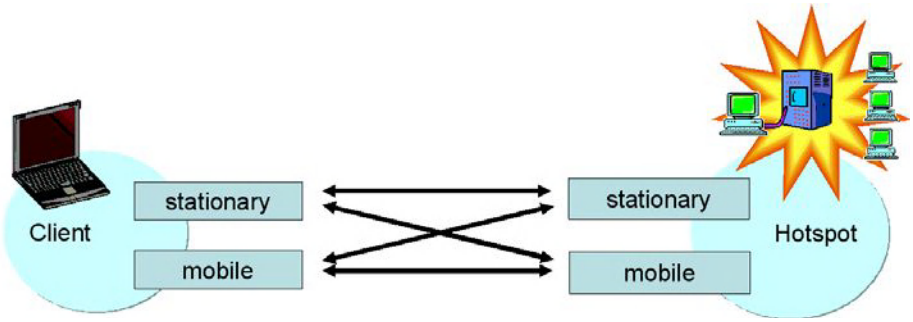


Figure 1. Possible Combinations of Client and Hotspot

a) both stationary (speed = 0 km/h), b) both mobile (same speed), c) Client stationary (speed = 0 km/h), Hotspot mobile (different speed), d) Client mobile (different speed), Hotspot stationary (speed = 0 km/h)

Within the context of this work an evaluation is presented that shows whether a successful use of a mobile client inside a stationary hotspot (d) is possible or not, like Table 1 shows for different wireless technologies like GSM, DECT or HiperLAN.

To evaluate the possible user velocity for Wireless LAN, different scenarios as described in section 2 have been carried out. The measurements have been done with software as explained in section 3. Section 4 analyses the measured results while section 5 summarizes the evaluation and gives a conclusion as well as perspectives for further tests.

Standard	GSM	DECT	WLAN 802.11b	HiperLAN
Band	900/1800/1900 MHz	1900 MHz	2,4 GHz (DSSS)	5,15-5,25 GHz
Bit Rate	270,8 KBit/s	1152 KBit/s	1-11 MBit/s	> 20 Mbit/s
Number of Carriers	125	10	14 at max (regionally different)	-
Channels per Carrier	8	24 (12 duplex)	variable	-
Cell Size	< 35 km	< 400 m	Indoor < 50 m Outdoor < 500 m	Indoor < 50 m Outdoor < 500 m
User Velocity	< 250 km/h	< 50 km/h	To be evaluated	< 10 m/s

Table 1. Overview on Wireless Technologies

2 Scenarios and Experimental Setup

To evaluate the behaviour of Wireless LAN at high speed, two scenarios were developed. They are mainly distinguished by the amount of access points used. While in the first scenario a single access point was positioned in the centre of a test range, the second scenario used two access points building an hotspot to evaluate influences of roaming processes.

To get a comparison basis for later measurements, all transmission parameters (e.g. signal level, noise level and signal-to-noise ratio) as well as the transfer rate have been measured at fixed points of the test range.

For our testing, the client (a notebook) has been integrated into a car. The car went along the test range with a constant speed and for each tested velocity the measurements where repeated several times. The speed has been increased in steps of 10 km/h.

In the second scenario the primarily focus lied on the analysis of the roaming mechanisms provided by the two installed access points.

2.1 The Experimental Setup

Since Wireless LAN is based on electromagnetic waves, physical effects arise, which can impair the efficiency of data communication processes substantially. Among these damping, dispersion and reflection may lead to a higher path loss and signal variations called shadow fading, as well as multiple versions of a signal that arrive at the receiver at different times which is referred to as multi-path fading. According to minimize these effects, a test range was chosen, which is straight, has less buildings nearby and a less public traffic to achieve the maximum speed of 100 km/h. To work with even conditions, the rage has been measured as described in section 2 to determine the measurement start and end points.

Due to the assumption of a cell range of approximately 300 m outside of buildings for Wireless LAN, the test range has been defined with a distance of 600 m altogether. Reflecting to the first described scenario, a single access point has been

placed directly in the middle of the range and connected to a laptop running the server part of the measurement application. To accomplish the second scenario each of the access points has been placed 200 m from the centre. As shown in Figure 2 one of the access point is connected directly to the laptop with the server process while the other has been connected using a separate wireless bridge. All access points as well as the client inside the car have been equipped with di-pol antennas increasing the emitted output by 2.5 dbi. The client side antenna has been installed outside of the car to reduce reflections of the car body.

3 Software

To evaluate a wireless communication system different parameters of the connection must be taken into account as well as the real transmission of data which depends indirectly on the quality of the signal, because it might be high, but due to a high path loss during a transmission, the data throughput can be rather low.

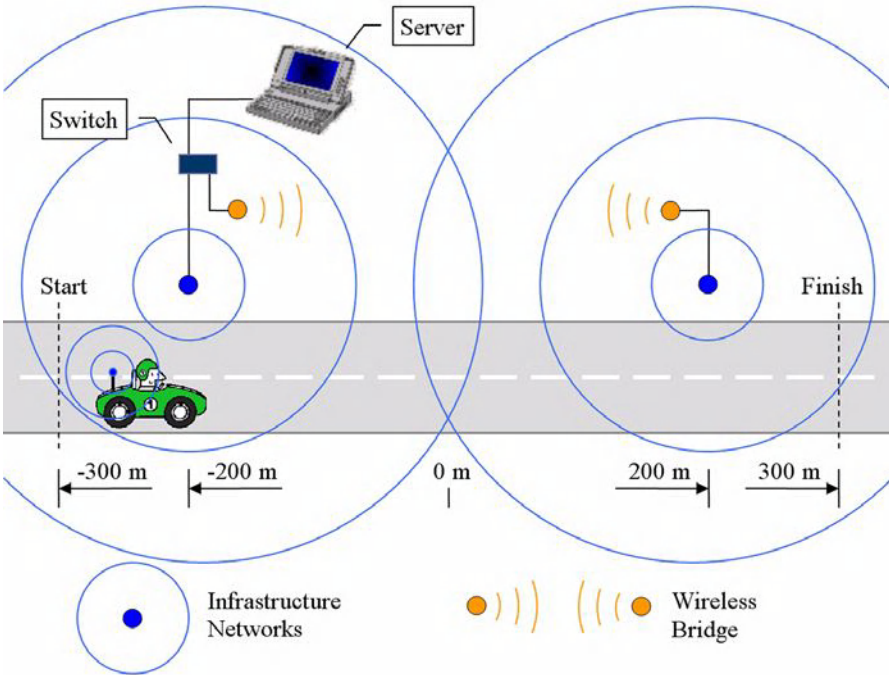


Figure 2. Experimental Setup for Scenario 2

To continuously measure the wireless parameter that reflect the signal quality such as signal level, noise level and the derived signal-to-noise-ratio (SNR) a software tool provided by the Wireless LAN [6] equipment manufacturer, the RoamAbout Client Utility by Enterasys [7] has been used. The tool is able to log those values mentioned before during the movement along the test range every 0,2566 seconds at

lowest which is suitable for a sufficient number of values even for a velocity of 100 km/h.

To measure the data throughput a test suite containing a server-side and a client-side module was implemented, which uses simple TCP/IP socket connections to send out packets of a variable length (between 1,028 byte and 64 KByte with a default of 4,096 byte) from the client to the server. As the server receives the data it sends them back immediately in analogy to a Round-Robin-Algorithm. The collected information containing the number of packages send and in return received again was used to determine the gross data trough-put while moving along the test range.

4 Measurements

The following sections describe the different measurements and analyse the received results for fixed clients as well as the two described scenarios.

4.1 Fixed Clients

To better analyse the measured values while moving along the test range with a constant velocity, a measuring took place with static clients before. Table 2 shows different distances between the single access point located in the centre of the test range and the client. It can be seen, that even in a distance of 300 m the average data throughput was not decreased by any kind of influences. It is likewise outstanding, that a very low level of the noise value results out of the uniform process of the signal-to-noise ratio. This led to a possible data throughput of 11 Mbit/s net even 300 m away from the access point. Thus a single access point allows coverage of the overall testing range of 600 m.

Distance in m	Noise Strength in dB	Signal-to-Noise Ration in dB	Average Transmission Rate in Byte/s
50	-96,6	30,4	427,000
100	-97,2	26,4	401,000
150	-97,4	18,2	406,000
200	-97,4	13,2	404,000
250	-97,4	16,0	406,000
300	-97,0	6,4	435,000

Table 2. Measurements for Transmittion Quality, Signal-to-Noise-Ratio (SNR) and Average Transmission Rate of a fixed Wireless LAN client.

4.2 Mobile Clients

4.2.1 Scenario I: Single Access Point

In order to record the data throughput depending on the speed systematically, the measurements have been started at 5 km/h and increased gradually. The tests were carried out up to a speed of 90 km/h with up to five measurements for each speed. Both, figure 3 and figure 4 show, that the overall data throughput remained constant and is not depending on the aimed speed. The break-ins in the measurement occurred mostly at the same position about 100 m beside the access points. As the used di-pol

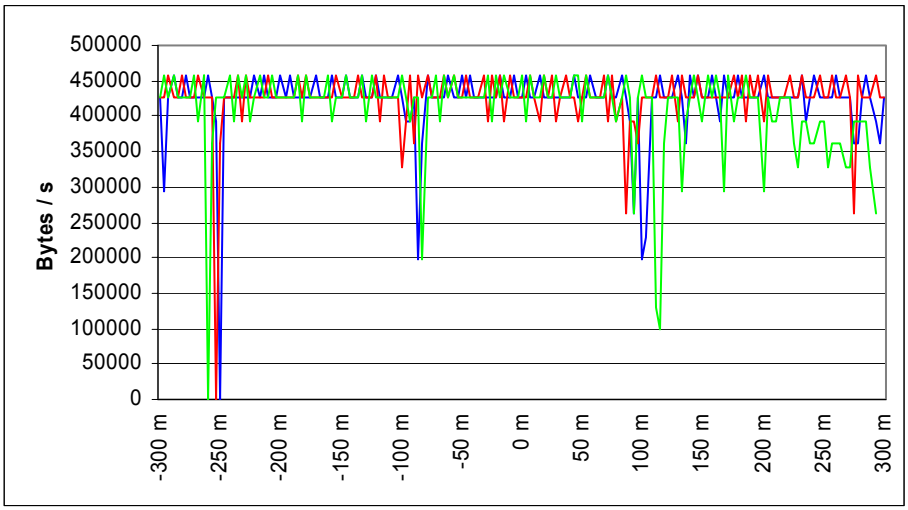


Figure 3. Data throughput at 50 km/h in the first scenario using a single access point

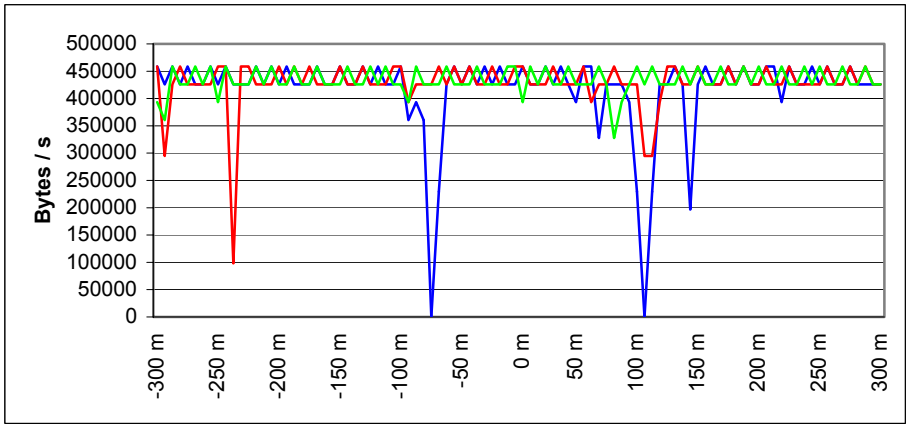


Figure 4. Data throughput at 90 km/h in the first scenario using a single access point

antennas lead to a formation of so-called main and minor lobes, non-uniform radiation occurred, leading to 0 Byte/s values in the figures. Due to the coarse-grained timer (of granularity 500 ms) used in TCP implementations, loss detection latencies are very high. This latency allows sufficient time at the link layer to recover from break-ins and thus no collapse of the data transfer occurred. The very high data throughput was surprisingly stable over the entire testing range. A wireless communication using IEEE 802.11b is therefore possible for speeds up to 90 km/h in general.

4.2.2 Scenario II: Two Access Points

Corresponding to the good results with a single access point in the first scenario, the results of the second scenario are shown in figure 5 and figure 6. The data throughput shown is as already seen in the first scenario relatively constant, but the roaming between the different access point cells is clearly recognizable. The data transmission rate dropped down during the roaming but no link loss could be recognized at all. In case the roaming took place between the access point directly connected to the server and the access point connected via a wireless bridge the data throughput decreased about 100 KByte/s due to the additional wireless communication. Since the data transfer took place over three access points, the latency increased about 50%.

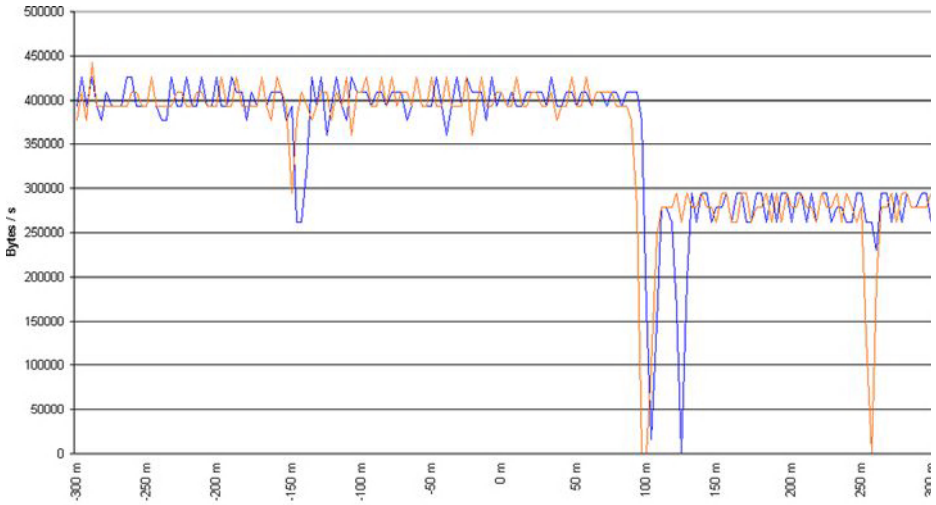


Figure 5. Data throughput at 50 km/h in the second scenario using two access points

The collapse as shown in during the roaming phase was always shorter than a second. The technical specifications of the wireless equipment manufacturer indicates a roaming time of approximately 300 msCabletron Systems: . Due to the used measurement facilities, this time could not be verified during the tests. The second scenario showed, that wireless roaming operates successful even at clients moving with 90 km/h. Again, the coarse-grained timers of TCP allowed the link layer to recover the connection during the roaming.

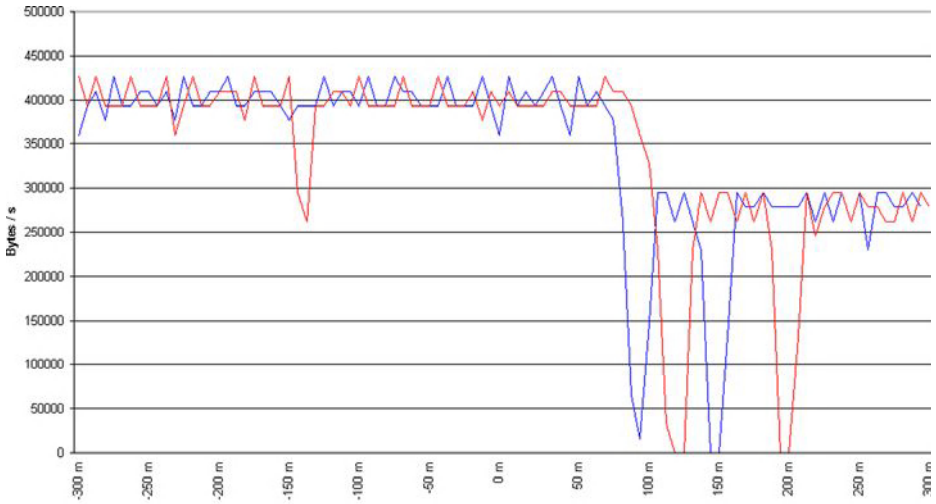


Figure 6. Data throughput at 90 km/h in the second scenario using two access points

5 Conclusions and Perspectives

The previously described scenarios show, that wireless communication using WLAN IEEE 802.11b is even possible at speeds up to 90 km/h. Although drop downs of the data through-put occurred, they did not lead to connection loss nor decreased it significantly. It can be assumed, that TCPs timer to detect a connection loss are that coarse-grained to allow the link layer of the Wireless LAN to recover. The second scenario uses two access points to test wireless roaming. However, the obtained results will not significantly change for more than two access points. If already inside a wireless network, a client will take a seamless hand-over from one access point to another.

Another fact our tests discovered is, that the cell range of about 300 m stated by the manufacturer for a single access point is very pessimistic. The test range of about 600 m could be covered by a single access point as well (as seen in the first scenario).

Taking the results into account, different scenarios for use of wireless communication can be given. For example a non-continuous network along a highway would be suitable for instant messaging or information upload, such as news or traffic information while moving along the highway. Those scenarios have in common, that there is no need for continuous network access. This leads to the fact, that the used client software must be able to handle different IP addresses as it will be less practicable to have all network stations, e.g. along a highway, use the same network. Current systems do not cover such scenarios, so a seamless roaming would be possible only in case the wireless network is open and thus new arriving clients will recognize the network in combination with a dynamic IP system like DHCP. Because the client losses its connection when leaving a hot spot, a re-authentication

must be done when arriving in the next. Standard applications like web clients will not be affected by changing IP addresses as long as no secure connections exist.

The made tests cover mobile clients with speeds up to 90 km/h. Further tests, planned in the nearer future will be made to test mobile clients at speeds up to 200 km/h. Those tests will use special hardware with decreased roaming time and are done along a highway to achieve the mentioned velocities. Additionally, the use of parallel clients as well as Voice over IP will be tested.

6 References

[1] Wife (Wireless Fidelity); http://www.wi-fi.org

[2] Anonymous: IEEE 802.11Working Group. http://grouper.ieee.org/groups/802/11/index.html.

[3] IEEE: *IEEE Std 802.11, 1999 Edition*, 1999

[4] IEEE: *IEEE Std 802.11b, 1999 Edition*, 1999

[5] SANTAMARIA, A., L□PEZ-HERN□NDES, F. J: *Wireless LAN - Standards and Applications*, Artech House, Inc., 2001

[6] Cabletron Systems: *RoamAbout Access Point 2000 Hardware Installation Guide*, 2000

[7] Cabletron Systems: *RoamAbout 802.11 Wireless Networking Guide*, 2000

[8] GEIER, J.: *802.11 MAC layer Defined*, http://www.80211-planet.com/tutorials/article.php/1216351

[9] Anonymous: *WLANA - Learning Center,* http://www.wlana.org/learning_center.html.

[10] GEIER, J.: *802.11 Beacons Revealed*, http://www.80211-planet.com/tutorials/article.php/1492071

Small Worlds of Concepts and Other Principles of Semantic Search

Stefan Bordag, Gerhard Heyer, Uwe Quasthoff

Leipzig University Computer Science Institute,
Natural Language Processing Department
Augustusplatz 10 / 11 D-04109 Leipzig
{bordag, heyer, quasthoff}@informatik.uni-leipzig.de

Abstract. A combination of the strengths of both classic information retrieval with the distributed approach of P2P networks can avoid both their weaknesses: The organisation of document collections relevant for special communities allows both high coverage and quick access. We present a theoretical framework in which the semantic structure between words can be deduced from a document collection. This structural knowledge can then be used to connect document collections to communities based on their content.

Introduction

Comprising more than 3 billion documents, the WWW today contains the fastest growing collection of text. From the user's perspective, the traditional information retrieval (IR) problem of finding relevant documents that satisfy a particular description, has therefore become very urgent again..

A document is searched by locating all available (and indexed) documents that satisfy the search query description. In general, however, the set of retrieved documents does not consist of all and only the relevant documents, so that most likely the query needs to be repeated by a modified search. The actual process of generating a suitable description in a real life is frequently accelerated, or even made possible, by recurring to the semantic knowledge that is implicitly stored in the structure of human society or communities. In his famous article on small worlds [Milgram 1967], has succinctly drawn attention to the fact that passing an information from a source to some unknown addressee crucially depends on exploiting the semantic knowledge implicit in the description of the addressee to drastically reduce the search space for passing the message from one set of acquaintances to the next. In social reality, we apparently rely very heavily upon knowledge about which persons or communities are related to or engaged with which topics. It is the intention of this paper to explore how this implicit relation between the structure of a community and the structure of contents can be made more precise and exploited for the purposes of a semantic search.

Technically, advanced information retrieval systems like Gnutella, FreeNet, or NeuroNet, are based on the peer-to-peer (P2P) approach. This approach consists of a set of similar or compatible software agents that live on a network of connected computers (paradigmatically the internet). Each software agent can likewise act as client

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 10-19, 2003.

and as server, and is accordingly called a servent. Each servent comprises a data base with the IP-address of its neighbours that host servants belonging to the same system. At present, P2P systems are mainly used as filesharing systems; semantic principles for processing queries as sketched above have not been pursued so far.

Words, Document Collections, Link Structure, and Communities

If we define a community as a group of people sharing some common interest, there should be a collection of documents which is of interest to all of them. If, moreover, the documents are available in the web, some of them will be linked via hyperlinks. Many of these documents will contain some words or phrases which are specific to the interest of this community. Any meaningful classification of words can help to classify documents and hence, to identify communities.

However, any such classification has to be deal with polysemy and ambiguity. These properties are inherited from natural language, but also appear in documents (because a document can adress more than one interest) and members of a community (because they can have multiple interests).

In contrast to this common difficulties we also find the following common structural feature: Members of communities, the hyperlink structure of the web [Barabasi 2000] and words according to their semantics [Ferrero 2001] all form so-called small worlds [Strogatz 1998]. These similarities are used to describe a framework how to extend a classification of words to find communities in the web.

Semantic Structures

To motivate the classification of words we start with structuralist semantics and put these relations in a statistical context.

Structuralist Semantics

Our main thesis is that the structure of a content can be derived from a set of documents produced and exchanged within a community. Following the famous Swiss linguist Ferdinand de Saussure, meaning (and other notions of linguistic description) can be defined solely by reference to the structural relations existing amongst the words of a language [Saussure 1916]. Syntagmatic and paradigmatic relations between words constitute the basis for such relations.

Examples of syntagmatic relations typically include dependencies between nouns and verbs, enumerations or the compounding of nouns and nouns, and head-modifier constructions based on adjectives and nouns or nouns and nouns. Paradigmatic relations vary depending on the measure of similarity presumed. On the syntactic level, paradigmatic relations typically comprise distribution classes for the main syntactic categories. On the semantic level, paradigmatic relations range from semantic fields

to well defined logical relations such as hyponymy, co-hyponymy, hyperonyms, synonyms and antonyms.

Let L be a natural language and W be the set of full form words of this language.

Then any sentence S of L represents a sequence of word forms $S = \{w_1,...,w_i,...,w_n\}$ with all $w_k \in W$.

By the *context* of a word form $w_i \in S$ we mean a subset of all word forms occurring in S suitably chosen, i. e. a set of word form contained in the power set of *S*:

$$K_S(w_i) \subset \{w_1,...,w_i,...,w_n\}.$$

Usually this subset will contain the meaningful words according to some statistical measure to be defined later. Similarily, the exact meaning of the *most*-operator $M\ M$ and the set similarity ~ used below have to be defined.

Abstract *syntagmatic* and *paradigmatic* relations of two word forms w_i and w_k can now be defined as follows:

Common joint appearance of two word forms w_i and w_k defines the abstract *syntagmatic relation SYN*: Two word forms w_i and w_k are related syntagmatically if most of the contexts of w_i contain the word w_k:

$$(M\ S : w_i \in S)(w_k \in K_S(w_i) \rightarrow SYN(w_i, w_k))$$

Joint context shared by two word forms w_i und w_k defines the abstract *paradigmatic relation*: Two word forms w_i and w_k are related paradigmatically if they usually appear within similar contexts:

$$(M\ S : w_i \in S\ \exists K(w_k))((K_S(w_i) \sim K(w_k)) \rightarrow PARA(w_i, w)).$$

Co-occurrences

Some words co-occur with certain other words with a significantly higher probability and this co-occurrence is semantically indicative. We call the occurrence of two or more words within a well-defined unit of information (sentence, document) a *collocation*. For the selection of meaningful and significant collocations, an adequate collocation measure has to be defined.

Let a, b be the number of sentences containing A and B, k be the number of sentences containing both A and B, and n be the total number of sentences.

Our significance measure calculates the probability of joint occurrence of rare events. The results of this measure are similar to the *log-likelihood*-measure:

Let $x = ab/n$ and define:

$$sig(A,B) = \frac{-\log\left(1 - e^{-x}\sum_{i=0}^{k-1}\frac{1}{i!}\cdot x^i\right)}{\log n}$$

For $2x < k$, we get the following approximation, which is much easier to calculate:

$$sig(A,B) = \frac{\left(x - k\log x + \log k!\right)}{\log n}$$

In general, this measure yields semantically acceptable collocation sets for values above an empirically determined positive threshold. Hence, we can use this measure to select the relevant words in a sentence and to determine the context of a word as described in the above section.

Example: *space*

Fig. 1 shows the collocations of the word space. Two words are connected if they are collocations of each other. The graph is drawn useing *simulated annealing* (see [Davidson 1996]). Line thickness represents the significance of the collocation. The resulting picture represents semantic connectedness surprisingly well. In Fig. 1 we find three different meanings depicted: real estate, computer hardware, and astronautics.

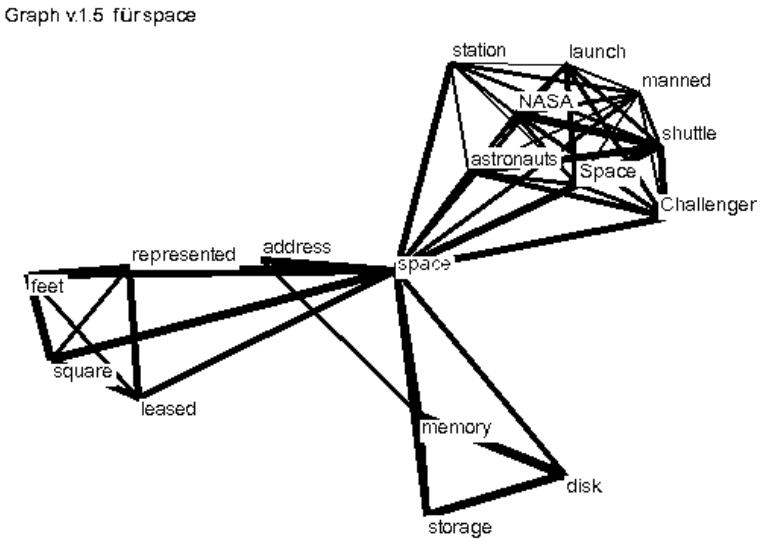


Fig. 1. Collocation Graph for space

The connection between *address* and *memory* results from the fact that address is another polysemous concept. This kind of visualization technique can also be applied to knowledge management techniques, esp. the generation and visualization of Topic Maps [Böhm 2002].

Small Worlds

It is worth to take a closer look at the graph which is implicitly defined by the co-occurrence measure. Let $G = (N, C)$ be a graph with N as the set of nodes and C the set of edges between the nodes. The nodes are labelled by different words of our language, we only consider words having collocations as described above. Two nodes are connected with each other if they are collocations. The resulting graph has several important properties that will be briefly described:

- The graph is *nearly fully connected*, i.e. for two randomly chosen nodes there is a high probability that they are connected in this graph.
- It is *sparse*. Usually the number of edges is approximately only one order higher than the number of nodes.
- It has the *small world* property.

A graph having the *small world* property is a graph which has both short average path lengths like a random graph, as well as high local clustering coefficients like a regular graph. First formalizations and the explicit differentiation of the small world property of graphs contrasting the traditional extremes, the regular and random graphs, have been introduced by [Strogatz 1998]. Further work can be found at [Kleinberg 2000] and [Ferrero 2001]. In short, a *random graph* is a graph where nodes are connected randomly, and which has a given degree distribution of the nodes which can, for example, be power-law or exponential. A *regular graph* is a graph where all nodes have the same number of connections to other nodes.

The *path length* between two nodes in the graph is denoted by $d_G(i, j)$ with $i, j \in N$ and measures how many connections part the two given nodes at least. The average path length d_G over the graph is then calculated as the arithmetical mean over all possible distances:

$$d_\Omega = \frac{1}{|N|} \sum_{i=|N|}^{1} \left(\frac{1}{i} \sum_{j=1}^{i} d_G(i, j) \right)$$

The *clustering coefficient* c_i of a node i compares the number of connections T_{Γ_i} between the neighbours Γ_i of the given node with the number of possible connections:

$$c_i = \frac{2T_{\Gamma_i}}{|\Gamma_i| \cdot (|\Gamma_i| - 1)}$$

For the whole graph, the clustering coefficient c_G can then be calculated as a mean over the clustering coefficients of each node in the graph:

$$c_\Omega = \frac{1}{|N|} \sum_{i=1}^{|N|} c_i$$

The clustering coefficient thus measures the probability that two nodes are connected with each other if they are connected to a common third node.

A comparison of the two graphs indicates that a random graph will always have much shorter path lengths than a regular graph if both have the same number of nodes and connections. If graphs are sparse graphs, then with growing graph size the path length will grow linearly for the regular graph and only logarithmically for the random graph because of the many possible shortcuts throughout the entire graph. On the other hand, the clustering coefficient will always be very low for the random graph as opposed to the regular graph.

The small-world property has originally been used to explain why certain random graph based models resulted in wrong predictions about phenomena like disease spreading. In these cases, the high neighborhood clustering together with the short path lengths lead to a much more efficient model of graph formation when compared to random graphs. It is the intention of the present paper to draw attention to the fact that similar phenomena of small world formation can be detected with respect to the conceptual space of internet communities, and that this fact might be exploited for implementing more efficient, semantically based search strategies.

Disambiguation

Based on the co-occurrences of word forms and the small-world property of their collocations graph, an approach to solve the polysemy problem has been introduced by [Bordag 2002b]. Applications include improved text classification methods, improvements in Word Sense Disambiguation algorithms, better query expansion, intelligent spell checking and more.

The algorithm is based on two assumptions: first, words in the graph cluster semantically and, second, any three given words are unambiguous (there are only few cases where this does not hold). If three words are semantically homogenous, they are located in the same cluster of the graph. The intersection of their direct neighbours will not be empty, and they will be semantically homogeneous as well. After generating an amount of such triplets (always including the input word), their neighbour-intersections are clustered with hierarchical agglomerative clustering.

As a result, for a given word one or more sets of semantically homogeneous words are found along with a set of words which are either semantically unrelated to the input word (although they are co-occurring with it), or whose statistical count is not high enough to make a reliable decision. Problems occur when a corpus is unbalanced with respect to certain sub-languages where certain usage contexts of a word are missing.

Graph Based Automatic Semantic Convergence (ASC)

Combining the algorithm sketched above with known methods and linguistic data resources, we introduce a first framework for a semantically based search, aiming at a system by which a variety of textual information can be processed in a fully unsupervised manner.

Instead of keeping a central index of the content of a WEB network, we propose agents analogous to the common P2P agents. But unlike agents that simply collect a set of IP addresses of neighbours in the network and broadcast queries to them, we propose *dynamic* agents. These dynamic agents should be able to decide intelligently what to do with a query, either answer it based on the documents available to the agent, or forward it to an agent which would better satisfy the query.

Abstract Definition of the Convergence Framework

Basically, an ASC-agent consists of a set of documents, the related semantic knowledge database, and a set of IP addresses of neighbours where other agents with the same interface reside. The lifecycle of an agent consists of periodic comparisons of its knowledge database with those of its neighbours. After such a comparison it is decided which of its neighbours has the least semantically fitting content, and the worst ones are dropped in favor of better ones. How exactly the new neighbours are chosen, or how many are dropped, and many other options can be left as implementation parameters and may easily differ from one agent to another to tune agents to specific needs. Nevertheless, an agent should be required to reserve a small fraction of its connections to agents that have a large number of outgoing links. This corresponds to small worlds where a subclass differs from others in that it has only a few hub-like nodes that connect to large parts of the network at once. It can also be left open whether the connections should be symmetrical or directed.

We begin by defining an agent A as a sixtuple $A = (\Gamma, \Delta, \phi, \Theta, c, a)$ where

Γ	set of neighbours
Δ	set of owned content (i.e. a set of Documents)
$\phi : (A_1, A_2) \rightarrow [0..1]$	node similarity operation $[0..1]$
$\Theta : A \rightarrow A'$	convergence operation
c	connectivity threshold
a	activity threshold

Both the node similarity operation ϕ and the convergence operation Θ are left unspecified although some possibilities will be discussed.

For the similarity operation, any traditional document comparison model can be used but there are two important constraints: First, the set of all documents is always unknown (even its size) and, second, the set of terms used in these documents is unknown as well. That means that for example the Vector Space Model will have to be built from scratch from Γ_{A_1} and Γ_{A_2} dynamically for each comparison.

The convergence operation is more intricate as it might have great impact on the overall behavior of the network. Two main possibilities emerge at this point. The first has already been mentioned:

- Compare own content to content of neighbours
- Drop some of the worst ones
- Replace them with better ones
- Keep a fraction of connections to highly connected nodes, no matter how bad they fit semantically

The second one is more radical:

- Use Text classification algorithms to build clusters from contents of neighbours including own content
- Replace the worst cluster with neighbours of the best cluster and a fraction of random connections with highly connected nodes

In the second case, a node might enter a state where it would have to remove itself along with the worst cluster. It would then have to start again completely randomly at a different place in the network to perform better than the first time. Improvements can be imagined by making use of links found in the own document set and trying to use them to find relevant agents directly, speeding up the semantic convergence.

Another important aspect of such document-representing agents is that they can be inherently ambigue themselves. They will certainly contain documents from more than one topic. This means that they will have to be able to handle this properly. Here is where the idea of above described disambiguation algorithm can be reused. As it is based on a very similar construct, a sparse graph having clusters, it should be possible to alter it in order to fit it to this task. As such it will provide a robust unsupervised clustering of the topics in the local document collection.

Research on the small-world properties of graphs indicates that the above network is most likely to converge to clusters of agents with similar content, exhibiting the small-world property. From that follows that queries, once they are handed over to an agent in such a cluster, are either answered immediately or by handing them just one or two steps further to the best possible agent without broadcast. In case that a query begins with a completely unfitting agent, the agent decides to hand it over to the agent which has the highest connectivity trusting that this new agent will have a connection to a distant agent which might be more fitting than anything it had itself. The short path lengths in this network will have the effect that a search query, although never broadcasted, will not have to travel far until it reaches its destination.

Conclusion

By our approach, the role of the classic P2P agent changes in that it is not only a mechanic collection of links and files. The network comprised by the semantic agents sketched above evolves on the basis of the content on which they ‘reside’. In a sense, the agents are not only aware of where they are but also of what they represent. It is important that all components of such agents are well known and robust algorithms and methods. The most important aspect, however, is that a user who decides to participate in a network by installing an agent has to do nothing except pointing the agent to the documents it should represent. This is in contrast to current WEB projects like semantic web where users are encouraged to improve the quality of the WEB and its services by providing manually created metadata for their data.

References

[Adar 2000] E. Adar, B. Hubermann. Freeriding on Gnutella. Firstmonday 5(10), 2000

[Barabasi 2000] A.L. Barabasi et al . Scale-free characteristics of random networks: the topology of the World-wide web, Physica A (281)70-77, 2000

[Bolloba 1985] B. Bolloba. Random Graphs, Academic Press, London, 1985

[Bordag 2002b] Stefan Bordag, Sentence Co-occurrences as Small-World Graphs: A solution to Automatic Lexical Disambiguation, A. Gelbukh (Ed.): CICLing 2003, LNCS 2588, pp. 329-332, Springer-Verlag Berlin Heidelberg, 2003

[Bordag 2002a] S. Bordag. Vererbungsalgorithmen von semantischen Eigenschaften auf Assoziationsgraphen und deren Nutzung zur Klassifikation von natürlichsprachlichen Daten, Diplomarbeit, Universität Leipzig, Institut für Mathematik und Informatik, 2002

[Böhm 2002] K.Böhm, G. Heyer, U.Quasthoff, Chr. Wolff. Topic Map Generation using Text Mining, Journal of Universal Computer Science, Vol.8, Issue 6, http://www.jucs.org/jucs_8_6, 2002

[Clarke 2000] I. Clarke, O. Sandberg, B. Wiley, T. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. ICSI Workshop on Design Issues in Anonymity and Unobservability, Berkeley, CA, 2000

[Davidson 1996] R. Davidson, D. Harel. Drawing graphs nicely using simulated annealing. ACM Transactions on Graphics. vol. 15, num. = 4, pp. 301-331. 1996

[Deo 2001] N. Deo, P. Gupta. World Wide web: a Graph Theoretic Approach. Technical Report CS TR-01-001, University of Central Florida, Orlando Fl. USA, 2001

[Ferrero 2001] R. Ferrero i Cancho, R. V. Solé. The Small-World of Human Language. (http://www.santafe.edu/sfi/publications/), 2001

[Gnutella] Gnutella. www.gnutella.com. 2002

[GRACE] GRACE http://www.ub.uni-stuttgart.de/grace/ 2002

[Heyer 2000] G. Heyer, U.Quasthoff, C.Wolff. Aiding Web Searches by Statistical Classification Tools. Knorz, Gerhard; Kuhlen, Rainer (edd.). Informationskompetenz - Basiskompetenz in der Informationsgesellschaft. Proc. 7. Intern. Symposium f. Informationswissenschaft, ISI 2000, Darmstadt. Konstanz: UVK, 163-177, 2000

[Heyer 2001] G. Heyer, U.Quasthoff, T.Wittig, C.Wolff. Learning Relations Using Collocations, A. Maedche, S. Staab, C. Nedellec and E. Hovy, (eds.), Proc. IJCAI Workshop on Ontology Learning, Seattle/ WA, 2001

[Heyer 2002a] G. Heyer, U.Quasthoff, C.Wolff. Automatic Analysis of Large Text Corpora - A Contribution to Structuring WEB Ccommunities, in: H.Unger, Th. Böhme (Hrsg.), Proceedings I2CS - 2002, Advanced Lecture Notes in Computer Science, Springer: Berlin, Heidelberg, New York 2002

[Heyer 2002b] G. Heyer, U.Quasthoff, C.Wolff. Knowledge Extraction from Text: Using Filters on Collocation Sets, LREC-2002 IICS 2002, Lectures Notes in Computer Science 2346, 153 - 162, Springer, 2002

[Joseph 2001a] S. Joseph: NeuroGrid - Freenet Simulation Results.

[Joseph 2001b] S. Joseph: NeuroGrid White Paper.

[Kleinberg 2000] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. Proc. 32nd ACM Symposium on Theory of Computing, 2000

[Lechner 2002] U. Lechner: Peer to Peer beyond Filesharing. In H.Unger, T. Böhme (Hrsg.)

[Lifantsev 2000] M. Lifantsev. Voting Model for Ranking Web Pages, In Peter Graham and Muthucumaru Maheswaran, editors, Proceedings of the International Conference on Internet Computing (Las Vegas, Nevada, U.S.A.), CSREA Press, pages 143-148, Las Vegas, 2000

[Lifantsev and Chiueh 2002] M.Lifantsev and T.Chiueh. I/O-Conscious Data Preparation for Large-Scale Web Search Engines. Proceedings of 28th International Conference on Very Large Data Bases, August 20-23, 2002, Hong Kong, China, Morgan Kaufmann, Hong Kong, 2002

[Milgram 1967] S. Milgram. The small world problem. Psychology Today 2, pp. 60-67, 1967.
[Neurogrid] www.neurogrid.com
[Newman 2000] M.E.J. Newman. Models of the Small World, 2000
[OpenGriD] Open GriD http://www.cs.sunysb.edu/~maxim/OpenGRiD/ 2002
[Ritter 2002] J. Ritter. Why Gnutella can't scale. No, really. www.nearlydeaf.8m.com/ygnutellwnwrk.html. 2002
[Sanderson 1996] M. Sanderson (1996): Word Sense Disambiguation and Information Retrieval; Proceedings of the 17th ACM SIGIR Conference, pp. 142-151
[Saussure 1916] Saussure, F. de Saussure, Cours de linguistique générale, 1916
[Schmidt 1999] F. Schmidt. Automatische Ermittlung semantischer Zusammenhänge lexikalischer Einheiten und deren graphische Darstellung, Diplomarbeit, Universität Leipzig, 1999
[Manning & Schütze 1999] C. D. Manning & H. Schütze. Foundations of statistical natural language processing, 1999.
[Sebastiani 2001] F. Sebastiani. Machine Learning in Automated Text Categorization, 2001.
[Singla 2002] A. Singla, C. Rohrs. Ultrapeers: Another Step towards gnutella scalability. Lime Wire LLC, Working Draft, 2002. www.limewire.com/developrer/Ultrapeers.html
[Steyvers & Tenenbaum 2002] M. Steyvers, J. B. Tenenbaum. The large-scale structure of semantic networks: statistical analyses and a model of semantic growth. M. Steyvers, J. B. Tenenbaum, Cognitive Science, 2002
[Strogatz 1998] D. J. Watts, S.H. Strogatz. Collective dynamics of 'small-world' networks, Nature 393:440-442, 1998.
www.firstmonday.org
www.neurogrid.net/ng-simulation.html (2001).
www.neurogrid.net/WhitePaper0_3.html (2001).

Communities in Graphs

Michael Brinkmeier

Technical University Ilmenau
Faculty of Computer Science and Automation
Institute for Theoretical and Technical Computer Science
mbrinkme@tu-ilmenau.de

Abstract. Many applications, like the retrieval of information from the WWW, require or are improved by the detection of sets of closely related vertices in graphs. Depending on the application, many approaches are possible. In this paper we present a purely graph-theoretical approach, independent of the represented data. Based on the edge-connectivity of subgraphs, a tree of subgraphs is constructed, such that the children of a node are pairwise disjoint and contained in their parent. We describe a polynomial algorithm for the construction of the tree and present two efficient methods for the handling of dangling links vertices of low degree, constructing the correct result in significantly decreased time. Furthermore we give a short description of possible applications in the fields of information retrieval, clustering and graph drawing.

1 Introduction

Intuitively a community in the WWW is a set of inter-linked documents, homepages or servers covering a common topic or which are in some sense closely related to each other. The strength of this relation can be measured in several ways. We focus our attention on the purely graph-theoretical edge-connectivity of subgraphs. Given an undirected, weighted graph G, we define the community of an arbitrary subgraph H, as the largest subgraph of G containing H with maximal edge-connectivity.

We are going to prove, that communities of arbitrary subgraps are unique and completely determined by the communities of vertices and edges. Furthermore we are going to see that the communities in a graph G can be partially ordered, resulting in a natural clustering of G (cmp. [FCE95], [Fen97]). More detailed, we obtain a tree of subgraphs whose leaves are the vertices of G and whose inner node represent the communities. Furthermore the children of a node are disjoint and completely contained in the parent.

In section 4 we are going to extend our definition of communities to k-*communities*, i.e. the largest subgraphs of edge-connectivity at least k.

In addition to a polynomial algorithm, calculating the tree of communities, we are going to provide two efficient methods for the handling of dangling links and vertices of low degree. An experiment on a real-world web-graph shows a dramatical decrease in runtime, which was confirmed by experiments on different types of random graphs.

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 20–35, 2003.

Due to space restriction we omit all proofs and some necessary lemmata. These can be found in the technical report [Bri02]

Related Work

Our communities are very closely related to the k-components introduced by D.W.Matula in [Mat69] and [Mat72]. In fact the community of an edge or vertex x is the k-component with maximal k, containing x. Our k-communities are precisely the k-components. But our approach seems more algorithmic and the description of the resulting structures, i.e. the tree of communities, is different. Furthermore the two efficient methods seem to justify the effort.

Since our approach is very general, the techniques presented here can be applied for many different problems involving the clustering of data. In fact similar approaches are already presented in the literature. For example in [FLG00] Flake et al. use minimal cuts - our main tool - for the detection of sets of vertices, which have at least as many edges inside the set, as outside. But their communities are not unique, and their algorithm requires the knowledge of two vertices in different communities, both having a larger degree than the edge-connectivity of the whole graph.

In [HSL$^+$99] and [HS00] Hartuv and Shamir describe an algorithm which - similar to ours - determines minimal cuts recursively. If one of the resulting two parts has an edge-connectivity larger than half the number of its vertices, then it is interpreted as a cluster. Otherwise it is cut again and its two parts are examined. Unfortunately the resulting clustering is not uniquely determined.

In [Bot93] R. Botafogo extended his former work on the clustering of Hypertexts by biconnected components (cmp. [BS91]) to k-components in the sense of Matula. His aggregates can be constructed from our communities, by removing all subcommunities. The resulting connected components are Botafogo's clusters. In his paper, as in Matulas work, the resulting tree of aggregates is not mentioned. Furthermore he clusters arbitrary graphs by a *stratified clustering* corresponding to our k-communities.

In [GKR98] Gibson et al. and in [KRRT99] Kumar et al. use a different notion of communities, specialized to the WWW. They use directed bipartite cores, i.e. complete bipartite subgraphs, for the characterization of communities. Their approach is based on the consideration that related pages in the WWW do not necessarily have links between them, but are often referenced together. The detected structures are of very localized nature and do not imply a global hierarchy of communities.

Future Work

As already mentioned our approach is very general, resulting in a wide range of possible applications. Each of this applications gives rise to experiments and questions which might be interesting to examine.

The main task we had in mind, was the detection of communities in the WWW. In combination with ranking algorithms based on link-analysis, like

PageRank ([PBMW98]) and HITS ([Kle98]), the results of purely text-based searches may be structured more efficiently and precise by the covered topic and the relation between them. Especially the problem of the results tending to a tightly knit community (TKC) and the drift to more general topics, as observed with HITS ([BH98], [BRRT01], [CDK+99], [Kle98], [Lit]) may be compensated.

Another field of application is clustering in general. Since we use weighted graphs, nearly every clustering problem may be treated with communities. Unfortunately, the communities may be very fragmented if real numbers are used as weights. This is caused by the fact that only slight variations in the edge weights cause different edge-connectivities (or weights of minimal cuts). Here the usage of k-communities or the discrete categorization of edge weights may be appropriate.

A third area of applications lies in the visualization of graphs. The natural clustering, i.e. the tree, may be used for the graphical browsing of trees. Communities may be expanded or collapsed at will, reducing the visible information. Furthermore drawing techniques for clustered graphs may be applied.

2 Communities

Intuitively a community in the World Wide Web consists of a set of highly interlinked pages or sites, i.e. there are more connections between two members of the community than between two arbitrary sites in the WWW. A graph theoretic measure for this property is the *edge-connectivity*. In this section we will use this notion and introduce a definition of communities in an undirected graph. In addition we describe and prove several results regarding the structure and nature of these communities, which will lead to an algorithm.

2.1 Graphs and Edge-Connectivity

Throughout the paper we assume that $G = (V, E)$ is a finite, undirected, multigraph Furthermore we assume that there exists a function $w : E \to \mathbb{R}^+$ of positive edge-weights, i.e. $w(e) > 0$ for each edge e. We denote the sum of weights of edges between u and v with $w(u, v)$, and the sum of the weights of all edges adjacent to u with $w(u)$. Since the weight function does not occur directly, we usually omit it. Nonetheless all results are true for positive edge weights.

A *cut* C of G is a subset of edges, such that $G \backslash C = (V, E \backslash C)$ is disconnected. The weight $w(C)$ of a cut C is the sum of the weights of its edges. The *edge-connectivity* $conn(G)$ of G is the minimal weight of cuts of G. A cut with weight $conn(G)$ is called *minimal cut* of G. We define the edge-connectivity of an isolated vertex to be 0. This has the effect, that an isolated vertex is *not* connected in our sense.

It is a well-known fact that a minimal cut C of a connected graph G separates it into two connected parts G_1 and G_2. The vertex sets of these subgraphs form a partition $\{V_1, V_2\}$ of V. Two vertices u and v belong to the same part, if and only if there exists a path between them, which does not contain an edge of

the minimal cut. We usually identify the set of edges of a cut with the induced partition of the vertex set, or with the pair (G_1, G_2) of induced subgraphs of G[1].

If H is a subgraph of G we denote the sets of vertices and edges of H by $V(H) \subseteq V$ and $E(H) \subseteq E$, resp. For $U \subset V$ the *induced subgraph* $G[U]$ is the graph, whose vertices are the elements of U and whose edges are all edges of G between them.

Lemma 1. *Let H_1 and H_2 be two subgraphs of G such that $H_1 \cap H_2 \neq \emptyset$. Then the induced subgraph $H := G[V(H_1) \cup V(H_2)]$ has an edge-connectivity of at least* $\min(conn(H_1), conn(H_2))$.

2.2 Communities

As said before, we interpret a community as a set of vertices, which are stronger connected to each other than to the rest of the graph. In addition we require a community to contain a given subgraph.

Definition 1. *Let G be a graph and $H \subseteq G$ a subgraph. A* community of H in G *is a subgraph C of G, such that the following properties are satisfied:*

1. $H \subseteq C$.
2. $conn(C) \geq conn(D)$ *for each subgraph* $D \subseteq G$ *with* $H \subseteq D$.
3. $D \subseteq C$ *for each subgraph* $D \subseteq G$ *with* $H \subseteq D$ *and* $conn(D) = conn(C)$.

In other words, a community is the largest subgraph of maximal edge-connectivity among all subgraphs of G containing H.

In the following we are going to prove, that the community of an arbitrary subgraph exists and is uniquely determined. First observe, that two communities C_1 and C_2 of H in G, both have maximal edge-connectivity $k := conn(C_i)$. Therefore the third property of communities leads to $C_1 \subseteq C_2$ and $C_2 \subseteq C_1$. Therefore there can exist at most one community of H in G.

By Lem. 1 the union of all subgraphs containing H with maximal edge-connectivity, is a community of H in G. Hence we obtain the following result.

Theorem 1. *For each subgraph $H \subseteq G$ exists a uniquely determined community $Comm_G(H)$ of H in G. Furthermore $Comm_G(H)$ is an induced subgraph of G.*

The additional property of $Comm_G(H)$ being an induced subgraph is caused by the fact, that the addition of edges increases the edge-connectivity.

Definition 2. *Let $H \subset G = (V, E)$ be a subgraph. The* strength $str_G(H)$ of H in G *is the edge-connectivity of its community, i.e.*

$$str_G(H) := conn(Comm_G(H)).$$

[1] In the literature it is not unusual, that a cut is defined to be a partition of V into two non-empty sets.

Remark 1. Since we assume that an isolated vertex has edge-connectivity 0, our definition of communities causes one problem. In contrast to the intuition, the community of an isolated vertex with self-loops is the whole graph and not the vertex itself. But this is relaxed by the fact, that a disconnected graph itself is only the community of its isolated vertices and all disconnected subgraphs, spread among at least two connected components, as we are going to see later. As a consequence of this observation, we can safely ignore isolated vertices and subgraphs touching more than one connected component. Additionally, isolated vertices and subgraphs touching two or more connected components are not very valuable in the analysis of communities, since they combine completely unrelated vertices.

Theorem 2. *For two subgraphs H_1 and H_2 of G exactly one of the following statements is satisfied:*

1. $Comm_G(H_1) \cap Comm_G(H_2) = \emptyset$
2. $Comm_G(H_1) \subsetneq Comm_G(H_2)$
3. $Comm_G(H_2) \subsetneq Comm_G(H_1)$
4. $Comm_G(H_1) = Comm_G(H_2)$

If G is disconnected and H is completely contained in one connected component $\bar{G}$ of G, then $Comm_G(H) = Comm_{\bar{G}}(H)$, i.e. the community of H is a subgraph of $\bar{G}$. Otherwise the community would have edge-connectivity 0, which is less than $conn(\bar{G}) \geq 1$. This observation allows us to restrict to connected graphs. We only have to check whether the subgraph H is spread among several components of G, or whether it is an isolated vertex with self-loops. In this case $Comm_G(H) = G$ is satisfied. Otherwise $Comm_G(H) = Comm_{\bar{G}}(H)$, as seen before.

Theorem 3. *Let $H \subseteq G$ be a connected subgraph and let (G_1, G_2) be a minimal cut of G such that $H \subseteq G_1$. Then the following statements are equivalent.*

1. $Comm_G(H) \neq G$
2. $Comm_G(H) \subseteq G_1$
3. $Comm_G(H) = Comm_{G_1}(H)$
4. $str_{G_1}(H) > conn(G)$

3 Communities of Vertices and Edges

In this section we are going to prove, that the communities of arbitrary connected subgraphs are completely determined by their vertices and edges. Furthermore the communities of vertices and edges can be represented quite efficiently using trees. Furthermore the representation allows us to calculate the communities of arbitrary subgraphs of a connected graph very easily.

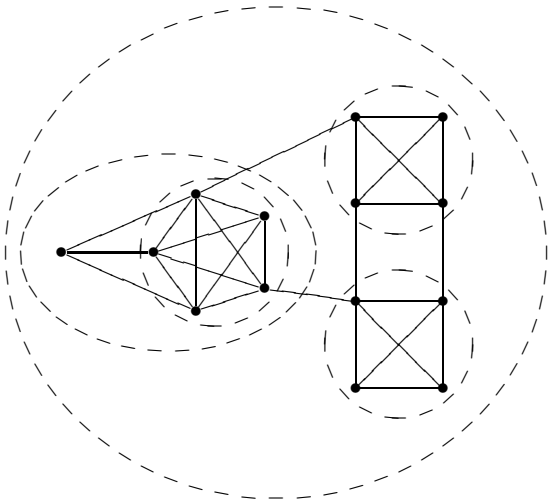


Fig. 1. An example of communities.

3.1 Vertex- and Edge-Communities

Since in the following the communities of edges and vertices will become the most important objects, we will write:

$$Comm_G(v) := Comm_G((\{v\}, \emptyset)) \text{ and } Comm_G(e) := Comm_G((\{u, v\}, \{e\}))$$

for each vertex v and each edge e between u and v.

As proved in [Bri02] we can safely ignore self-loops of vertices and join multiple edges. This allows us to state the main result without bothering about too many special cases.

Theorem 4. *Let $H \subseteq G$ be a connected subgraph containing at least one edge. If $e \in E(H)$ is given with $str_G(e) \leq str_G(e')$ for all $e' \in E(H)$, then $Comm_G(H) = Comm_G(e)$.*

Corollary 1. *Let G be a connected graph. If G has at least one edge, then there exists an edge e, such that $G = Comm_G(e)$. If G contains no edge, i.e. if G consists of one vertex v, we have $Comm_G(v) = G$.*

3.2 Descriptions of Communities

As we have seen, we only need to know the communities of all vertices and edges, to be able to determine the communities of all connected subgraphs. Furthermore Thm. 2 implies that they can be arranged in a tree, as will be made precise in this section.

For the representation of all vertex- and edge-communities of a graph G we use clustered graphs as introduced by Feng et al. in [FCE95] and [Fen97]. The foundation is a directed tree T which has precisely $\|V\|$ leaves together with a bijection $\varphi : \mathbf{leaves}(T) \to V$, associating a unique vertex of G to each leaf of T. We assume that all edges of T are directed from parent to child. The pair (T, φ) is called a *clustering* of G.

We inductively define

$$V_{(T,\varphi)}(n) := \begin{cases} \{\varphi(n)\} & \text{if } n \in \mathbf{leaves}(T) \\ \bigcup\limits_{n \to m} V_{(T,v)}(m) & \text{if } n \notin \mathbf{leaves}(T) \end{cases}$$

and

$$G_{(T,\varphi)}[n] := G\big[V_{(T,\varphi)}(n)\big].$$

Hence $V_{(T,\varphi)}(n)$ is the set of vertices of G, corresponding to leaves of the subtree rooted at n, and $G_{(T,\varphi)}[n]$ is the subgraph induced by this set. As a special case we have $V_{(T,\varphi)}(root(T)) = V$ and $G_{(t,\varphi)}[root(T)] = G$.

For the remainder of the paper we will drop the index (T, φ) to reduce the notational overhead.

Definition 3. *Let G be a graph and (T, φ) a clustering of G.* (T, φ) describes the communities of G *if there exist two maps*

$$vcomm : V \to V(T) \;\; \textit{and} \;\; ecomm : E \to V(T)$$

which map vertices and edges of G to nodes of T, such that

$$G[vcomm(v)] = \mathrm{Comm}_G(v) \;\; \textit{and} \;\; G[ecomm(e)] = \mathrm{Comm}_G(e)$$

for all $v \in V$ and $e \in E$. A description $(T, \varphi, vcomm, ecomm)$ of communities of G is called reduced *if each node n of T satisfies one of the following two conditions:*

1. *n has no children, i.e. is a leaf.*
2. *n has at least two children, and at least one of the following holds:*
   (a) *$vcomm^{-1}(n) \neq \emptyset$,*
   (b) *$ecomm^{-1}(n) \neq \emptyset$ or*
   (c) *n is the root of T.*

Before we proceed to the algorithm, we describe in which way the community of an arbitrary subgraph can be obtained from a reduced description of communities.

Theorem 5. *Let $D = (T, \varphi, vcomm, ecomm)$ be a reduced description of communities of G and H a subgraph of G. If n is the root of the smallest subtree of T containing all leaves l with $\varphi(l) \in V(H)$, then $\mathrm{Comm}_G(H) = G[n]$.*

The reduction of the description results in a tree T with the following properties:

- The leaves of T correspond to the vertices of G.
- Each node n of T, which is neither a leaf nor the root, represents a community in G, induced by the leaves of the subtree rooted at n.
- Each community in G is represented by a node of T.
- The root of T represents the whole graph.
- The generators of a community represented by a node n of T are:
  - The vertices of G corresponding to leaves being direct children of n
  - The edges of G between leaves of different subtrees rooted at n.

3.3 The Algorithm

Up to this point we used descriptions of communities without proving their existence. Now we are going to present an algorithm *CommunityTree* (see fig. 2) for the construction of one for an arbitrary undirected graph G. The basic type of the tree representation is a **node**. Each **node** n can either have several children or it describes a vertex of G. In addition it allows us to store the edge-connectivity of the subgraph $G[n]$ and the number of vertices and edges which are mapped to n by *vcomm* or *ecomm*. The latter one speeds up the subsequent reduction of resulting description of communities. Furthermore we need two global arrays:

1. **node**[] *vcomm* indexed by vertices of G, representing the map *vcomm*, and
2. **node**[] *ecomm* indexed by edges of G, representing the map *ecomm*.

Remark 2. As mentioned earlier we can safely ignore self-loops and replace all parallel edges between two vertices u and v with one edge of weight $w(u, v)$. Hence we can assume that the graph given for the algorithm does not contain self-loops and multiple edges.

Theorem 6. *If G is a non-empty graph without self-loops and multiple edges, and $T := CommunityTree(G)$, then $(T, \varphi, vcomm, ecomm)$ is a description of the communities of G.*

To obtain a reduced tree we have to traverse the tree returned by the algorithm $CommunityTree$ and check every internal node, whether the conditions are satisfied, i.e. if it is referenced and has two or more children.

Theorem 7. *Let G be a graph and n its number of vertices and m its number of edges. If the minimal cut of a graph G can be calculated in $O(f(n, m))$, such that f is monotone increasing in n and m, then the reduced description of communities of G can be calculated in runtime $O(nf(n, m))$.*

Using the $O(nm + n^2 m \log n)$-algorithm of Nagamochi and Ibaraki ([NI92]) for the calculation of minimal cuts, we obtain a runtime of $O(n^2 m + n^3 m \log n)$.

```
1   node CommunityTree(graph G)
2   begin
3      if G = ({v}, ∅) then
4         root = new node(v)
5         vcomm[v] = root
6         vcomm[v].conn = 0
7         vcomm[v].references = 1
8         return root
9      endif
10     if G = ({u, v}, {e}) then
11        root = new node(new node(u), new node(v))
12        vcomm[u] = root
13        vcomm[v] = root
14        ecomm[e] = root
15        root.references = 3
16        root.conn = w(u, v)
17        return root
18     endif
19     (G1, G2) = MinimalCut(G)
20     child1 = CommunityTree(G1)
21     child2 = CommunityTree(G2)
22     root = new node(child1, child2)
23     root.conn = conn(G)
24     root.references = 0
25     for v ∈ V(G) do
26        if vcomm[v].conn ≤ conn(G) then
27           vcomm[v].references − −
28           vcomm[v] = root
29           root.references + +
30        endif
31     next
32     for e ∈ E(G) do
33        if (ecomm[e].conn ≤ conn(G)) or
                    (ecomm[e] not defined) then
34           ecomm[e].references − −
35           ecomm[e] = root
36           root.references + +
37        endif
38     next
39     return root
40  end
```

Fig. 2. The algorithm *CommunityTree*.

4 *k*-Communities

Sometimes the notion of communities in our sense might be to restrictive. Therefore we are going to relax the conditions to produce larger subgraphs. But, as we

are going to see, they are directly related to communities and can be determined from the reduced description of communities of a graph.

Definition 4. *Let k be a positive real number, G a graph and H a subgraph of G. The* k-community $Comm_G^k(H)$ of H in G *is the union of all subgraphs U of G containing H, with edge-connectivity at least k.*

Remark 3. For and edge or vertex x the k-community $Comm_G^k(x)$ is precisely the k-component containing x, as defined by Matula in [Mat72].

Remark 4. Let G be a graph and H a subgraph of G.

1. $Comm_G^k(H)$ is a complete subgraph of G, because the addition of edges increases the edge-connectivity.
2. $conn(Comm_G^k(H)) \geq k$ or $Comm_G^k(H) = \emptyset$.
3. $Comm_G^k(H) = Comm_G(H)$ for $k = str_G(H)$.
4. $Comm_G^k(H) = \emptyset$ for $k > str_G(H)$.
5. $Comm_G^k(H) = G$ for $k \leq conn(G)$.
6. $Comm_G^k(H) \subseteq Comm_G^l(H)$ if $l \leq k$.

Theorem 8. *Let $D = (T, \varphi, vcomm, ecomm)$ be a reduced description of communities of a graph G and H a subgraph of G and $0 < k \leq str_G(H)$. If n_H is the node of T with $Comm_G(H) = G[n_H]$, then $Comm_G^k(H) = G[m]$, where m is the root of the largest subtree containing n with $conn(G[m]) \geq k$.*

5 Efficient Methods

The two methods for the treatment of dangling links (nodes of degreee one) and low degree vertices prevent a large number of trivial minimal cuts, which would seperate only one vertex from the rest of the graph and hence are especially efficient for excerpts of the web graph.

5.1 The Dangling Link Handling

The *Dangling Link Handling* (DLH) allows us to treat dangling links, i.e. vertices which are connected to only one neighbor, in a very efficient way, reducing the number of minimal cuts needed during the construction of a reduced description of communities.

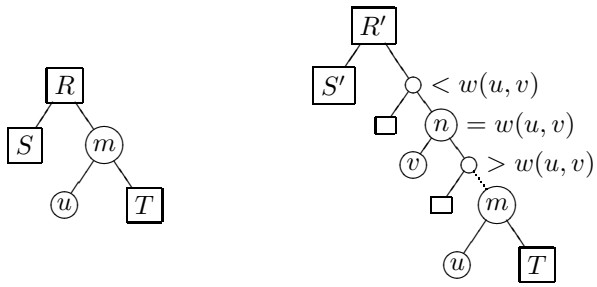
Theorem 9. *Let $G = (V, E)$ be a graph and $v \in V$ a vertex which is connected to exactly one other vertex $u \in V$. If $G' = G \setminus \{v\}$, i.e. G' is obtained from G by removing v and the adjacent edges, and if $w(u, v)$ is the sum of weights of all edges between u and v, then we have for $u \neq w \in V$:*

$$1.\ Comm_G(w) = \begin{cases} Comm_{G'}(w) & \text{if } str_{G'}(w) > w(u,v) \\ & \text{or } u \notin Comm_{G'}(w) \\ G[V(Comm_{G'}(w)) \cup \{v\}] & \text{otherwise} \end{cases}$$

$$2.\ Comm_G(u) = \begin{cases} Comm_{G'}(u) & \text{if } str_{G'}(u) > w(u,v) \\ G\left[V\left(Comm_{G'}(u)\right) \cup \{v\}\right] & \text{if } str_{G'}(u) = w(u,v) \\ G[\{u,v\}] & \text{if } str_{G'}(u) < w(u,v) \end{cases}$$

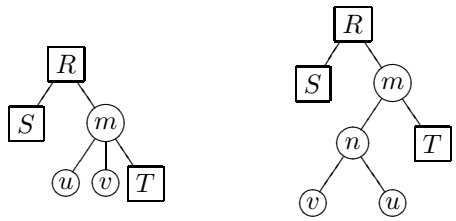
$$3.\ Comm_G(v) = \begin{cases} G[\{u,v\}] & \text{if } str_{G'}(u) < w(u,v) \\ G\left[V\left(Comm_{G'}^{w(u,v)}(u)\right) \cup \{v\}\right] & \text{otherwise} \end{cases}$$

If we assume that we have a reduced description of communities of G' of the form shown on the left above, then the following trees describe the reduced description of G. Here m is the node of the tree describing the community of u, i.e. it is the parent of the leaf corresponding to u.



If $str_{G'}(u) > w(u,v)$, then the tree of the reduced description of communities of G is of the right type above. Let m' be the node representing $Comm_{G'}^{w(u,v)}(u)$. We either have $m' = n$, if its edge-connectivity is precisely $w(u,v)$, or n is a new node added between m' and its parent. If m' is the root of the tree the new node becomes the new root.

If $str_{G'}(u) = w(u,v)$ the adapted tree has the form on the left below. In the last case, $str_{G'}(u) < w(u,v)$, the new tree is of the form on the right, where a new node n has to be added. In the latter situation one has to check, whether the node m still represents a community of an edge or vertex. If this is not the case, it has to be removed to obtain a reduced description of communities of G.



5.2 The Vertex Degree Handling

The *Vertex Degree Handling* (VDH) allows us to calculate a minimal cut of the given graph and iteratively remove all vertices from the two subgraphs, whose degree in the already reduced graph is lower than or equal to the connectivity of the whole graph. This method is very valuable, since it prohibits many minimal cuts which would only separate one vertex of low degree from the remaining graph. These cuts are very expensive and do not contain much information. But the following theorem allows us to detect these vertices - and several additional ones - in advance, to remove them and to proceed with the reduced subgraphs.

Theorem 10. *Let G be a graph, (G^1, G^2) a minimal cut of G and $(v_1, \ldots, v_n)$ for $n \geq 1$ a sequence of pairwise different vertices of G^l for one $l \in \{1, 2\}$. We define $G_0 := G^l$ and $G_i := G[V(G^l) \setminus \{v_1, \ldots, v_i\}]$ for $1 \leq i \leq n$. Furthermore let $w_{G_i}(v)$ be the sum of weights of all edges in G_i adjacent to v.*

If $w_{G_{i-1}}(v_i) \leq k := conn(G)$ is satisfied for all $1 \leq i \leq n$, then for each subgraph $H \subset G_n$ the following holds:

- $Comm_G(H) = G \quad \Leftrightarrow \quad str_{G_n}(H) \leq k.$
- $Comm_G(H) = Comm_{G_n}(H) \quad \Leftrightarrow \quad str_{G_n}(H) > k.$
- $Comm_G(v_i) = G$ *for* $1 \leq i \leq n.$

5.3 Experiments

To judge the runtime of the algorithm and the effectiveness of the efficient methods, we conducted some simple experiments[2]. Due to space restrictions only the result of a "real-life"-experiment, namely the structuring of the web graph of the domain *theoinf.tu-ilmenau.de*, is given here. It consists of 6984 documents (including directly referenced, exterior ones) and 16499 links. The results are presented in Tab. 1.

Similar results were observed for several types of random graphs. Depending on the exact model and the parameters, the decrease of runtime using VDH varied between factor 30 and 1000. The DLH is not as successfull, and may even cause an increase in runtime, since the time for the detection of dangling links may exceed the gain.

6 Applications

6.1 Structuring the WWW

The motivation for this paper was the identification of communities in the WWW. Intuitively a community of the WWW is a set of web pages covering the same or closely related topics. Since most of the hyperlinks are made by purpose, they usually indicate that the target covers topics related to the

[2] The algorithm was implemented using LEDA 4.3 and the experiments were run on a Linux system (kernel 2.4.18) with 1.5 GHz Pentium processor and 512 MB memory.

Table 1. Measurements of the web graph of *theoinf.tu-ilmenau.de*.

Web graph of *theoinf.tu-ilmenau.de*							
Nodes	**Edges**	**Add.**	**Total(s)**	**Cutting(s)**	**Cuts**	**DLH**	**VDH**
6984	16499	None	198336	198195	6890	-	-
		DLH	2360.35	2311.48	4040	2859	-
		VDH	54.34	52.98	118	-	6960
		All	44.55	43.34	111	2795	4180

topics covered by the source. Hence two pages can be interpreted as having similar content if there exist many edge-disjoint connections between them, and the edge-connectivity of a set of pages is an indicator, to which degree these pages are closely related. In this approach the direction of a hyperlink is neglected. A link is interpreted as a "vote" for the fact, that the two connected pages cover related topics.

At this point our notion of communities comes into play. It allows us to construct a natural clustering from the graph, describing sets of pages of highest possible connectivity. By using k-communities we can weaken the similarity structure and be less restrictive. At the same time we obtain a hierarchy of communities and therefore of vertices and edges. We have *specialized communities*, which do not contain more than one other community, and *generalizing communities*, which contain two or more smaller communities. The intuition behind this hierarchy is that specialists cover a very specific topic or aspect of a topic and generalizing communities cover more general themes or a union of several topics.

If one does not want to neglect the additional information of the direction of links, a different approach may be used. It is based on the basic idea of the HITS algorithm by Kleinberg, introduced in [Kle98]. Instead of counting each link between two pages as a "vote" for the similarity of the two pages, we take a look at *cocitations*. This means that we construct a graph whose vertices are the pages of the WWW or a part of it, and we have an edge $\{u, v\}$ between two of them, if there exists a page w which links to both of them. The weight of the edge (or its multiplicity) is the number of cociting pages. Then the communities of this graph are interpreted as communities of the WWW. As above we obtain specialized and generalizing communities.

A dual approach to cocitations would be common citations, i.e. the weight of an edge $\{u, v\}$ is the number of pages w which are cited by both, u and v. Combinations of these two values are possible, too.

An intuitive problem of our approach regarding communities in the WWW arises, because each vertex is assigned to only one community. But in reality one document (homepage, server) may belong to different communities. A direct application of our approach may cover this partially by generalizing communities, joining two or more topics. But with very different topics this can fail. Instead

another approach may be more suitable. Usually a link of a document to another one is made because of a specific reason. Hence a page covering several topics has several links, one for each topic. Therefore it would be interesting to construct the *communities of edges* instead of nodes. This can easily achieved by replacing the original graph with its line graph, i.e. by a graph whose vertices correspond to the original edges and whose edges describe "connections" between original edges. A community in this line graph is generated by a set of edges in the original graph and the WWW community would correspond to the induced subgraph. As a consequence a vertex of the original graph may belong to several communities (one for each community of one of its adjacent edges).

6.2 Clustering

The tree of communities allows us to partition the vertices of G. Generally we just have to choose a set of disjoint subtrees (i.e. nodes of the tree of communities, such that no node is contained in the subtree rooted at another node), representing the partitions. This may leave several vertices unassigned (the children of nodes closer to the root), which can be treated as *singletons*.

The way in which the basic partitions are chosen can vary widely. On one hand we can choose all communities not containing a sub-community, resulting in a very fine partition. Or we can choose all k-communities containing no k-sub-community.

Another way to vary the clustering is the adaption of the underlying graph before calculating the communities and selecting the partition. Examples were given in the previous section.

6.3 Graph-Drawing and Browsing

The natural clustering of graphs given by communities can also be applied to draw them. In the resulting picture the vertices are grouped by the hierarchy of communities, i.e. members of a certain community are usually drawn closer to each other than member of different communities.

In addition to the drawing, an interactive interface for the browsing in a graph may be implemented. At each time the shown graph consists of vertices representing either communities or real vertices of the graph. The edges are induced by the underlying graph. Each vertex representing a community may be expanded. The expansion replaces the vertex by a subgraph whose vertices are its sub-communities and members. A collapse of an arbitrary vertex, causes its super-community to remove each of its sub-communities and members to be removed from the drawing and to be represented by a single vertex. Such an interface would allow the user to browse the structure quite efficiently.

References

[AB99] R. Albert and A. Barabàsi. Emergence of scaling in random networks. *Science*, 286:509–512, October 1999.

[ABJ00] R. Albert, A. Barabàsi, and H. Jeong. Scale-free characteristics of random network: The topology of the world wide web. *Physica A*, 281:69–77, 2000.

[AJB99] R. Albert, H. Jeong, and A. Barabási. Diameter of the world-wide web. *Nature*, 401:130–131, September 1999.

[BH98] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, AU, 1998.

[Bot93] Rodrigo A. Botafogo. Cluster analysis for hypertext systems. In *Proc. of the 16-th Annual ACM SIGIR Conference of Res. and Dev. in Info. Retrieval*, pages 116–125, 1993.

[BRRT01] Alan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *World Wide Web*, pages 415–429, 2001.

[BS91] Rodrigo A. Botafogo and Ben Shneiderman. Identifying aggregates in hypertext structures. In *UK Conference on Hypertext*, pages 63–74, 1991.

[Bri02] Michael Brinkmeier. Communities in Graphs. Technical Report. Downloadable at *http://citeseer.nj.nec.com/brinkmeier02communities.html*.

[CDK+99] Soumen Chakrabarti, Byron E. Dom, S. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, Andrew Tomkins, David Gibson, and Jon Kleinberg. Mining the Web's link structure. *Computer*, 32(8):60–67, 1999.

[FCE95] Qingwen Feng, Robert F. Cohen, and Peter Eades. How to draw a planar clustered graph. In *COCOON'95*, volume 959 of *Lecture Notes in Computer Science*, pages 21–31. Springer, 1995.

[Fen97] Qingwen Feng. *Algorithms for Drawing Clustered Graphs*. PhD thesis, Department of Computer Science and Software Engineering, University of Newcastle, 1997.

[FLG00] Gary Flake, Steve Lawrence, and C. Lee Giles. Efficient identification of web communities. In *Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–160, Boston, MA, August 20–23 2000.

[GKR98] David Gibson, Jon M. Kleinberg, and Prabhakar Raghavan. Inferring web communities from link topology. In *UK Conference on Hypertext*, pages 225–234, 1998.

[HS00] Erez Hartuv and Ron Shamir. A clustering algorithm based on graph connectivity. *Information Processing Letters*, 76(4–6):175–181, 2000.

[HSL+99] Erez Hartuv, Armin Schmitt, Jorg Lange, Sebastian Meier-Ewert, Hans Lehrachs, and Ron Shamir. An algorithm for clustering cDNAs for gene expression analysis. In *RECOMB*, pages 188–197, 1999.

[Kle98] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[KRRT99] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1481–1493, 1999.

[Lit] Bruce Litow. A review of world wide web searching techniques, focusing on hits and related algorithms that utilise the link topology of the world wide web to provide the basis for a structure based search technology.

[Mat69] David W. Matula. The cohesive strength of graphs. In G. Chartrand and S.F. Kapoor, editors, *The Many Facets of Graph Theory*, volume 110 of *Lecture Notes in Mathematics*, pages 215–221. Springer, 1969.

[Mat72] David W. Matula. k-components, clusters, and slicings in graphs. *SIAM J. Appl. Math.*, 22(3):459–480, 1972.

[NI92] Hiroshi Nagamochi and Toshihide Ibaraki. Computing edge-connectivity in multigraphs and capacitated graphs. *SIAM J. Disc. Math.*, 5(1):54–66, 1992.

[PBMW98] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

Randomly Distributed Tasks in Bounded Time

Alain Bui[1⋆], Marc Bui[2], and Devan Sohier[1,2⋆⋆]

[1] Département de Mathématiques et Informatique
Université de Reims Champagne Ardennes,
BP1039 F-51687 Reims cedex, france,
alain.bui@univ-reims.fr, devan.sohier@univ-reims.fr
[2] LRIA - EPHE rue G. Lussac,
F-75005 Paris, France
marc.bui@univ-paris8.fr

Abstract. Random walk is the standard modeling for a randomly circulating token in a network, in distributed computing. In particular, this attractive technique can be used to achieve a global computation using a subset of computers over a network. In this paper, we present two original methods to automatically compute the processing time through hitting times. We also propose a solution to determine the number of resources necessary to achieve a global computation.

1 Introduction

Random walks are often used in the design and the analysis of distributed algorithms. For example, a leader election can be achieved in anonymous networks using random walks. Each process participating to the election sends a token to one of its neighbors chosen at random. When two tokens or more meet at a processor then they merge to one. It is easy to prove that within a finite time, eventually all tokens will merge to one.

Random walks are useful to get a global information or for structuring anonymous networks, and they are naturally well-adapted to ad-hoc mobile networks. Indeed, a good solution to cover the network efficiently without any knowledge of its structure, and to provide easy solutions that do not depend on processors identifier can be achieved with random walks. For example, a spanning tree construction can be achieved as follows. Each processor, when receiving a token for the first time, sets its *parent* variable to the sender, and sends the token each time it receives to one of its neighbour chosen uniformly at random. The result of this procedure gives a *random* spanning tree, chosen uniformly at random among all possible spanning trees.

Original solutions using random walks have been designed for many control problems in distributed computing *e.g* [15] for self-stabilizing mutual exclusion, [5] for mobile agent in wireless networks. The most efficient edge-disjoint paths

⋆ corresponding author
⋆⋆ partially supported by the ACI GRID

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 36–47, 2003.

algorithm on unknown topology currently known is based on random walks [3]. An algorithm itself can be seen as a random walk, as shown in [9].

Recently, a new way to design computation using available computational components of local networks to wide area networks as Internet have been extensively developed such as GRID computing paradigm [13, 14].

In [12], a fully distributed peer to peer middleware is designed to compute a global set of independent tasks. The set of computers in the network using this peer to peer middleware is structured as a ring. To gather sites local computation results, a deterministic token circulation approach is used.

A random approach will provide an efficient solution to this problem. The token is circulated by all the sites in a random walk policy. Simulation tests that we have realized show that each site is up to 20 % more often visited in the random policy approach. Network subsets as Internet sub-sets are also subject to topological changes. Unlike the deterministic approach, the random token circulation strategy is well designed to tolerate such changes. A new site can join the computation and eventually this new site will be reached by the token. Such solution for the deterministic approach is more intricate and expensive.

In order to evaluate the complexity of algorithms using random walks, mathematical tools [1, 17] such that hitting times, cover time *etc.* are used. Given a graph, there is no effective way to compute the cover time which is defined by the maximum of the expected number of steps it take to a random walk to visit all vertices of the graph. [2, 11, 10, 17] give results on upper or lower bounds.

In this paper we use *the cyclic cover time* notion and we introduce the notion of *the total hitting time measure* that we are able to compute exact value. These notions being function of *hitting times*, we propose two new methods to automatically compute them. The first method provide an automatic way to compute resistance on graph. By use of the well-known relationship between electrical networks and random walks, total hitting time and cyclic cover time (for symmetrical graph only) are deduced. The second method uses results from Markov chain theory. Given the adjacency matrix of a graph, total hitting time and cyclic cover time can be computed automatically without any restriction. We obtain a relation between the computation time and the number of resources needed to achieve a global task. We have applied our results to complete graphs.

The paper is organized as follow: in next section we describe the distributed systems, model and definitions we consider in the paper. We also state the different notions of time computation we used. In section 3, we give results on how compute electrical resistance to be able to capture time computation. In section 4, we present the second approach based on matrix computation. Conclusion is given in section 5.

2 Preliminaries

Distributed systems A distributed system can be viewed as an undirected connected graph $G = (V, E)$, where V is a set of processors with $|V| = n$ and E is the set of bidirectional communication link with $|E| = m$. (We use the terms

"nodes", "vertex" and processor" interchangeably). We consider *asynchronous* networks. A communication link (i, j) exists if and only if i and j are neighbors. Every processor i can distinguish all its links and i maintains its sets of neighbors denoted as N_i. The degree of i is the number of neighbors of i, i.e. $|N_i|$, denoted as $deg(i)$.

Random walks A random walk is a sequence of vertices visited by a token that starts at i and visited other vertices according to the following transition rules: if the token is at i at time t then at time $t+1$ it will be at one of the neighbors of i, this neighbor having been chosen uniformly at random among all of them.

More formally, a random walk is a finite homogeneous Markov Chain with state set V and with transition matrix probability $P = (p_{ij})_{(i,j)\in V\times V}$ given by

$$p_{ij} = \begin{cases} \frac{1}{\deg(i)} & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \end{cases}$$

where $\deg(i)$ is the degree of node i.

Let P^t the t^{th} power of P, whose entries are $p_t(i,j),\ (i,j) \in V \times V$.

Since G is connected, if it is not bipartite, the Markov Chain has only one acyclic ergodic class of states, then $\lim_{t\to\infty} P^t$ exists and is a matrix Q with identical rows $\pi = (\pi_i,\ i \in V)$, *i.e.* $\forall (i,j) \in V \times V, \lim_{t\to\infty} p_t(i,j) = \pi_i$. π is the stationnary distribution and can be computed such that $\pi = \pi.P$.

Note that, in the particular case of random walks, the stationnary distribution satisfies

$$\pi_i = \frac{\deg(i)}{2|E|} \tag{1}$$

Some characteristic values are useful in the context of distributed computing.

The mean time to reach vertex j (state j), starting from the vertex i (state i) which may be regarded as the conditional expectation of the random number of transitions before entering j for the first time when starting from i, is called *hitting time* and denoted h_{ij}. In particular, we have $h_{ii} = 1/\pi_i$. We often use the quantity $\max\{h_{ij}/j \in V\}$, which is an upper bound for a random walk starting at i to hit a fixed, but unknown vertex, for example, when the average time to look for an information owned by a unknown vertex is required.

$h_{ij} + h_{ji}$ called the commute time, is the expected number of steps for a random walks starting at vertex i to reach vertex j for the first time and reach i again. It can be viewed as the average time to fetch back to i an information owned by the vertex j.

The expected time for a random walk starting at i to visit all the vertices of the graph is called the *cover time* C_i. Let $C = \max\{C_i/i \in V\}$. C_i will be the average time needed by i to build a spanning tree thanks to the algorithm described above. C will be an upper bound of the average time for an unknown vertex to build a spanning tree thanks to the algorithm described above.

We added the two following notions to estimate global time computation. Both notions are based on hitting times.

Definition 1. *The* cyclic cover time *[7] is defined to be the average time to visit all the vertices in the best deterministic arrangement:*

$$CCT = \min \left\{ \sum_{j=1}^{n-1} h_{\sigma(j)\sigma(j+1)} / \sigma \in S_n \right\}$$

Cyclic cover time is closed to the cover time notion, since a random walk has necessarily visited each vertex when it has visited every vertex in a given arrangement.

Definition 2. *The* total hitting time *measure T is defined by*

$$\sum_{(i,j)\in V^2} h_{ij}$$

This time can be viewed as the worst-case computation time.

Random walks and resistive networks Useful quantities such that cover time are hard to compute. On arbitrary graphs, only bounds on of these are available.

We know ([17, 11, 10]) that in a connected graph G on n nodes:

$$\forall (i,j) \in V \times V,\ h_{ij} \leq \frac{4}{27}n^3 - \frac{1}{9}n^2 + \frac{2}{3}n + O(1)$$

and

$$(1 - o(1))n\log(n) \leq C \leq (1 + o(1))\frac{4}{27}n^3$$

Correspondence between electrical networks is also known [6, 8]. Results have showed a tight link between cover time, hitting time and resistances in electric networks. This provides an efficient and easy way to compute the complexity of many distributed algorithms.

It has been shown ([6]) that:

Lemma 1.

$$h_{ij} + h_{ji} = 2mR_{ij} \tag{2}$$

where i and j denote two distinct vertices, m, the number of vertices, and R_{ij} the effective resistance between nodes i and j, if we replace each edge in the graph by a 1Ω resistor.

From this equation, one can deduce that:

Lemma 2.

$$mR < C < mR\log n \tag{3}$$

where R denotes the maximal effective resistance between two nodes of the network.

3 Computing Automatically Electrical Resistance

Thanks to the results about the link between random walks and resistive networks, we are able to compute some hitting times. Let consider a ring of size n and let i and j be two vertices on this ring. The electrical circuit built from this ring and an equivalent circuit is represented on fig. 1.

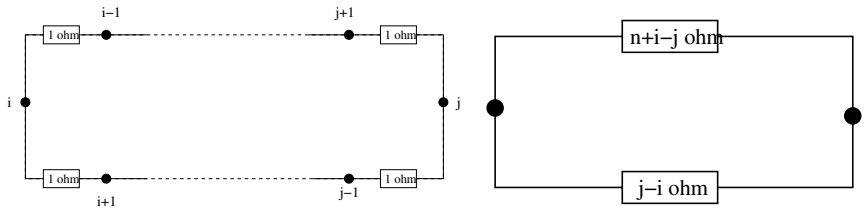


Fig. 1. the electrical circuit built from a ring and an equivalent circuit

Then, the two resistors being wired in parallel:

$$R_{ij} = \frac{1}{\frac{1}{j-i} + \frac{1}{n+i-j}} = \frac{(n+i-j)(j-i)}{n}$$

By equation (2) and because $h_{ij} = h_{ji}$ (the graph being symetrical), and $m = n$, we have $h_{ij} = (n+i-j)(j-i)$

Theorem 1 (Millman).

$$\forall k \in V, \sum_{l \in N_k} \frac{v_l - v_k}{r_{ij}} = 0$$

r_{kl} being the resistance of the resistor between k and l, v_i the potential at node i.

In our case, $r_{kl} = 1\Omega$. Then, assuming that, for certain i and j, $V_i = 1$ and $V_j = 0$, we can solve this system and know all the potentials. We can then deduce the current going out of i (since $U = RI$) and compute the effective resistance between i and j. We have:

$$V_i = 1,\ V_j = 0 \ \text{ and } \forall k \in V \backslash \{i, j\}, \sum_{l \in N_k} (V_l - V_k) = 0$$

Example Let apply Millman theorem to the graph on fig. 2, to compute R_{17}.

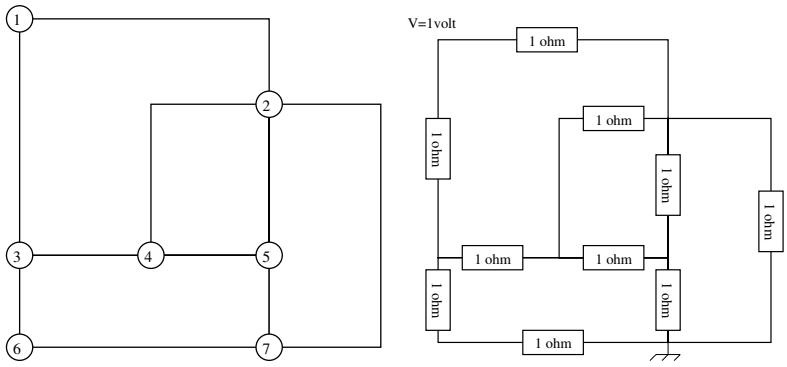


Fig. 2. Example

Then $V_1 = 1$, $V_7 = 0$.

$$\begin{aligned} V_1 - V_2 + V_4 - V_2 + V_5 - V_2 + V_7 - V_2 &= 0 \\ V_1 - V_3 + V_4 - V_3 + V_6 - V_3 &= 0 \\ V_2 - V_4 + V_3 - V_4 + V_5 - V_4 &= 0 \\ V_2 - V_5 + V_4 - V_5 + V_7 - V_5 &= 0 \\ V_3 - V_6 + V_7 - V_6 &= 0 \end{aligned} \tag{4}$$

From (4), we obtain $V_6 = \frac{2}{7}$, $V_4 = \frac{3}{7}$, $V_5 = \frac{2}{7}$, $V_3 = \frac{4}{7}$ and $V_2 = \frac{3}{7}$. The current going out of 1 is $\frac{V_2}{1\Omega} + \frac{V_3}{1\Omega} = 1$, and $R_{17} = 1$. Thus, $h_{17} + h_{71} = 10 \times 1 = 10$.

The algorithm Effective resistance can be computed automatically with Millman theorem. Consider a graph G. Let R_{ij} the effective resistance between two nodes i and j of G. Consider the matrix M defined by

$$\begin{cases} M_{ii} = M_{jj} = 1 & \\ M_{ik} = 0 & \text{if } i \neq k \\ M_{jk} = 0 & \text{if } j \neq k \\ M_{kk} = \deg(k) & \\ M_{hk} = -1 & \text{if } h \in N_k,\ h \neq i,\ h \neq j \\ M_{hk} = 0 & \text{if } h \notin N_k,\ h \neq i,\ h \neq j \end{cases}$$

The potential at each node of the circuit is given by:

$$V = M^{-1}.S$$

where S is a vector with all entries to 0 except for the ith line with entry to 1, that is, $S = {}^T[0 \dots 0\ 1\ 0 \dots 0]$

Then the current going out of i is $\sum_{k\in N_i} i_{ik}$ where $i_{kl} = (V[k] - V[l])/r_{kl}$ with $r_{kl} = 1\Omega$. But $v[i] - v[j] = 1$ and by definition of the effective resistance, we have

$$R_{ij} = \frac{V[i] - V[j]}{\sum_{k\in N_i} i_{ik}}$$

This solution can be implemented thanks to the algorithm below:

```
PROCÉDURE R(i,j) = Millman (G: graph, i: node in G, j: node in G) Is
  S(i)←1
  For all node k in G except for i Do
    S(k)←0
  For all node k and h in G Do
    If h=i Then
      If k=i Then
        M(h,k)←1
      Else
        M(h,k)←0
    Else
      If h=j Then
        If k=j Then
          M(h,k)←1
        Else
          M(h,k)←0
      Else If h=k Then
        M(h,k)←the number of neighbors of k
      Else If h is a neighbor of k Then
        M(h,k)←-1
      Else
        M(h,k)←0
  S←M^(-1)S
  intensite←0
  For all neighbor k of i Do
    intensite← intensite+1-V(k)
  return 1/intensite
End Millman
```

Application We intend to compute $R_{12} = R_{ij}$ for all i and j in V in a complete graph. Since each site has $n-1$ neighbors, the matrix M is:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ -1 & -1 & n-1 & -1 & \dots & -1 \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ -1 & \dots & -1 & n-1 & \dots & -1 \\ \vdots & & & \ddots & \ddots & \vdots \\ -1 & \dots & \dots & \dots & -1 & n-1 \end{bmatrix} \text{ and } M^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{3}{2n} & \frac{1}{2n} & \cdots & \frac{1}{2n} \\ \vdots & \vdots & \ddots & \ddots & \dots & \vdots \\ \frac{1}{2} & \frac{1}{2} & \dots & \dots & \frac{1}{2n} & \frac{3}{2n} \end{bmatrix}$$

So, S being: ${}^T[1\ 0 \dots 0]$, V is: $\left[1\ 0\ \frac{1}{2} \dots \frac{1}{2}\right]$. Thus, the current going out of i is $\sum_{k\in N(i)} i_{ik} = \sum_{k\in N(i)} V_i - V_k = n - \sum_{k=2}^{n} V_k = n - 1 - \sum_{k=2}^{n} \frac{1}{2} = \frac{n}{2}$. Then, $R = \frac{2}{n}$. We have proved:

Proposition 1. *The resistance between two vertices in a complete graph is* $\frac{2}{n}$

Corollary 1. *For the complete graph,* $h_{ij} = n - 1$ *if* $i \neq j$, $h_{ii} = n$, *and the total hitting time is* $n^3 - n^2 + n$.

Proof. In a complete graph on n vertices, $m = \frac{n(n-1)}{2}$, $h_{ij} = h_{ji}$, then

$$h_{ij} = mR_{ij} = \frac{n(n-1)}{2} \times \frac{2}{n} = n - 1$$

and

$$h_{ii} = 1 + \sum_{j\in N(i)} \frac{h_{ij}}{n-1} = 1 + n - 1 = n.$$

Thus,

$$\begin{aligned} \sum_{(i,j)\in V^2} h_{ij} &= n(n-1) \times (n-1) + n \times n \\ &= n^3 - 2n^2 + n + n^2 \\ &= n^3 - n^2 + n \end{aligned} \tag{5}$$

Corollary 2. *For the complete graph,* $n - 1 \leq C \leq (n-1)\log n$, *where* C *is the cover time.*

A classical result in electrical theory (the Rayleigh's *"short/cut"* principle: [6]) tells that removing an edge from a resistive network can only strengthen resistances in this network, and that adding an edge weakens them. Then, the resistance between two nodes in a graph of n nodes is always greater than in a complete graph and less than in a path of $n-1$ edges from i to j (the graph is connected).

The resistance between two vertices in a path of $n-1$ sites is less than $n-1$, due to their being wired in serie. Thus:

Proposition 2. *For any arbitrary graph,*

$$\begin{aligned} 4\frac{m}{n} \leq h_{ij} + h_{ji} \leq 2mn \\ 2\frac{m}{n} \leq C \leq 2m(n-1)\log n \end{aligned} \tag{6}$$

Those bounds are not very tight. However, the one with the hitting times is the best regarding the few information we have about the architecture of the considered networks (a complete graph matches the lower bounds, wheras a path of $n-1$ edges between the considered vertices matches the upper bounds).

Note that, moreover, we only bound mean values: actual results can be far from them.

4 Computing Automatically Hitting Times

4.1 Generic Procedure

Previous method based on electrical resistance establishes a relation between effective resistance and commute time. It is sufficient to determine total hitting time measure for any graph and cyclic cover time for any symmetrical graph (such that $h_{ij} = h_{ji}$). In this section, we use another systematic method in order to obtain total hitting times and cyclic cover time for any graph.

We use the results from Markov chain theory [16, 4]. From any graph adjacency matrix, we obtain the mean first passage matrix, that is to say the hitting time matrix $H = (h_{ij})_{(i,j)\in V\times V}$ by the following procedure:

From the adjacency matrix, we form the random walk transition probability matrix P. Then we compute $Q = \lim_{n\to\infty} P^n$, the matrix with each row π (*cf.* section 2)

Proposition 3. *The hitting time matrix H is*

$$H = (h_{ij})_{i,j\in V\times V} = (I - Z + EZ_{dg})D$$

where

I is the identity matrix,

$Z = [I - P + Q]^{-1}$ E is the matrix with all entries 1*, Z_{dg} is the matrix resulting from Z by setting off-diagonal entries equal to* 0*,*

D is the diagonal matrix with j-th entry $d_{jj} = \frac{1}{\pi_j}$.

This algorithm provides a systematic method which can be applied for any graph. But in particular, a general expression of the hitting times can be achieved for some graph. In the following, we applied it to the complete graph.

4.2 Application to the Complete Graph

Consider a complete graph on n vertices, The transition probability matrix P for such a graph is

$$P = \begin{bmatrix} 0 & \frac{1}{n-1} & \cdots & \frac{1}{n-1} \\ \frac{1}{n-1} & 0 & \cdots & \frac{1}{n-1} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{1}{n-1} & \cdots & \frac{1}{n-1} & 0 \end{bmatrix}$$

A first remark, its absolute stationary probability is

$$\pi = \left(\frac{1}{n}, ..., \frac{1}{n}\right).$$

Proposition 4. *The hitting time matrix M is the matrix with all diagonal entries n and all nondiagonal entries $n - 1$*

Proof. We apply the previous algorithm.

The matrix $I - P + Q$ is

$$\begin{bmatrix} \frac{n+1}{n} & \frac{1}{n(1-n)} & \cdots & \cdots & \cdots & \frac{1}{n(1-n)} \\ \frac{1}{n(1-n)} & \frac{n+1}{n} & \frac{1}{n(1-n)} & \cdots & \cdots & \frac{1}{n(1-n)} \\ \vdots & \ddots & \ddots & \ddots & \cdots & \vdots \\ \frac{1}{n(1-n)} & \cdots & \frac{1}{n(1-n)} & \frac{n+1}{n} & \frac{1}{n(1-n)} & \vdots \\ \vdots & \cdots & \cdots & \ddots & \ddots & \vdots \\ \frac{1}{n(1-n)} & \cdots & \cdots & \cdots & \frac{1}{n(1-n)} & \frac{n+1}{n} \end{bmatrix}$$

So, calculus gives that:

$$Z = \begin{bmatrix} \frac{1+n^3}{n^2(n+1)} & \frac{1}{n^2} & \cdots & \cdots & \cdots & \frac{1}{n^2} \\ \frac{1}{n^2} & \frac{1+n^3}{n^2(n+1)} & \frac{1}{n^2} & & & \frac{1}{n^2} \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ \frac{1}{n^2} & \cdots & \frac{1}{n^2} & \frac{1+n^3}{n^2(n+1)} & \frac{1}{n^2} & \vdots \\ \vdots & & & \ddots & \ddots & \ddots \\ \frac{1}{n^2} & \cdots & \cdots & \cdots & \frac{1}{n^2} & \frac{1+n^3}{n^2(n+1)} \end{bmatrix}$$

Then, EZ_{dg} being $\frac{1+n^3}{n^2(n+1)}.E$ and D being $n.I$, short calculus give that $h_{ij} = n-1$ if $i \neq j$ and $h_{ii} = \frac{1}{\pi_i} = n$, by equation (1).

4.3 Computation Time

From the results of section 3 or section 4 for the complete graph, we have,

Corollary 3. *The total hitting times measure is*

$$\sum_{(i,j)\in V^2} h_{ij} = n^3 - n^2 + n.$$

Since for all i and j in V with $i \neq j$, $h_{ij} = n - 1$, the cyclic cover time is:

$$\begin{aligned} CCT &= \min\left\{\sum_{i=1}^{n-1} h_{\sigma(i)\sigma(i+1)} / \sigma \in S_n\right\} \\ &= \min\left\{\sum_{i=1}^{n-1} (n-1) / \sigma \in S_n\right\} \\ &= (n-1)^2 \end{aligned} \tag{7}$$

Corollary 4.

$$CCT = (n-1)^2$$

For the complete graph, given a total computation time, the number of necessary resources can be obtained.

Proposition 5. *Given the total hitting time T the size r of the network must be* $\left\lceil T^{\frac{1}{3}} \right\rceil + 1$.

Proof. Let we now consider $f(x) = x^3 - x^2 + x$. The function f is an increasing one, and it is such that

$(x-1)^3 < f(x) \leq x^3$, for $x \geq 1$ and the right inequality being strict for $x > 1$.

Indeed, $x^3 - f(x) = x^2 - 1 = (x-1)(x+1) \geq 0$, for $x \geq 1$, this inequality being strict for $x > 1$. On the other side, $f(x) - (x-1)^3 = 2x^2 - 2x + 1 > 0$,for $x > 0$. Consequently, we have the enounced double inequality. We are now able to state that: given T, the integer $r > 1$ such that $(x-1)^3 < f(x) < x^3$ is:

$$r = \left\lceil (f(x))^{\frac{1}{3}} \right\rceil + 1$$

$\lceil y \rceil$ denoting the whole number of y.

5 Conclusion

Random walks requires only local information about the network while they have nice global properties. This makes random walks very useful to determine global information on dynamic networks. Given a graph modeling a network, we have presented two methods to determine cyclic cover time and total hitting time. The first method computes electrical resistance. The second method uses matrix calculus from Markov Chain theory to determine exact hitting time values. We have state an automatically way to compute *total hitting time* for both methods since we have,

$$T = \sum_{(i,j) \in V^2} h_{ij} = m \sum_{(i,j) \in V^2} R_{ij} + \sum_{i \in V} h_{ii}$$

Then, given a network on unknown topology, possibly dynamic, through the total hitting time we are able to determine the number of resources necessary in order to achieve a wide range of problems in distributed computing using random walks.

References

[1] D. Aldous and J.A. Fill. *Reversible Markov Chains and Random Walks on graphs.* (book draft).

[2] R. Aleliunas, R.; Karp, R. Lipton, L. Lovasz, and C. Rackoff. Random walks, universal traversal sequences and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science*, pages 218–223, 1979.

[3] Andrei Z. Broder, Allan M. Frieze, Stephen Suen, and Eli Upfal. Optimal construction of edge-disjoint paths in random graphs. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, 1994.

[4] A Bui, M Bui, and C Lavault. On the hierarchy of functioning rules in distributed computing. *RAIRO operations research*, 33(1):15–27, 1999.

[5] Marc Bui, Sajal K. Das, Ajoy Kumar Datta, and Dai Tho Nguyen. Randomized mobile agent based routing in wireless networks. *International Journal of Foundations of Computer Science*, 12(3):365–384, 2001.

[6] Ashok K. Chandra, Prabhakar Raghavan, Walter L. Ruzzo, Roman Smolensky, and Prasoon Tiwari. The electrical resistance of a graph captures its commute and cover times. *Computational Complexity*, 6(4), 1997.

[7] Don Coppersmith, Uriel Feige, and James Shearer. Random walks on regular and irregular graphs. *SIAM Journal on Discrete Mathematics*, (9(2)):301–308, 1996.

[8] Peter G. Doyle and J. Laurie Snell. *Random Walks and Electric Networks*. 2000 (first edition 1984 Mathematical Association of America).

[9] M. Duflot, L. Fribourg, and C. Picaronny. Randomized finite-state distributed algorithms as markov chains. In *15th Int; Conf. on Distributed Computing (DISC'2001)*, pages 240–254. LNCS, Springer-Verlag, 2001.

[10] U Feige. A tight lower bound for the cover time of random walks on graphs. *Random structures and algorithms*, 6(4):433–438, 1995.

[11] U Feige. A tight upper bound for the cover time of random walks on graphs. *Random structures and algorithms*, 6(1):51–54, 1995.

[12] O. Flauzac, M. Krajecki, and J. Fugere. Confiit: a middleware for peer to peer computing. In *ICCSA 2003 International Conference on Computational Science ans its Applications*. LNCS, Springer-Verlag, to appear, 2003.

[13] I. Foster and C. Kesselman. *The GRID: Blueprint for a new Computing Infrastructure*, chapter 2. Morgan Kauffman, 1998.

[14] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150, 2001.

[15] Amos Israeli and Marc Jalfon. Token management schemes and random walks yield self-stabilizing mutual exclusion. In *9th ACM symposium on Principles of distributed computing*, pages 119–131, 1990.

[16] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Springer-Verlag, 1976.

[17] László Lovász. Random walks on graphs: A survey. In T. Szõnyi ed. D. Miklós, V. T. Sós, editor, *Combinatorics: Paul Erdos is Eighty (vol. 2)*, pages 353–398. János Bolyai Mathematical Society, 1993.

e-Learning and Communities, Supporting the Circulation of Knowledge Pieces

Gerald Eichler

T-Systems International GmbH, Service-Line Systems Integration
Technologiezentrum, D-64307 Darmstadt, Germany
Gerald.Eichler@t-systems.com

Abstract. The life cycle of knowledge decreases dramatically. This influences both, the business life, namely in technical science and the cooperation in virtual communities. While human resources departments look for well developed, process based concepts to identify, plan, and carry out high level qualification according to enterprise's current needs, communities use electronic support to create, share, store and archive their knowledge.
In the past further education was driven by expensive presence seminars. Today a wide range of computer-aided multi-media applications are available, which allow distance learning, even in virtual groups. Multiple criteria are taken into consideration to create efficient and more and more modular curricula.
Elements like Computer and Web Based Training, Virtual Classrooms and Multi-media Content Data Bases are brought together with Knowledge Management techniques, using Internet technology to create an environment, which can be accessed by an easy to handle customized, personal portal anywhere and anytime. This contribution will emphasize multiple views, including learning concepts, technical components and platform development.

1 About Learning in the Digitalized World

"Of course it's true, I saw it on the Internet!" How often this argument is taken to convince someone of the latest news. But there is so much information in the dawn of the network of networks. It does not take a long time to find proof for whatever proof is required for. The information society suffers from an overcrowded information pool with less structure and no quality guarantees.

Today search engines deliver matches at extremely short response times. Result scoring becomes better. Experts look for the latest news on their fields of activity, while newcomers have problems to access the basic knowledge of a community. Often there is less structured information not taking any personal previous knowledge into account. There is still a big gap between authors and consumers of specific contents, although in communities they come closer to each other than elsewhere. The Internet easily allows to form virtual communities, joining people at spreaded locations.

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 48–64, 2003.

This contribution is structured as follows. Chapter 2 emphasizes the combination of e-Learning and Knowledge Management for communities. Basic definitions are given and components are identified. Chapter 3 discusses learning collaboration support and didactic experience, while Chap. 4 concentrates on multiple views for enabling educational events. Events are broken down to modules and their selection. Platforms provide the technical background for content hosting, communication tools, single systems and portals. This is described in Chap. 5. Finally, some hints and conclusions for the future of collaborative learning, emphasizing the modularity and circulation of knowledge pieces, are given in Chap. 6.

2 e-Learning, Communities, and Knowledge Management

Communities are an important enabler to bring *e-Learning* and *Knowledge Management* closer together. While e-Learning mainly concentrates on the content and its consumption process, Knowledge Management can help to structure the content and navigate through it. Figure 1 visualizes the triplet.

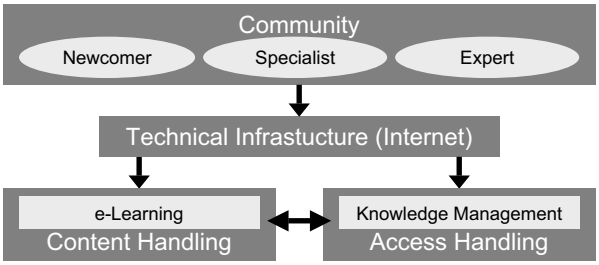


Fig. 1. Relationship between community, e-Learning and Knowledge Management.

2.1 Why Not Take a Lesson between Two Cups of Coffee?

e-Learning has been becoming a buzzword during the recent years. The assumption that electronic tools will completely replace presence seminars is not true. There are a lot of advantages, which can be extracted into a *blended learning* model. This means to select appropriate methodical and technical solutions to fulfill selected steps of a complex educational goal in parallel to the job and daily life. In a very general definition **e-Learning**

- describes learning on demand by means of latest information and communication technologies,
- is based on methodical and didactic teaching and learning concepts, and
- applies multi-media training materials.

Following the history of computer supported knowledge exchange, a tendency from stand-alone over off-line toward on-line network applications can be noticed. Classical single-user application like *Computer-based Training (CBT)* or *Web-based Training (WBT)* are still the core method of content and knowledge distribution for individual asynchronous learning. *Virtual Classroom Systems (VCS)* replace presence seminars as synchronous group-ware application. Communication tools, like *chat* or *newsgroups* support information exchange among learners.

Situation-oriented qualification becomes more and more interesting for both, personal and business life. e-Learning should be seen as an enabler to meet this needs anyplace and anytime. Small and classified knowledge pieces help to access content even in a limited time frame.

2.2 Where to Find a Good Friend Thinking about Similar Problems?

e-Learning leads not necessarily from common learning within large groups to individual knowledge consumption. Even there are many asynchronous components available, which support personal needs, synchronous tools allow to form virtual groups. There are good reasons to share knowledge among interests of a certain topic. *Knowledge Communities* are already a fixed term [1].

Typically, there are two kinds of virtual groups driven by the given business background. *Expert groups* usually are formed with commercial background. They share their views within a very limited area and are known for their expertise. The second group, *communities*, form a loosely league of people with different social background but having a common subject of interest [2]. Differentiated by the grade of proficiency a community unifies newcomers, specialists, and experts under one roof. A *codex*, sometimes called etiquette, defines common rules. They are accepted by all community members. Guidelines help newcomers to orientate and find their place in the new environment. Generally speaking, a **community** is characterized by

- giving people with common interests a home,
- uniting people with interdisciplinary background on a volunteer base,
- self-organisation within defined borders, and
- following limited financial and business objectives.

Beside the social aspects, todays communities rely on a network based *communication infrastructure* to collaborate with each other. An Internet access plus selected well-known e-Learning components e.g. message boards, are appropriate basics therefore.

2.3 What about Professional Assistance?

Contents become richer and richer. New media come into place. A big variety of qualification offers is available. But the orientation on the educational market

becomes more and more complex. *Knowledge Management (KM)* can help to answer the question, what selected piece of knowledge to access at what given situation. At higher level *Learning Management Systems (LMS)* provide control of structured access as a step-by-step orientation for learners. Today, usually the content for educational events is pre-structured following a fixed scheme. KM will support more flexible, user-driven content consumption. Therefore **Knowledge Management** should be defined as

- tools for navigation, administration and maintenance of knowledge,
- means to keep multiple views to the same objects in context, and
- enhanced support for and extension of document management systems and search engines.

Mind maps are already very fashionable to structure personal notes. A typical KM tool is the *knowledge map*, which is quite more than a graphical representation. The appliance of ontology and taxonomy allows quick navigation and flexible relationship presentation in a scalable manner. Experience from graph theory can be applied. To lodge knowledge maps with content, enhanced *meta-indexation* is an important pre-requisite. For details refer to Chap. 4.5.

3 Collaboration Support and Methodology

Knowledge consumption has two different faces, individual learning and qualification in groups. Communities used to share their knowledge as an open source. Each member is allowed to profit from the common knowledge base. As the community's subject is often very specific, knowledge exploitation becomes more and more difficult.

3.1 Evolution in Learning Methods

Depending on the position a community member takes, the grade of familiarity with the given subjects and therefore the language and used terms vary. Just to be informed, basic knowledge is enough, while active participation requires a certain level of proficiency. There are different ways to acquire information. Let's call them *learning methods*. Figure 2 shows three approaches.

Object oriented learning seems to be the easiest way. This is true for common situation. No pre-requisites are required. Occurring questions are answered directly. A single circumstance can be simply imparted. Based on a dedicated object, information are stressed and taken in. Sometimes, parallels to practice are very rare.

Abilities are achieved by *method oriented learning*. Basic learning technologies are imparted independently of the taken object. This is missing often at universities. The knowledge of context allows appropriate assessment of a given situation. Alternative ways to overcome a specific problem are known, even if sometimes decisions are not yet optimal. There is still a lack of experience but the "tools" are known.

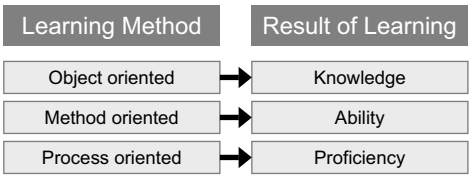


Fig. 2. Development of learning methods and their results regarding practical applicability of contents.

Only repeated step-by-step doing in the right environment leads to proficiency. *Process oriented learning* simulates the real world in abstracted models. Key reactions are trained, which is time consuming.

Besides the subject itself, ergonomics of learning plays a major role. There is a high risk to constrain human creativity by monotone repeated activities. Therefore, a mixture of learning methods is the optimal procedure.

3.2 Perceptual Results

The sustainability of learning or *learning awareness* is a measure for the percentage of consumed facts a learner can keep in mind. There are different ways for knowledge consumption, depending on the used materials and media. Passive or perceptional methods support a high delivery rate, but result in a low recapitulation rate. Active or productive methods require a high grade of interaction and full concentration. The *recapitulation rate* is defined as the ratio of known by heart facts to presented facts. Figure 3 gives quantitative details.

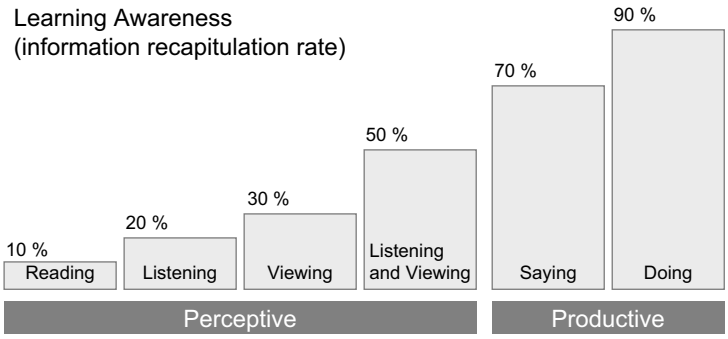


Fig. 3. Recapitulation rate for selected perceptive and productive learning methods over a medium time frame.

The second fact, which influences the recapitulation rate, is the learner's type. While some people rely on extended text passages, other prefer graphics or

mind maps for visualization of inter-relations. In general, the parallel appliance of single methods increases the recapitulation rate. It is the task of didactic concepts to select appropriate combinations to optimize the learning awareness.

3.3 Basic Support for Collaborative Work

Electronic support for communities becomes more and more fashionable. It starts with organisational management support. Experiences show that currently easy to use and free of charge single tools are preferred. This meets the original idea of *Basic Support for Cooperative Work (BSCW)*, which is well-known since the mid-nineties. The system of the same denominator by Fraunhofer FIT and OrbiTeam Software GmbH is a simple web based groupware exchange platform [3]. Both, *client-server solutions* and *peer-to-peer computing* are of potential interest. There is a need for both, asynchronous and synchronous communication support. *Asynchronous* communication tools are

- e-mail,
- newsgroups and discussion forums,
- organizers, and
- shared work space for document storage and evaluation.

Synchronous communication tools cover

- chat,
- audio and video conferencing,
- streamed broadcasting, and
- application sharing e.g., white-board, web-safari.

There is a trend to make use of typical e-Learning tools, like CBT or WBT for self-study purposes. The VCS becomes of interest as an umbrella for collaborative communication applications. Professional tools like the VCS Centra 7 by Centra [4] are already very sophisticated, but still too expensive for community use.

Today customer premises equipment is more powerful. PCs in combination with fast Internet access over bundled ISDN channels or DSL access offer enough capacity for streaming application in acceptable quality. Mobile devices make collaboration independently of the place and offer location based services. They tend to use packet based data exchange technology, which enables better pay-per-use models.

3.4 Roles and Interaction

Classical e-Learning solutions have a clear distinction of roles of people who are involved in the interaction. This concept is usually driven by security guidelines and access rights of the technical environment.

Classical Roles. The typical roles for people who share an e-Learning system are

- learner or trainee, who consumes the content,
- trainer, who leads educational events and presents the content,
- author, who creates and structures the content, and
- administrator, who schedules events and keeps the system running.

It is possible to assign multiple roles to a single person, respectively a single account, which is unique by a login-password tuple.

New Roles. There is no need to strictly distinguish between the classical roles within community's life. Every single person finds its own place within a mixture of such roles. Having a look to the roles, three dimensions can be identified, which are visualized in Fig. 4. The grade of proficiency spans a triangle. The covered area is a measure for deepness of integration in the community.

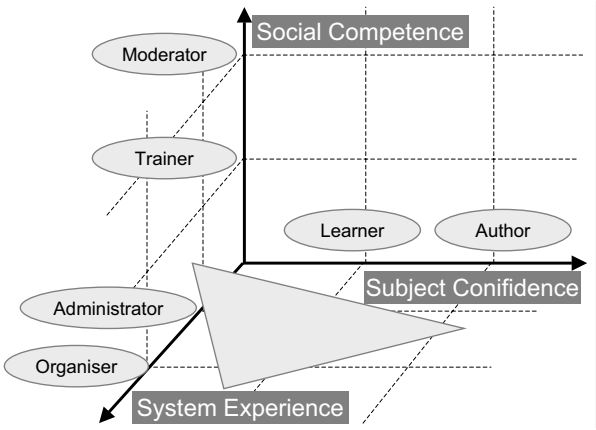


Fig. 4. Roles of community members in a multi-dimensional view of competence, confidence and experience.

Firstly, *subject confidence* can be assigned in increasing grade to newcomers, specialists and experts. As soon as a learner reaches an advanced level, he or she can become an author to re-establish new content for community use. It is practically a difficult step, because authoring tools are in a very early stage, while learner's interfaces are often already provide easy and intuitive handling.

Secondly, *social competence* keeps a community together. The classical role of a trainer mutates to a moderator, who is not leading activities authoritarian but acting as a well-accepted broker.

Thirdly, *system experience* is required. There is no system administrator anymore, who has the universal power to play with users. What communities

are looking for, is a kind of organizer who takes care of all administrative and technical belongings.

4 Learning Events

4.1 Planning Criteria

The successful planning of learning events requires the compliance of multiple criteria. Figure 5 structures them in different areas. The categorization shows that different experts are required to meet all needs step-by-step [5]. Besides contents, methodology and organisation the feedback is a very important fact for professional learning events with high sustainability.

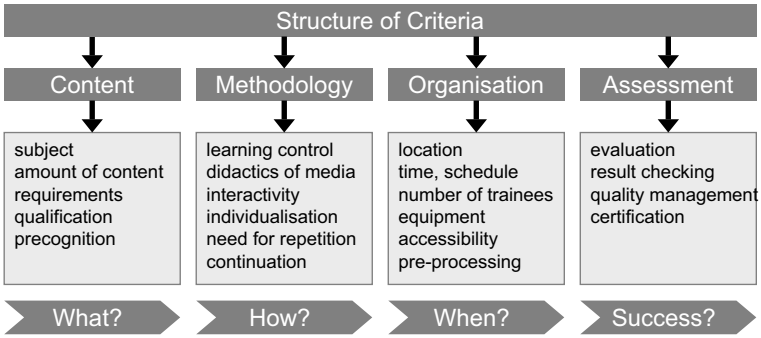


Fig. 5. Multiple criteria for the planning of e-Learning events.

Usually, communities do not rely on professional further training. But it makes sense to be aware of success criteria of professional curricula to create new content for the members and change or share it with other interest groups. The idea that members write for members is one of the community's strength.

4.2 Cutting Down the Monolith

Classical further training is characterized by one more multiple day seminars once or twice a year. Catalogs provide predefined training programs, where the learner can select from on first come first serve base. Booking has to be performed long times in advance. Organizational criteria, see Chap. 4.1, have to be met. Optimal capacity planning is required to sell programs on reasonable prices. Educational managers have to do quite a difficult job. Neither for short living business nor for communities such long term pre-planning is practical and efficient.

En Block – The Fixed Event. Multiple day presence seminars run out of fashion not only because of travel costs. Todays business life makes it quite difficult to coordinate the timing of single persons, independently, they are spare time activists, specialists, or managers. On the other hand, if possible, it is a great advantage to get the head free off any other business and to be faced with the lecturer and other persons addressing the same topic. Last but not least smalltalk, while having a glass of wine together, is an important fact.

The Chain – Guided Step by Step. There is a trend to split up single events into a course chain. This does not really solve the timing problem, because learners have to reserve multiple slots over a certain time. Missing one of the events, the entire chain becomes useless. The good news is that some of the single events, lets call them *modules*, can be scheduled individually within a given period. This is typical for CBTs or WBTs or an asynchronous group work. As shown in Fig. 6, a good mixture of presence and e-learning modules combines advantages of both scenarios, known as *blended learning*.

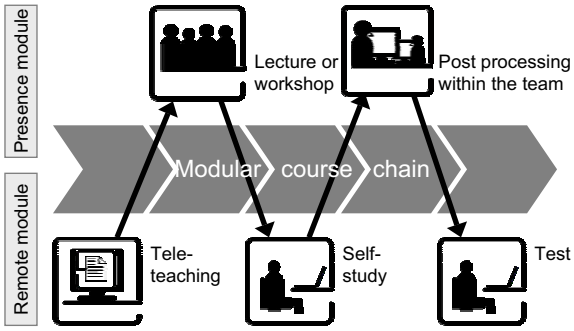


Fig. 6. Presence and remote modules form a pre-defined course chain.

The Matrix – Selection of Alternatives. The course chain relies on a fixed schedule, no alternatives of selection are given to the learners. Depending on the learner's type, see Chap. 3.2, the learning success will be different. Options for the selection of different modules supporting the same step of the chain help to overcome this disadvantage of the chain approach.

Writing the steps on the x-axes and noting the different media, respectively learning method on the y-axes a matrix is spanned. There is no need to fill this matrix completely with modules. The idea is to provide an offer of appropriate options per step. A learner has to complete at least one module to move to the next step. For personal interest it is quite possible to work trough an parallel choice optionally. In Fig. 7 the dotted lines in step 2 symbolize this piece of freedom.

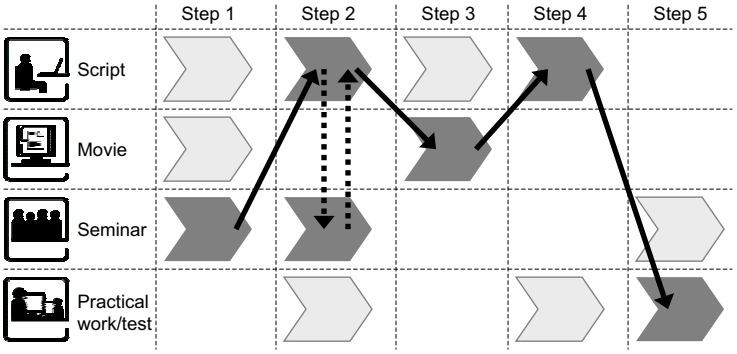


Fig. 7. Matrix of learning modules offers to select between given options per step.

The Map – Hierarchical Topic Clouds. While the chain approach is driven by organizational aspects, the matrix adds some more methodology. However, in Chap. 4.1 the content was mentioned as the first criteria for the design of a learning event. The topic cloud approach follows this thesis. By means of a *topic map* modules of similar content are clustered to clouds, as shown in Fig. 8. If required this hierarchical classification can be done on several levels. Prerequisite are fairly small modules, sometimes mentioned as *knowledge bits* with a clear description by indexation, see Chap. 4.5. A topic map is a graph, where the modules are the vertexes. Edges are drawn if there is a content relationship between two modules. This edges form the base for the selection of a personal virtual learning path, as described in Chap. 4.3.

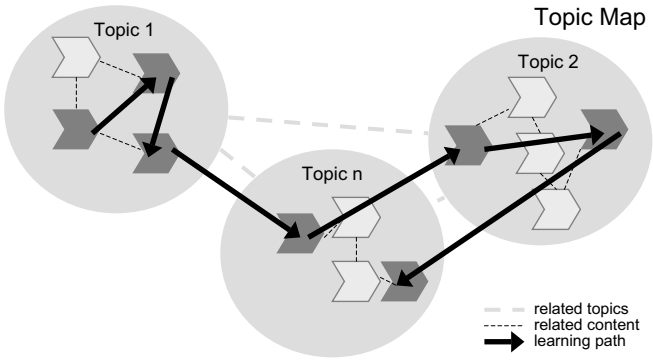


Fig. 8. Learning modules are linked to each other and clustered to topic clouds in a topic map.

4.3 Individual Defined Learning Path

LMS, see Chap. 2.3, usually support the chain approach. It is a consequent step-by-step description. Sometimes alternative branches are controlled by intermediate tests. This helps learners to be challenged at the right level and to save time to reach the goal.

By means of topic maps, as introduced in Chap. 4.2, new flexibility is given to decide about an individual detailed training plan, the *virtual learning path (VLP)*. The VLP is a personal navigation route through the topic map, as depicted in Fig. 8. The learner should be given an entry topic and a list of topics, which have to be passed. Within a certain topic the learner gets the freedom to navigate individually through available modules, which should be small but closed pieces. Edges indicate, which module might be accessed next. The VLP should be traced for later assessment. For breaks the current position is stored in the individual settings.

4.4 Creation of New Content

Vocational and further training is mainly limited to knowledge consumption. Content is provided by professional authors and trainers. Within communities distributed knowledge and experience exists. Therefore, modular e-Learning offers a great chance. While following the VLP a learner sometimes will identify missing content in a topic cloud, where he or she has already good background knowledge. Public annotation can be provided.

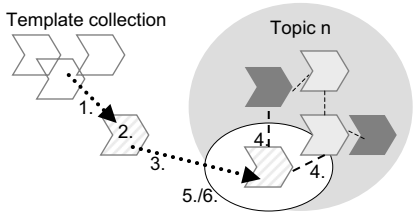


Fig. 9. Steps for the creation and inclusion of new modules in a topic map.

By means of easy to handle authoring tools, new modules can be created. Appropriate templates should be available to reduce the administrative overheads and to guarantee a unified appearance of new modules. They should perfectly fit into existing content. Some steps are required to make the content available for others.

1. select of an appropriate module template from a collection,
2. create the new content by filling the template,

3. upload the new module for quality check,
4. propose linking of the new module in the topic map, either by a graphical editor or by detailed indexation,
5. let the module and links check by a moderator/topic coordinator,
6. make the module and links public and accessible via the topic map.

The numbers in Fig. 9 illustrate the process of integrating new modules using the same numbering.

4.5 Content Indexation

The importance of indexation was mentioned several times during the previous chapters. Indexation means to give a document an additional description by filling in meta index information. Office programs already provide a large number of meta data fields for documents, but the majority of users ignores this opportunity. However some statistical fields like file name or authors initials are filled automatically.

Indexation is both, a chance and a risk. The chance is the right classification of a document, done by the author. The risk is misleading indexation to push web content identified by search engines, which rely on the truth of index values.

Module indexation should be based on a well-defined mandatory set of attributes. Standards like *Learning Object Metadata (LOM)* or *Sharable Content Object Reference Model (SCORM)* already provide a conceptual variety [6]. SCORM is a collection of specifications adapted from multiple sources to provide a comprehensive suite of e-learning capabilities that enable interoperability, accessibility and re-usability of Web-based learning content [7]. A carefully selected subset is still enough. The strength of indexation does not lie in new extraordinary attributes but in the KM based combination of simple categories. This requires also a limited pre-defined number of attribute values. Drop down menus can support the selection. Multiple choices per attribute should be supported. The *German Initiative for Networked Information (DINI)* provides recommendations for useful indexation of scientific materials [8]. Libraries maintain index lists.

Topic maps and indexations have a close relationship. There are two different ways. Detailed indexation can be used to create automatically a topic map by means of KM tools. On the other hand a graphical editor can be applied to create a topic map, while indexation is done automatically in the background, depending on the crated relations of modules and topics. The second proposal is easy to handle for new modules created by occasional authors, who do not want to deal with attribute lists and their pre-defined values. Professional content creators should have designed a taxonomy or ontology in advance, where index lists are a powerful support.

5 Platforms

5.1 Single Systems

There is quite a large variety of single systems on the market. Professional client-server systems compete with peer-to-peer computing for individual use. In Chap. 3.3 single supporting tools are mentioned. They can be classified in four groups. Dependencies are shown in Fig. 10.

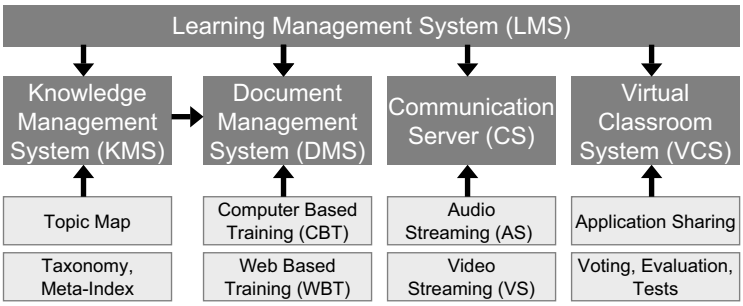


Fig. 10. Classification and relationship of single system groups supporting e-learning.

Document Management Systems (DMS) stand for an asynchronous data base to access and share content. KM is an add-on for navigation, see Chap. 2.3. *Communication Systems (CS)* support synchronous and asynchronous interaction between learners and trainer or learners, respectively community members to each other. Today basic support is already given by components, which are shipped with the operating system or office packages. Streaming applications come into fashion. A development of recent years are *Virtual Classroom Systems (VCS)*. The idea of application sharing among multiple end-points is organized together with typical classroom activities e.g., raise hands, vote with "yes" or "no", or fill multiple choice tests. Emoticons e.g., clapping hands or smileys are used by learners with great pleasure. Evaluation Systems could be considered an extra category but should be covered by VCSs. The roof is given by *Learning Management Systems (LMS)*. They abstract from the technical thinking and provide a personalized process based view.

5.2 Integrated Solutions

To abstract from a single system's view an integrated solution is preferred. Selected single systems, as classified in Chap. 5.1, form the technical core. A system platform unifies the access to different systems. This linearizes the number of necessary interfaces, because single systems only have to talk to the middleware

and not directly to each other. XML based message exchange is preferred for inter-communication. Common components, like a user data base can and should be shared.

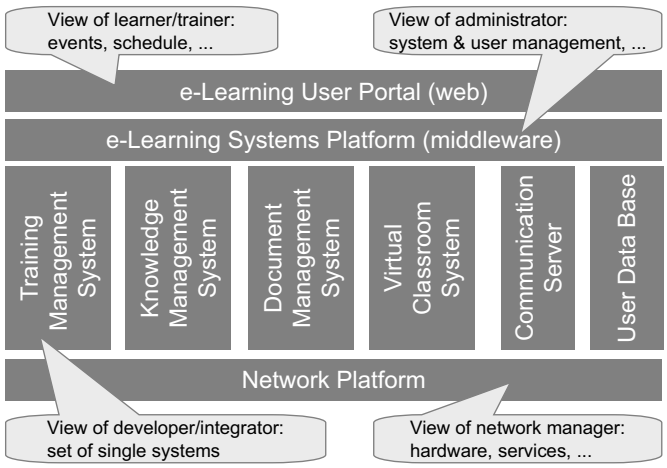


Fig. 11. Multiple views to an integrated e-Learning platform solution.

The only interface to the learner is a *web based portal*, see Chap. 5.3, which is established on top of the middleware. Administrator, integrator or network manager have a different prime focus with their views, as depicted in Fig. 11.

5.3 User Portal

The user portal is the most important part for success of a learning system. Intuitive look and feel increases user acceptance of this central entry point. The typical appearance of web sites should be met. Reaction times should be less than a second.

No extra applications or even secure and small smart clients should be installed locally. Administrator permission rights should not be required for plug-in installations. Furthermore, there is a long list of guidelines regarding user groups and mandates, personalization, optional integration of additional information services, help desk, or corporate design, to give just some examples.

The registration and access procedures are of potential interest to minimize personal overheads for administration. Self-registration processes should be supported. A user should be assigned a single account at portal level. This account consisting of a login and password, known as *Single Sign-On (SSO)*, should imply several access rights

- membership of groups,
- access to subscribed single systems,

– individual learning plan with status report, and
– personalized portal configuration.

Access to unique events or evaluation sheets is typically done by single valid tokens, which can be distributed on demand by e-mail.

6 The Future of Collaborative Learning

Training and learning do not have a good image in times of economical weakness. Basic investments are required to establish powerful solutions. But technology is not enough. Various specialists, expert and professional, trainer and designer, administrator and manager, technician and integrator have to come together to make e-Learning run.

Communities can participate at solutions at reasonable prices. They are, like universities, a perfect source to create new content for life-long learning. There is a new business field for *Learning Service Providers*. This is comparable to an application service provider who feeds multiple mandates. High-level content is a very expensive good but required add-on, which should be shared. Rare experts can give remote lessons or be consulted on demand.

6.1 Modularity

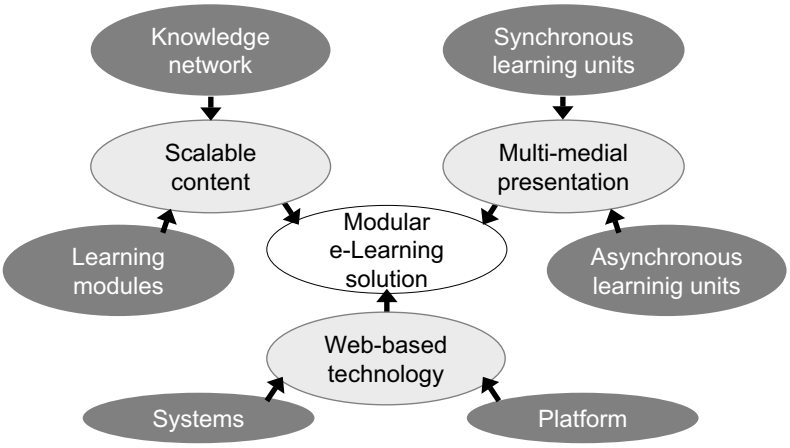


Fig. 12. Modular solutions for content, presentation and technology increase flexibility of e-Learning applications.

Modularity is one of the keywords to increase the flexibility of e-Learning solutions and to make them profitable for communities. Creation of new content must become as simple as consumption of learning modules. Modularity

addresses not only the content itself but also its presentation and the underlying technologies. Even the philosophic view demands unity of content and style re-usability of modules in various contexts requires technically a separation of both. Just for consumption it is dynamically linked together. This introduces the chance to create a homogeneous customized appearance of modules along the *Virtual Learning Path*.

6.2 Circulation of Knowledge Pieces

Created modules are not fixed entities. Every time a piece of knowledge is employed to guide the fulfillment of a given task it follows a cycle of different steps. Figure 13 emphasizes typical states.

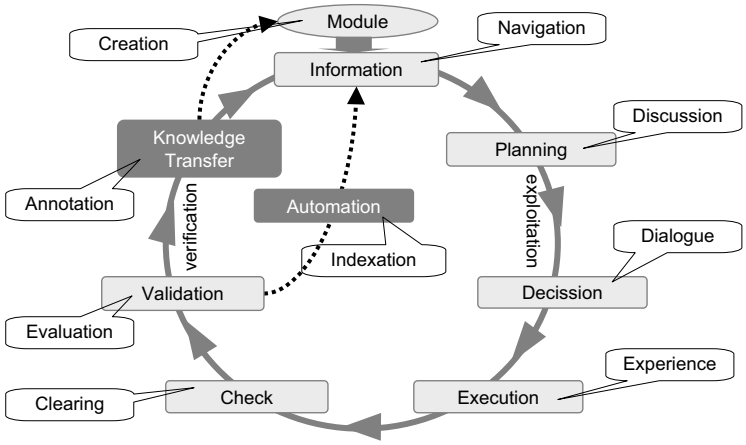


Fig. 13. Knowledge pieces follow a circle of processing.

The exploitation phase on the right hand side should be followed by a recycling phase on the left hand side. This includes a verification of the achieved results and an assessment to feed back experiences either to annotate existing information or to create new modules. The navigation can be improved by enhanced indexation of contents.

6.3 Next Steps

There are multiple directions of further work. All statements given here were done under the assumption of a peaceful and cooperative interaction of all involved partners.

Digital Rights Management (DRM). Fortunately, there is a trend to make knowledge public like a recent initiative of the MIT shows. Increasing number of on-line dissertations worldwide meets this goal too. But intelligent property is a big issue. It is still unclear, whether DRM in conjunction with micro-payment will be able to handle this problem satisfactory. Authors must be motivated to generate valuable content. More general, liable for costs contents arise a global debate, while often education is considered to be free of charge.

Knowledge Management (KM). KM is still in an early stage. The use of graphical tools makes it assessable even for unexperienced users. Appropriate indexation needs more experience. Edges and vertexes in learning topic maps will describe various properties of relationship. Views for different purposes can be created.

Authoring Tools and Module Templates. The separation of content and style requires more intelligence for multi-media contents. The creation of animations should not require specialists e.g., a flash insider. Authoring and annotation tools look for ease of use and system independence.
This contribution could only give a rough overview of multiple factors that make e-Learning to be a success. The author believes that communities have an optimistic future as a home for peacefully together working people who are sharing their competence on an open-minded base.

Many thanks to the conference board for the invitation, to T-Systems for supporting scientific research and to all my friends and colleagues who made this presentation possible with their fruitful comments and idea.

References

[1] Schmidt, M.P.: Knowledge Communities. Addison-Wesley, München (2000)
[2] Wenger, E.: Communities of Practice. Cambridge University Press, Cambridge (1998)
[3] OrbiTeam, Fraunhofer: Basic Support for Cooperative Work (BSCW). OrbiTeam Software GmbH, URL http://www.orbiteam.de/en/. (2003)
[4] Centra: Centra 7, URL http://www.centra.com/products/index.asp. (2003)
[5] Daly, D., Scott, A.: Best Practices for Advanced Distributed Learning. Joint ADL Co-Laboratory, URL http://jointadlcolab.org/. (2002)
[6] LTSC: Learning Object Metadata (LOM). IEEE P1484.12 Learning Object Metadata Working Group of Learning Technology Standards Committee, URL http://ltsc.ieee.org/wg12/. (2003)
[7] ADL: Sharable Content Object Reference Model (SCORM). Advanced Distributed Learning Initiative (ADL), URL http://www.adlnet.org. (2003)
[8] DINI: Electronic Publishing in Higher Education. Deutsche Initiative für Netzwerkinfomation e.V. (DINI), URL http://www.dini.de/. (2002)

Where to Start Browsing the Web?*

Dániel Fogaras[1,2]

[1] Department of Computer Science and Information Theory,
Budapest University of Technology and Economics,
H–1521 Budapest, Hungary
[2] Computer and Automation Research Institute,
Hungarian Academy of Sciences (MTA SZTAKI)
11 Lágymányosi u., H–1111 Budapest, Hungary
fd@cs.bme.hu

Abstract. Both human users and crawlers face the problem of finding good start pages to explore some topic. We show how to assist in qualifying pages as start nodes by link-based ranking algorithms. We introduce a class of hub ranking methods based on counting the short search paths of the Web. Somewhat surprisingly, the Page Rank scores computed on the reversed Web graph turn out to be a special case of our class of rank functions. Besides query based examples, we propose graph based techniques to evaluate the performance of the introduced ranking algorithms. Centrality analysis experiments show that a small portion of Web pages induced by the top ranked pages dominates the Web in the sense that other pages can be accessed from them within a few clicks on the average; furthermore the removal of such nodes destroys the connectivity of the Web graph rapidly. By calculating the dominations and connectivity decay we compare and analyze the proposed ranking algorithms without the need of human interaction solely from the structure of the Web. Apart from ranking algorithms, the existence of central pages is interesting in its own right, providing a deeper insight to the Small World property of the Web graph.

1 Introduction

Recent years witnessed an extensively developing interest on link-analysis algorithms to improve textual based Web search engines. Inevitably, the most influential results on this field are HITS [15,8] and Page Rank [7] algorithms; since then many improvements and extensions appeared [9,17,13], see [5] for a comparative study. HITS assigns a pair of scores to the pages belonging to a query. The authority score of a page is proportional to its importance, and hub score describes the quality of a page as a link collection within the topic. Page Rank, on the other hand, overall quality scores that are applied in any query search later. Following HITS' terminology, the Page Rank scores act as overall authority values of pages independently from any topic. Overall hub scores of the whole Web, however, earned less attention in the link-analysis literature. Remarkable exceptions [18,6] evaluate the rank of a page by summing the rankes

* Research is supported by grants OTKA T 42559 and T 042706 of the Hungarian National Science Fund.

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 65–79, 2003.

of those linked by the page itself iteratively, which in turn acts as some hub score over the Web.

In the first part of the paper we focus on finding good starting points for browsing from which a large number of pages can be accessed within a few clicks. To express the quality of pages as starting points, we define overall hub scores of the Web, which can be evaluated for the whole Web graph independently from queries. For instance, the hierarchically ordered link collection www.dmoz.org would be given much higher credit as a hub than for example the site www.weather.com with good quality content but only a limited amount of linkage outside its own domain. The Web does not only provide explicitly defined hierarchical link collections that are easy to find, but also contains several implicitly evolving search trees by the nature of hyperlink evolution. The root of such trees are excellent start nodes for browsing, but authority based ranking schemes rarely reveal such root pages.

Clearly it is advantageous to start browsing the Web from a page, if short sequences of clicks from that page lead to as many other pages as possible. We introduce *Start Rank*, a family of hub ranks through counting the search paths departing from each Web page. User defined parameters tune the credit given for each search path. Path-counting method appears first in the classical paper [14] about social networks defining an influence measure, *standing* of persons that is closely related to authority measure of Web pages. We slightly generalize their path-counting technique and apply the method to estimate the hub quality of pages. Notice that hub scores of HITS are proportional to the authority values of directly accessible pages, while Start Rank takes into account the pages accessible in more than one click on hyperlinks.

As a candidate for overall hub ranking, we investigate *Reversed Page Rank* that is computed after reversing the direction of all the hyperlinks, similar ranking algorithms were proposed in [6]. We formally prove that Reversed Page Rank is a member of the family of Start Rank scores supporting the assumption that Reversed Page Rank scores express hub quality. The equivalence of Page Rank and path-counting rank is interesting in its own right stating that Page Rank generalizes in-degree rank by taking into account longer than one-step paths.

Evaluating and comparing the performance of link-analysis algorithm seems hard, since there is no formal definition for the "qualities" of a Web pages. Typical practical approaches are based on expert evaluation [2], volunteer testing [21], notions of "spam" [10] or query examples [5], all depend on human judgment. In a theoretical approach one can formally analyze certain desirable features of ranking algorithms such as stability [3,19], locality and monotonicity [5]. These features are natural requirements for ranking algorithms, but neither of them acts as an objective measure of the quality of link-analysis algorithms.

We propose *centrality analysis* as a graph based tool to provide quantitative justification and comparison of the introduced ranking methods. The key idea is that top ranked pages play a central role in maintaining the connectivity of the Web graph. For any ranking over the nodes of the Web graph, the centrality of the set of top ranked pages can be evaluated numerically, yielding a qualification of the ranking algorithm. Although such a qualification only classifies the top scores assigned for the pages, we

believe that centrality analysis is an important step towards the automatic evaluation of ranking algorithms.

The centrality of a set of pages is either measured by *domination*, the average distance from the set to the other pages; or by the decay in the *diameter* of the Web graph after the removal of the central nodes. The former centrality measure is applied for hub ranking schemes, since from the set of strongest hubs the whole Web should certainly be available within a few clicks on the average. The latter notion of centrality will show the quality of a ranking algorithm that gives credit for popular hubs—nodes that are contained in a large amount of search paths of the Web. Our notions of centrality were motivated by the NP-hard combinatorial optimization problem k-domination [11] and by the experiments of [1] measuring the failure tolerance of real networks against removing the largest degree nodes.

Besides qualifying the outputs of ranking algorithms, centrality analysis also provides a deeper insight to the *small world phenomenon*, empirically proved for many implicitly evolving networks including the Web graph [4]. A network is referred to as a small world, if the diameter is low and the number of edges in the network is relatively small. Centrality analysis experiments reveal that a surprisingly small number of central nodes are responsible for the connectivity of the Web graph. Such experiments were pioneered by [1] implying that only a small set of largest degree nodes maintains the connectivity of small world networks. In our centrality analysis experiments we strengthen the results of [1] by showing more centralized nodes than the pages with largest degrees.

Our experiments containing both centrality analysis and query search examples were conducted on the `.ie` domain, the Irish Web. While this portion of the Web provides a computationally feasible test-bed, the contextual structure of a national domain will not differ so much from the entire Web that would result in significant bias in the experiment. Ranking is performed over the collection of near one million pages crawled in October 2002; however for keyword searches we also relied on queries to Google [12].

2 Hub Scores of the Web

Finding a good start page is a critical part of browsing the Web: it is clearly worth starting from a site from which a large amount of content can be reached within a few clicks. Slightly modifying the notion of Kleinberg's HITS algorithm [15] we refer to such pages as *good hubs*.

In this section we introduce Start Rank as a family of hub scores to measure the quality of pages as start nodes. Then we show one member of this family easily computable by slight modifying Page Rank. Finally, some combinations with other ranking algorithms are proposed.

2.1 Start Rank

Start Rank assigns a hub score for each web page on the basis of counting the search paths originating from the page in question. Each search path is taken into account by

a weight depending on the value of the target page, the length of the search path and the hyperlinks occurring in the search path. Note that Start Rank naturally generalizes out-degree as the simplest measure on the hub quality of pages, since out-degree counts all the one-step walks from each page.

The actual Start Rank scores are determined solely from the structure of the Web graph and from the following three user defined parameters.

- The *length weight function* assigns a real weight $\ell(i) \geq 0$ for each length $i \geq 0$. The requirement that longer search paths generally worth less than shorter paths can be achieved for a Start Rank by setting a monotone decreasing length weight function. Furthermore, to eliminate the false effect of extremely long search paths containing a large amount of cycles, it is reasonable to choose zero length weight beyond a threshold. In most of what follows exponentially vanishing length functions are employed with expected value falling into the range 5-15.
- The *target value* $t(v) \geq 0$ of a page v emphasizes the credit that is given for a search path for finding v. Setting the target value identical over the Web pages implies that all pages are treated equally worth as targets. Alternatively, an overall quality measure, such as the Page Rank [7] can be chosen as target value for each page. Then a node obtains high Start Rank, if a large amount of search paths lead to high quality pages from the node in question. Another approach is to set the target value topic specific by giving positive value only for a collection of pages inducing a topic of the Web.
- The *link factor* $m(u \to v)$ assigns a real weight for each hyperlink $u \to v$ of the Web. The appropriate choice of $m(u \to v)$ is inversely proportional to the effort spent by a surfer to select the link from the page u, when proceeding in a search path. For example, the effort can be measured by $d^+(u)$, the number of out-going links from page u, thus $m(u \to v) = \frac{1}{d^+(u)}$ can act as a link factor. More intimate link factor settings take into account the position or size of the anchor text of the hyperlink in the HTML document.

Definition 1 *For given user defined parameters –length weight function, target values, and link factors– the* weight $w(P)$ of a search path P *with length* i*, target node* v *is defined as follows,*

$$w(P) = t(v) \cdot \ell(i) \cdot \prod_{e \in P} m(e),$$

where the product is taken over all link e *contained by the path. The* start rank $SR(u)$ *of a node* u *is*

$$SR(u) = \sum_{P:\ u \rightsquigarrow v} w(P),$$

summing over all paths originating at u*.*

In the rest of this section, we show that the n-dimensional *Start Rank vector* $\underline{\mathrm{SR}}$ can be expressed as a linear combination of matrix powers, where n denotes the number of Web pages. Let M denote the n-by-n matrix with entries $M_{v,u} = m(u \to v)$ for each link $u \to v$; and $M_{v,u} = 0$, if the link $u \to v$ does not exist. (Equivalently,

M is obtained by transposing the adjacency matrix of the Web graph and by replacing each 1 entry with the link factor corresponding to the directed edge.) Furthermore, by introducing the $\underline{t}$ notation for the n-dimensional row vector of the target values, the weights arising for search paths with length i are $\ell(i) \cdot \underline{t} \cdot M^i$, thus the start rank scores can be expressed as

$$\underline{\mathrm{SR}} = \sum_{i=0}^{\infty} \ell(i) \cdot \underline{t} \cdot M^i. \qquad (*)$$

Evaluating such a formula seems hopeless due to the huge dimensions of M, however, the complexity of multiplying a vector with M is proportional to the number of non-zero entries of M, or equivalently the number of hyperlinks of the Web. Such a multiplication can be performed by external memory implementation, similarly to a Page Rank iteration [7]. Thus, if the length function vanishes for numbers over k, then the $\underline{t}M^i$ vectors can be evaluated with k external memory iterations even for the whole Web graph.

2.2 Reverse Page Rank

Since Page Rank (PR) acts as a successful authority score over the Web pages, one may intuitively feel by symmetry that reversing the direction of the hyperlinks and then applying PR yields an overall hub score of the pages. To justify the statement we formally prove the equivalence of reverse Page Rank with a special case of Start Rank scores with appropriate user parameter settings.

For the sake of simplicity in the rest of the paper, we assume that nodes with zero in- or out-degrees have been removed from the Web graph. Furthermore, *reversed Web graph* refers to the graph obtained from the Web graph by reversing the directions of the edges.

First, we recall the definition of PR scores defined on the Web graph through the *random surfer model* resembling the behavior of human users. The surfer takes a random walk visiting the Web sites by selecting the next page according to the following rule: with probability $1 - d$, the next page is chosen from those pointed by the currently visited page; and with probability d, it is selected from all the pages according to some *jump distribution* independently from the currently visited page. Intuitively, the above *damping factor* d is the probability that the random surfer gets bored and restarts surfing; in practical applications it is set to $d \approx 0.1 - 0.2$. The jump probabilities describes the preference of the random surfer among starting nodes to jump; in the simplest case this is uniform over all the Web pages. The random surfer model yields a Markov chain and the PR of a Web site is defined as the probability of the page in its stationary distribution [7].

Definition 2 *For given damping factor and jump probabilities, the* reverse Page Rank (RPR) *is defined as the PR computed on the reversed Web graph.*

Similar to Page Rank implementation [7], RPR can be computed by the power iteration method, and it can be evaluated for such an enormous input as the Web graph. RPR can

be easily interpreted in the random surfer model, with the modification that the random surfer follows the links backwards. However, the interpretation does not support the assumption that RPR is useful as a hub score—in the rest of this section we deduce that RPR is a member of the family of start rank (SR) scores.

Theorem 1. *The RPR with damping factor $0 < d < 1$ and given jump probabilities is equivalent to a SR with the following parameter settings. The length weight $\ell(i) = d \cdot (1-d)^i$, the target values are identical to the jump probabilities and the link factor $m(u \to v) = \frac{1}{d^-(v)}$ is inversely proportional to the in-degree of v.*

Proof. Let $\underline{j}$ denote the n-dimensional row vector of the jump probabilities and J the $n \times n$ matrix with all rows equal to $\underline{j}$ where n is the number of web pages. Let $\underline{\text{RPR}}$ and $\underline{\text{SR}}$ denote the RPR and SR vectors. Furthermore the stochastic matrix M is obtained from the adjacency matrix of the reversed Web graph by normalizing its rows. Note that normalization is equivalent to multiplying the entries of the adjacency matrix with the corresponding link factors.

For the transition matrix Π of the Markov chain defined by the random surfer model the following equation holds,

$$\Pi = dJ + (1-d)M.$$

Since RPR is the stationary distribution,

$$\underline{\text{RPR}}\ \Pi = \underline{\text{RPR}}. \qquad (**)$$

In order to show that the equation $\underline{\text{RPR}} = \underline{\text{SR}}$ holds, we will prove that $\underline{\text{SR}}$ satisfies (**). The $\underline{\text{SR}}$ probabilities can be expressed by equation (*),

$$\underline{\text{SR}} = \sum_{i=0}^{\infty} d(1-d)^i \underline{j} M^i,$$

since the length distribution is geometric with parameter d. By substituting this into (**)

$$\begin{aligned}
\underline{\text{SR}}\ \Pi &= \underline{j}\Big(\sum_{i=0}^{\infty} d(1-d)^i M^i\Big)(dJ + (1-d)M) \\
&= d\,\underline{j}J + \underline{j}\sum_{i=1}^{\infty} d(1-d)^i M^i \\
&= d\,\underline{j} + \underline{j}\sum_{i=1}^{\infty} d(1-d)^i M^i \\
&= \underline{j}\sum_{i=0}^{\infty} d(1-d)^i M^i \\
&= \underline{\text{SR}}.
\end{aligned}$$

The second equation comes from the fact that the matrix $N = \sum_{i=0}^{\infty} d(1-d)^i M^i$ is stochastic, and $NJ = J$ holds for any stochastic matrix, as the rows of J are equal. Similarly $\underline{j}J = \underline{j}$ was applied for the third equation.

Finally, we mention that a similar statement holds for the original PR citation index showing that the PR of each page can be expressed as the weighted sum of all paths arriving at the node in question. Hence PR generalizes the simple in-degree rank by taking into account all the in-coming walks not only the one-step paths.

2.3 Mixed and Aggregate Ranks

We investigate the alternatives to combine Reverse Page Rank (RPR) with other ranking strategies to obtain refined quality measures on Web Pages. From the several possible options, we especially focus on combinations with ordinary Page Rank (PR) — for more general aggregating methods we refer to [10].

The RPR of each page counts the short search paths leaving from the actual page, and the credit given for a target page can be tuned by setting the target value or equivalently the jump probability of the target as stated in Theorem 1. We propose the following methods for tuning the jump probabilities (target values) of RPR.

- *Uniform RPR* algorithm performs iterations with uniform jump distribution over the Web pages. Such a choice of jump probabilities raises the hub score of pages from which a large amount of nodes can be accessed, however the qualities of the accessed pages are not taken into account. In what follows, we always refer to uniform RPR, if the jump probabilities are not defined explicitly.
- *Popular RPR* algorithm precomputes ordinary PR, and then performs RPR iterations, where the jump probabilities of the nodes are set to the precomputed PR scores. By the assumption that ordinary PR measures the quality of pages, popular RPR will be raised for those pages from which a large amount of high quality content can be accessed within short click streams. Notice the analogy with HITS algorithm [15], where the hub score of a node is equal to the sum of the authority scores available with one step. Popular RPR refines this idea by taking into account the authority scores of nodes available in more than one step with exponential decreasing relevance in the number of clicks.
- *Personalized RPR* assigns non-zero jump probabilities only for the members of a certain topic of the Web following the idea of [20] originally proposed for PR. Personalized RPR scores then express hub quality only in a certain topic. Such approach seems practical for query searches or clustering, while personalized RPR would require on-line computation over the entire Web graph for each topic query.
- *Topic sensitive RPR* acts as an off-line alternative of personalized RPR by computing RPR with a few topic specific jump distributions belonging to some low-dimensional basis of the topic-space. Then, hub scores of an arbitrary topic are evaluated as some linear combination of the basis hub scores, which is practically computable on-line. The method was introduced in [13] for PR and the adaptation is straightforward for RPR.

Fixing the jump distribution with one of the above methods RPR algorithm yields scores expressing the quality of pages as hubs. Such score may present as a component of some overall quality measure of pages as in the following examples.

- *Mixed PR* refers to the family of scores evaluated as $f(\mathrm{PR}, \mathrm{RPR})$, i.e., some function of the already computed PR and RPR values. Mixed PR is a trade-off between hub and authority scores depending on function f.
- *Product PR* score of each page is defined as the product of PR and RPR values. (Notice that product PR specializes mixed PR.) Web pages possessing high product PR are both valuable hubs and authorities, so the numbers of in-coming and outgoing paths are both large. We believe that such pages play an important role in maintaining the connectivity of the Web graph.

3 Centrality Analysis

For a given ranking of the Web pages, centrality-analysis experiments numerically evaluate the centralities of small sets of top-ranked pages in the Web graph. Such an experiment requires graph theoretical definition of centrality; in the following section we propose different notions of centrality based on averaging some distances in the Web graph.

Distance averaging techniques face the problem of infinite distances that is handled by harmonic mean in our definitions. A further advantage of harmonic mean that it expresses the expected search efficiency of a surfer following the shortest paths of the Web.

3.1 Domination of a Start Set

From a general start set of pages most other nodes of the Web graph should be available within a few clicks. We introduce a qualification for start sets and an intuitive explanation of the formula through search efficiency.

Suppose that a user is searching for some target page. Let us assume that by carefully reading the contents of the intermediate pages, it is always possible to choose the best possible direction towards the target. In this case the surfer will follow a shortest path.

Next we consider how efficiently the user spent browsing time to find the target. If the target is reached in 3 clicks for example, then he spends one third of his time to read something interesting while the rest of it is wasted for visiting inner pages of the search path. Hence we say that the *efficiency of a start page* s to find target t is $\frac{1}{\mathrm{dist}(s,t)}$, where $\mathrm{dist}(s,t)$ denotes the minimum number of clicks to reach t from s. If there is no path from s to t, then $\mathrm{dist}(s,t) = \infty$ and the efficiency is zero.

More generally, the surfer uses some start set V_S of pages to find target t. As he always starts from the members of V_S, he knows well the contents of these pages. Therefore he can guess the closest page of V_S to t. Then the *efficiency of the start set* is $\frac{1}{\mathrm{dist}(V_S,t)}$, where $\mathrm{dist}(V_S,t)$ denotes the minimum of distances from the nodes of V_S to t. The *domination of a start set* is defined as the average efficiency over all possible web pages as goals. This can be interpreted as the expected efficiency, if a surfer starts searching a random goal page. Formally, the domination of V_S is determined as follows:

$$\mathrm{dom}(V_S) = \frac{1}{|V| - |V_S|} \sum_{t \in V \setminus V_S} \frac{1}{\mathrm{dist}(V_S, t)},$$

where V denotes the set of Web pages. Thus the domination of a start set is the inverse of the harmonic mean of distances between V_S and all the other Web sites.

Our first notion of centrality of a set of pages is equal to the above introduced domination. In the centrality analysis experiments of Section 4.2 we successively add the top ranked pages to a start set and evaluate the domination in each iteration. The experiment reveals the quality of the ranking algorithm to select graph theoretically good sets of hubs or starting points from which the rest of the Web is accessible within a few clicks on the average.

Our notion of domination resembles of the NP-hard combinatorial optimization problem of finding a minimum size subset of nodes in a graph G such that all the other nodes are within a given distance k from the subset [11]. In our scenario such a subset would be a start set from which the farthest node has distance at most k. Such a worst-case analysis cannot express a fine quality measure on the start set, hence we proposed to take the average of distances.

3.2 Attacking the Web

Besides domination, the centrality of a set of nodes can be measured by the *attacking ability* of the set—the decay in the connectivity of the Web graph after removing the set of nodes in question. In our centrality analysis experiments, the top ranked nodes are removed gradually, and then we evaluate the connectivity of the remaining part of the Web graph.

The connectivity is expressed by the *harmonic diameter* of the Web graph, the harmonic mean of distances between all the pairs of nodes. The reciprocal of the harmonic diameter, under the notion of the previous subsection, means the expected efficiency when a surfer starts searching a random goal from a random start node. Hence what we actually measure is the fraction of time spent on reading topics of interest in contrast to downloading pages just to find an appropriate link to move on. Formally if V denotes the set of Web pages, then let

$$\mathrm{diam} = \frac{|V|(|V| - 1)}{\sum_{u \neq v \in V} \frac{1}{\mathrm{dist}(u,v)}}.$$

Another advantage of our notion of harmonic diameter compared to other notions of diameter is that pairs of nodes unreachable from one another have contribution zero in the formula, hence harmonic distance measures both distance and reachable at the same time.

The idea of removing some small portion of Web pages and measuring how the diameter increases was originally proposed [1] for different purpose. They concluded that the failure caused by randomly chosen nodes hardly effect the connectivity of the Web, but an intentional attack removing the nodes with large degree raises the average distance rapidly. Notice that the degrees of nodes also induce a ranking on the nodes. In our experiments we investigate the effect of replacing degree rank with more subtle scores of the importance of pages.

4 Experimental Results

Our experiments were conducted on the `.ie` domain, the Web pages of Ireland. We believe that the structure and diversity of this domain is similar to that of the whole WWW. The graph of the `.ie` domain was small enough to store in internal memory, thus any variant of the proposed ranking algorithms were calculated within 15 minutes.

We downloaded 986,207 pages from the Irish Web in October, 2002. We used the open source Web robot Larbin [16] on a 1.8GHz Pentium IV CPU with a 10Mb Ethernet connection. The Web graph induced by the `.ie` domain had 792,902 nodes[3] and 10,037,951 edges. The ranks PR, RPR, popular RPR and product PR were computed with damping factor $d = 0.2$ using 100 power iterations that yielding an error smaller than 10^{-8} in all cases.

4.1 Ranking Keyword Search Hits

We investigate how well RPR or popular RPR serve in ranking keyword queries. We believe that by the nature of ranking link collections high our ranking strategies act well for a broad topic search—at least as a possible aggregated rank component combined with text and link based strategies. In our experiment we submitted keywords of broad topics to Google [12] and saved all the enumerated URLs. The number of available URLs was varying between 500 and 1000. Then we used RPR to reorder these URLs and compared the top ten Google hits with our ranking. Since the reordered list was computed from Google's top $500 - 1000$ hits, this can be treated as an aggregate of Google's ranking with popular RPR.

The query results are listed on Table 1 for "fishing" and "sailing"—typical broad topic query strings for exploring certain topic rather than searching for a specific piece of information. The number 1, 4 and 5 hits of Google on "fishing" are Web sites of specific famous fishing resorts and boats—inevitably these pages provide popular content. Popularity is however not appreciated by the RPR scores; instead credit is given to link collections. Such examples are the number 2, 4, 5, 7 and 8 hits of RPR query or 1, 2, 3 and 4 of popular RPR for "fishing". Hit number 8 of popular RPR on "sailing" is a remarkable example of a good link collection. Such a collection may act as an excellent start node to explore "sailing in Ireland".

A drawback of RPR and popular RPR can be also read from the lists of top ranked URLs. Both gives high credit to archives or large collections of databases within a Web site. Such examples are 1 and 3 from RPR with query "fishing". In some cases popular RPR was able to overcome the problem such as in the case of "fishing" query, since the members of the archive have low target probability.

4.2 Top Ranked Pages, Domination, and Diameter

In our centrality analysis experiments we selected the first few top ranked pages under different ranks and measured graph theoretic quantities related to distance and connectivity as a function of the number of pages selected. We graphed our results for multiples

[3] We have deleted those pages not linking within the `.ie` domain that would otherwise correspond to a node with zero out-degree in the graph.

Table 1. Query results for Google and by reordering the top 500-1000 hits of Google.

Google with query "fishing"	Google with query "sailing"
1 indigo.ie/~bwlodge/	1 www.sailing.ie/
2 indigo.ie/~bwlodge/fisreport.htm	2 www.iol.ie/ glenans/
3 www.infowing.ie/fishing/	3 www.iol.ie/ gerbyrne/
4 www.infowing.ie/fishing/Sligo2.htm	4 www.braysailingclub.ie/
5 homepage.tinet.ie/~bluewater/	5 www.braysailingclub.ie/sailing/ sailing_instructions.html
6 homepage.tinet.ie/~ncffi/	6 www.alia.ie/sailing/
7 www.shannon-fishery-board.ie/	7 www.alia.ie/sailing/afloat.html
8 www.shannon-fishery-board.ie/ fishing-open.htm	8 www.arklowsc.ie/
9 www.react.ie/Activities/Fishing.htm	9 www.arklowsc.ie/Sailing_Tips/ sailing_tips.htm
10 www.react.ie/Activities/ Fishingwhere.htm	10 homepage.tinet.ie/ bmcg/Cullaun/ cullaun.htm
RPR with query "fishing"	**RPR with query "sailing"**
1 www.ndpgenderequality.ie/statdata/ 2002/measure/measure4.html	1 sport.startpage.ie
2 www.nci.ie/holiday	2 www.irishferries.ie/sitemap.shtml
3 www.ndpgenderequality.ie/statdata/ 2002/topic/topics17.html	3 www.homefromhome.ie/properties.asp
4 kildare.local.ie/things_to_do_and_see	4 www.kellyco.ie/html/AvailRes.html
5 www.lakedistrict.ie/fishing/index.shtml	5 www.athlonechamber.ie/about-athlone/ tourism.htm
6 www.thecia.ie/patricks	6 www.oksports.ie/irish/water.html
7 westmeath.local.ie/things_to_do_and_see	7 www.wolfhound.ie/eveningclasses/ email.htm
8 www.oksports.ie/irish/water.html	8 doon.mayo-ireland.ie/moores.html
9 www.falconholidays.ie/locations/ 12/11.html	9 www.inside.ie/e_article000074755.cfm
10 www.cybercottage.ie	10 www.csis.ul.ie/staff/CiaranCasey/ personal.htm
Popular RPR with query "fishing"	**Popular RPR with query "sailing"**
1 www.nci.ie/holiday	1 www.irishferries.ie/sitemap.shtml
2 kildare.local.ie/things_to_do_and_see	2 sport.startpage.ie
3 www.infowing.ie/fishing	3 www.kellyco.ie/html/AvailRes.html
4 www.lakedistrict.ie/fishing/index.shtml	4 www.homefromhome.ie/properties.asp
5 www.connacommunitycouncil.ie	5 www.athlonechamber.ie/about-athlone/ tourism.htm
6 westmeath.local.ie/things_to_do_and_see	6 www.wolfhound.ie/eveningclasses/ email.htm
7 www.thecia.ie/patricks	7 www.rte.ie/aertel/p581.htm
8 tiara.ie/goingto.htm	8 www.oksports.ie/irish/water.html
9 indigo.ie/~bwlodge/fisreport.htm	9 www.tourismresources.ie/fh/ shannon.htm
10 www.cybercottage.ie	10 www.rosscarbery.ie

of a hundred Web pages. Under any reasonable ranking strategy the top few hundred nodes should form a subset of the Web with an important role in search and navigation. Note that the size in question is smaller than one percent of our document collection of pages from Ireland.

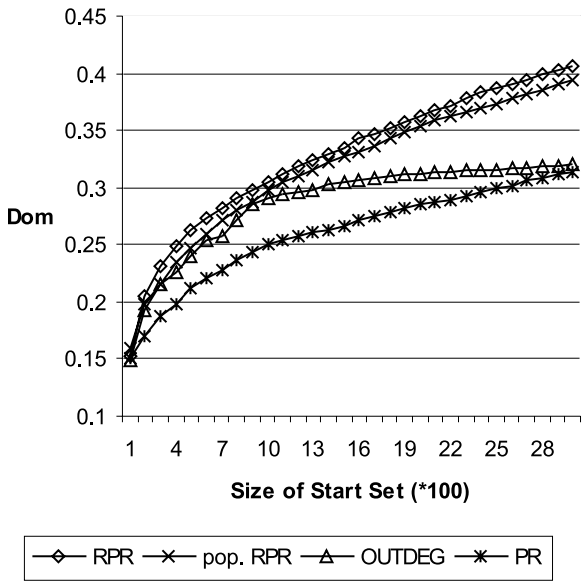


Fig. 1. Domination of start sets.

In our first experiment we constructed start sets of sizes falling into the range $100 - 3000$ from the top ranked nodes. Then we calculated the domination of these sets, the efficiency of searching a random node from the given set. The results for PR, RPR, popular RPR and out-degree rank are depicted on Fig. 1.

The diagram shows that nodes with large PR behave worse as start sets than even the simple heuristic of choosing out-degree as rank. On the other hand both RPR and popular RPR finds sets with large domination, i.e., from these sets all the other pages are accessible within a few clicks on the average. Recall that RPR scores are based on counting the weighted sum of all search paths as stated in Theorem 1. The domination of top ranked sets are calculated on the basis of shortest paths, thus we conclude from the success of RPR scores that RPR acts as some approximation of shortest path counting. We mention that such approximation results do not hold in arbitrary graphs, since in RPR all the search paths are taken into account not only the shortest paths.

The removal of the top ranked nodes should, in addition to having large domination, also destroy the connectivity of the Web. While removing the top ranked 100, 200, ..., 1000 nodes, we measured the harmonic diameter of the remaining graph.[4] The results are depicted on Fig. 2 for PR, RPR, popular RPR, degree rank, and the mixed rank computed as the product of PR and popular RPR.

[4] An exact computation of the diameter would require a Depth First Search from each node. Thus we approximated the result by computing DFS from 1000 randomly chosen nodes.

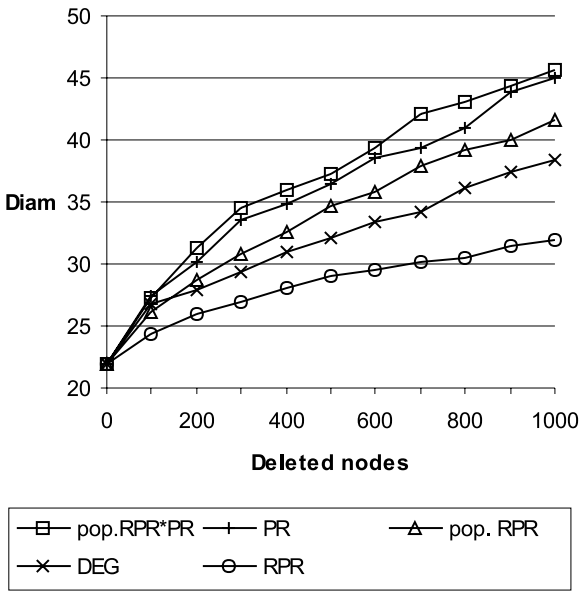


Fig. 2. Increasing the diameter of the Web graph by removing the top ranked nodes.

Product PR turns out the strongest "destructor" by increasing the diameter over 45 after removing 1000 nodes. The reason for this phenomenon is that product PR can only be high for a node having both high RPR and PR scores. High RPR scores imply that a large number of search paths depart from the page, and the PR score shows that a large amount of search paths arrive at the node in question. Thus, a node with high product PR is a typical inner node of short search paths of the Web. Therefore the removal of such central nodes destroys the connectivity of the Web as verified by our experimental results.

The fact that RPR has the lowest power of destruction among the measures appears surprising and contradicting the domination results. However it is easy to put the two results together and conclude that top start rank nodes, instead of acting central and interconnecting different topics and domains, serve for finding quick routes by possibly sitting on the top of large semi-local collections of specific and non-overlapping topics.

Except for RPR, all the ranking algorithms performed better than the degree rank, thus we strengthen the results of [1]. PR, product PR and popular RPR all provide central sets of nodes taking the responsibility for the low diameter of the Web graph. The existence of such centralized sets let us a deeper insight how the small world property is achieved for the Web graph.

5 Conclusion

Start nodes play important roles in exploring some part of the Web. We proposed start rank algorithms to express the qualities of pages as hubs based on short random walk arrival probabilities. The algorithm performs Page Rank computation on the reversed Web Graph. Thus, it is practically implementable in case of the Web graph. Graph theoretical tools are introduced to evaluate start ranking algorithms by measuring the domination and the attacking ability of the top ranked nodes. In our experiments on the Irish Web, the proposed start ranking algorithms selected start sets with largest domination justifying our intuitions. We believe that aggregating the start rank algorithms in text based query search engines improves the efficiency of browsing the Web.

6 Acknowledgment

I wish to thank Katalin Friedl, András Benczúr and András Lőrincz for the valuable discussions and for improving the level of this manuscript.

References

1. R. Albert, H. Jeong, and A. Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.
2. B. Amento, L. Terveen, and W. Hill. Does authority mean quality? Predicting expert quality ratings of web documents. In *Proceedings of the Twenty-Third Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2000.
3. Y. Azar, A. Fiat, A. R. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *ACM Symposium on Theory of Computing*, pages 619–626, 2001.
4. A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the word-wide web. *Physica A*, 281:69–77, 2000.
5. A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *10th International World Wide Web Conference*, pages 415–429, 2001.
6. J. Boyan, D. Freitag, and T. Joachims. A machine learning architecture for optimizing web search engines. In *Proceedings of the AAAI Workshop on Internet-Based Information Systems*, 1996.
7. S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.
8. S. Chakrabarti, B. E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the Web's link structure. *Computer*, 32(8):60–67, 1999.
9. B. D. Davison, A. Gerasoulis, K. Kleisouris, Y. Lu, H. ju Seo, W. Wang, and B. Wu. Discoweb: Applying link analysis to web search. In *Proceedings of the 8th World Wide Web Conference, Toronto, Canada*, 1999.
10. C. Dwork, S. R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *10th International World Wide Web Conference*, pages 613–622, Hong Kong, 2001.
11. M. Garey and D. Johnson. *Computer and Intractability : A Guide to the Theory of NP-completeness*. W.H. Freeman, San Fransisco, 1979.
12. Google. Commercial search engine founded by the originators of pagerank. located at `http://www.google.com`.

13. T. H. Haveliwala. Topic-sensitive pagerank. In *11th International World Wide Web Conference*, Honolulu, Hawaii, 2002.
14. L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, March 1953.
15. J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
16. Larbin. Multi-purpose web crawler.
17. R. Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. In *9th International World Wide Web Conference*, 2000.
18. M. Marchiori. The quest for correct information on the web: Hyper search engines. In *7th International World Wide Web Conference*, 1998.
19. A. Y. Ng, A. X. Zheng, and M. Jordan. Stable algorithms for link analysis. In *Proc. 24th Annual Intl. ACM SIGIR Conference*, 2001.
20. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.
21. M. Richardson and P. Domingos. The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

Small-World Networks Revisited

Thomas Fuhrmann

Institut für Telematik, Universität Karlsruhe (TH), Germany

Abstract. Small-world networks have received much attention recently. Computer scientists, theoretical physicists, mathematicians, and others use them as basis for their studies. At least partly due to the different mind-sets of these disciplines, these random graph models have not always been correctly applied to questions in, e.g., peer-to-peer computing. This paper tries to shed some light on common misunderstandings in the study of small-world peer-to-peer networks. It shows that, contrary to some recent publications, Gnutella can indeed be described by a model with power-law degree distribution. To further distinguish the proposed model from other random graph models, this paper also applies two mathematical concepts, dimension and curvature, to the study of random graphs. These concepts help to understand the distribution of node distances in small-world networks. It thus becomes clear that the observed deficit in the number of reachable nodes in Gnutella-like networks is quite natural and no sign of any wrong or undesirable effect like, e.g., network partitioning.

Index terms — Random Graphs, Small World Networks, Peer-to-Peer Computing, Gnutella

1 Introduction

Peer-to-peer networks have received much attention over the recent years. Many authors have studied the properties of Gnutella and similar overlay-networks with the help of measurements, simulations, and mathematical analysis. It has, for example, been demonstrated that Gnutella forms a *small-world-network* which has interesting properties that go beyond the *random graphs* originally studied by Erdös and Renyi. However, much confusion has arisen in the computer networks community about the nature of such networks and the consequences resulting from their special properties. These misunderstandings might, at least in part, be due to the fact that the characteristics of small-world-networks were first studied by the statistical physics community. Many properties that were considered to be obvious there, were omitted or stated only implicitly in their publications. Other properties, like the distance distribution of such networks, that are especially important for computer networks were not in the focus of the statistical physics community.

As a consequence, the construction of peer-to-peer systems was not always based on correct assumptions. Sometimes, the resulting contradictions became

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 80–92, 2003.

obvious with measurements, but not always the true nature of the conflict has been revealed. Therefore, incorrectnesses still flaw the peer-to-peer networking literature.

This paper tries to shed some light on major conflicts in the understanding of small-world peer-to-peer networks, i.e., Gnutella-like networks. To this end, section 2 gives a short overview of different models for random graphs. Section 3 then introduces the concept of dimension and curvature into the study of random graphs. Both concepts describe the number of network nodes reachable in a given number of hops. Especially, the concept of curvature naturally explains recent measurements that were incorrectly interpreted as effects of disconnected subgraphs.

Section 4 discusses a typical small-world network model, the Barabasi-Albert model, in its application to Gnutella-like overlay-networks. It is shown that this model fails to reflect the structure of Gnutella. Hence, an alternative model is proposed that better describes recent measurements of the degree distribution of Gnutella. Section 5 supplements this analysis with a study of the distance distribution resulting from these models. Using the concept of dimension and curvature this extended analysis shows that the structure of Gnutella-like networks might be best understood as a sphere with fractional dimension. Section 6 finally concludes with an outlook on future work.

2 Random Graphs

Random graphs were introduced by Erdös and Renyi [6, 7]. They studied the probability space of graphs with a constant number of vertices and edges, and of graphs into which edges are introduced with a constant probability. Watts and Strogatz, on the other hand, studied the process of randomly rewiring a regular graph [16]. Barabasi and Albert, introduced yet another model. They studied graphs that are built up gradually by adding new vertices and edges so that the probability of an existing vertex gaining one of the new edges is proportional to its degree [4, 1].

Although, in the literal sense all of these models are *random graphs*, namely probability spaces built from a set of graphs, often only the Erdös-Renyi models are called random graphs, while the Watts-Strogatz and the Barabasi-Albert type graphs are often termed *small-world networks*, in allusion to the work by Milgram [10] who studied social networks. The idea behind this term is the comparatively small average path-length between two arbitrary nodes in such networks. To correct a common misunderstanding, Erdös-Renyi random graphs, too, show the same small-world characteristic. So, small-world networks are small when compared to regular graphs, not when compared to Erdös-Renyi random graphs.

The Barabasi-Albert model is of special interest for the study of computer networks, especially of peer-to-peer networks because the process of adding nodes to an existing network models the growth of many such networks: [3] gives the argument that a useful web-site or one that is en vogue is referenced more of-

ten than an uninteresting page. The same argument was put forth for the autonomous systems of the Internet [8, 5], and the Gnutella network [12]. There, nodes linked to many other nodes spread the knowledge about their existence in the Gnutella overlay-network more efficiently and have thus a higher probability that newly connecting servants connect to them. This mechanism is more closely studied in section4 where yet another model is proposed that better reflects the measured properties of the Gnutella network.

3 Topology, Curvature, and Dimension

If mathematical strictness is omitted, *topology* can be be said to describe the structure of a set without requiring a metric. To this end, the concept of a *neighborhood* is used. Typically, one would think of infinite sets, although one can construct topologies for finite sets, too. The fact that the study of computer networks employs topology mainly in connection with graphs, more precisely, finite graphs, is hence misleading but not wrong. [1] The mathematical discipline named topology was however created by HAUSDORFF who introduced its classical concepts with the study of continuous functions, i.e. with infinite sets. It is helpful to be aware of these nuances when one studies random graphs and similar structures.

If a set is equipped with a *metric*, a neighborhood of an element x can be defined as the subset whose elements have less than a given distance from x. Varying this distance and varying x then yields the neighborhoods required to define a topology. In other words, a metric can directly induce a topology.

In computer networks, various properties can be used to define a metric, e.g., hop-count and transmission delay. While a computer scientist might think of a metric being such a concrete property, a mathematician thinks of a metric as a mere mapping that assigns distances to pairs of points with no other implied meaning. In order to qualify as a metric in the mathematical sense, such a mapping needs to be symmetric in its arguments and it needs to satisfy the triangle inequality. Luckily, both approaches coincide when a network has bidirectional links and employs shortest path routing.

In theoretical physics, the term *metric* is closely linked to the study of so-called *manifolds*, a mathematical structure that generalizes the concept of a vector space. A smooth surface, e.g., of a sphere or a torus, gives a good intuition of a two-dimensional manifold. As has been said above, the topology of a manifold can be derived from its metric. In addition to its topology a (differentiable) manifold is also equipped with *curvature* and *torsion*. Both, too, can be derived from the metric. With a fair amount of simplification one can say that the curvature is determined by the excess or deficit content of an infinitesimal piece of surface. Imagine, e.g., a piece of paper that was soaked with a spill of water and now starts to bend because the fibers extend locally at the soaked

[1] Books on the history of mathematics, e.g., associate this topic with EULER who studied the problem of the "Königsberger Brücken", a question linked to topology by the study of homotopic paths.

spot. Similarly, a circle with a given radius encloses a smaller amount of the surface if it is drawn on a sphere than on a flat piece of paper. Conversely, the enclosed surface is larger for a circle drawn on a saddle, i.e. a surface of negative curvature.

The relation to computer networks becomes immediately clear when one considers the number of network nodes that can be reached from a given node within a certain number of hops. This is the natural analogon of surface content and typical property studied with random graphs and small-world networks. Recently, some confusion arose from the question why the number of nodes reachable by flooding with a given time-to-live (TTL) did not increase with the TTL to the extend that was naively expected. It was even speculated that the missing nodes got somehow isolated from the network [13, 14]. The concept of curvature, however, demonstrates that deficits in the size of reachable areas are quite natural. There is no hidden land on earth. The earth's surface is smaller than naively calculated from a flat map because the earth is a sphere and not a flat disk. The very same argument applies to the size of networks, like, e.g., Gnutella.

Before analyzing the size and structure of such random graphs in detail, another fundamental mathematical concept needs to be introduced, the *dimension* of a set. It, too, deals with the rate of increase in reachable areas when the maximal traversed distance is increased. Without mathematical strictness, a set can be defined to have dimension d if the size of the area reachable within a distance r increases (for $r \to 0$) proportional to r^d. E.g., the area of a disk in a flat two-dimensional manifold increases with πr^2. On the unit sphere, it increases with $4\pi \sin^2 \frac{r}{2}$ which, for $r \to 0$ is again πr^2. So, both cases yield expectedly $d = 2$. [9] gives many illustrative examples for fractal sets, i.e. sets with non-integer dimension.

Summarizing this short overview, one can say that a topology describes a set without reference to a metric. With a metric, a set can be described further: On small scales, the rate of increase of the reachable parts of a set is measured by its dimension. On larger scales, the excess or deficit in the size of the reachable parts is described by the curvature. Of course, this summary is rough and lacks mathematical strictness. But it can serve as motivation for the terminology used in the following sections.

4 Modeling Gnutella-like Networks

Recently, the great interest in models for peer-to-peer networks, especially for Gnutella-like networks, and the discovery that many real-life networks can be described by random-graph models with power-law degree distribution has lead to the conviction that Gnutella-like networks can be modeled by such graphs, too. On the other hand, recent measurements seem to contradict this assumption [12, 11].

The simulation experiments described below show that both conclusions are only partly correct. They indicate that the measured properties of Gnutella

can in fact be described by a model with a power-law degree distribution. But the required model differs from the Barabasi-Albert model of the world-wide-web link graph. In this section the degree distribution of the two model types is analyzed. In the following section the difference between the two models is further studied with the notion of dimension and curvature illustrated above.

The Barabasi-Albert model builds up a random graph by gradually adding new vertices and edges such that the probability of an existing vertex gaining one of the new edges is proportional to its degree. A mathematical analysis of this model [2] predicts a degree distribution of the resulting network that follows a power law $P(degree = x) \propto x^{\alpha}$ with $\alpha = -3$. This prediction is in good accordance with simulations (see [2] and figure 1).

Unlike this model, with Gnutella, servants create more than one initial link to the Gnutella overlay-network. This behavior can be simulated by a modified Barabasi-Albert model where new nodes create more than one initial connection. Figure 1 shows that this modification does not change the power-law structure of the resulting graph.

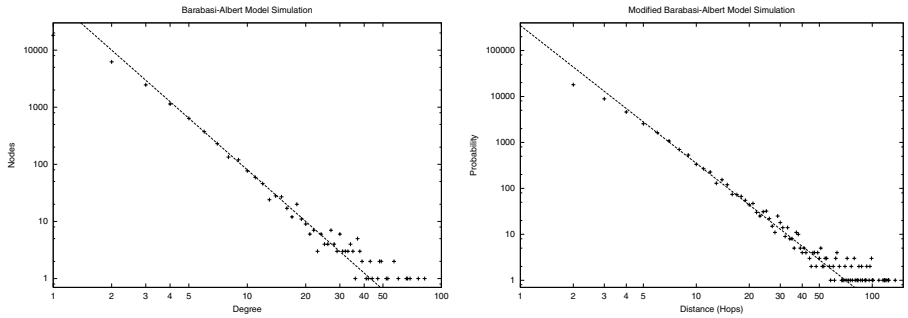


Fig. 1. Degree distribution from a 40 000 nodes simulation of the Barabasi-Albert model (left) and the modified Barabasi-Albert model (right)

Recent measurements, however, found that the degree distribution in Gnutella does not obey a pure power-law as would follow from the Barabasi-Albert model [12, 11]. Even more, unlike the world-wide-web, Gnutella can be assumed to respect some amount of locality in its network since the targets for new links for a node are found via the *ping-pong* algorithm that seeks for nodes in a TTL-limited neighborhood of the respective node.

We hence propose the following model to describe the behavior of Gnutella and similar networks. This model also incorporates the recent measurement results for the *ping-pong* mechanism and the uptime of Gnutella servants [15]:

> The network model consists of two types of nodes: Persistent nodes maintain long-lived connections to the Gnutella overlay network. Non-persistent nodes, e.g., servants with dial-up connections to the Internet,

> connect and disconnect rather frequently. (Note that in typical dial-up networks, reconnecting servants obtain a new IP address and thus appear as new nodes in the network.)
> Upon connection, each node creates one link to some node of the network. This node is chosen as a *3-neighbor* of a randomly selected node. The term *3-neighbor* describes a node that is at most three hops away from the randomly selected node.

In Gnutella neighbors are discovered by the ping-pong mechanism. For the model presented here, all the details of this mechanisms are neglected. Only the principal mechanism of neighbor-mediated link creation is maintained. This is an important difference to the Barabasi-Albert model that does not respect any locality in the creation of new links and that is hence *not* suited to model Gnutella-like networks. This difference is most important for the following refinement step that models the connectivity strengthening of a ripening Gnutella network:

> While non-persistent nodes are assumed to create all links upon initial connection to the network, persistent nodes remain connected long enough to create additional links to other nodes. However, the rate of this connection growth will typically be assumed to be small.

Figure 2 shows results from a simulation run for this model that contained 40 000 nodes of which 15 000 were assumed to be persistent. These nodes gained at least 5 additional links each. The total number of links per node was limited to at most 100.

In the plot in figure 2 one can clearly distinguish two components: For small degrees (one to three links per node) the network exhibits a power-law for the degree distribution. This is in accordance to the expectations for small-world networks. At higher degrees the distribution rises again and reaches a local maximum at a degree of about 10. Beyond that maximum the degree distribution again follows the expected power-law. Both power-laws have the same parameter, here, $\alpha = 3.8$.

This pattern found by the simulation of the proposed model reflects the structure found in measurements of the Gnutella network [12, 11]. In these publications, however, the conclusion was drawn that a ripened Gnutella network did not follow a power-law for the degree distribution. With the model described above it now becomes clear, that this conclusion is not correct and that Gnutella-like network can in fact effortlessly be described by a two-regime structure of persistent and non-persistent nodes, both of which exhibit a power-law for the degree distribution. The exact shape of the degree distribution is governed by the ratio of persistent and non-persistent nodes.

The following section will analyze further properties of this model. This will demonstrate that the simple notion of Gnutella-like networks being small-world networks with power-law degree distribution does not suffice to describe the properties of such networks.

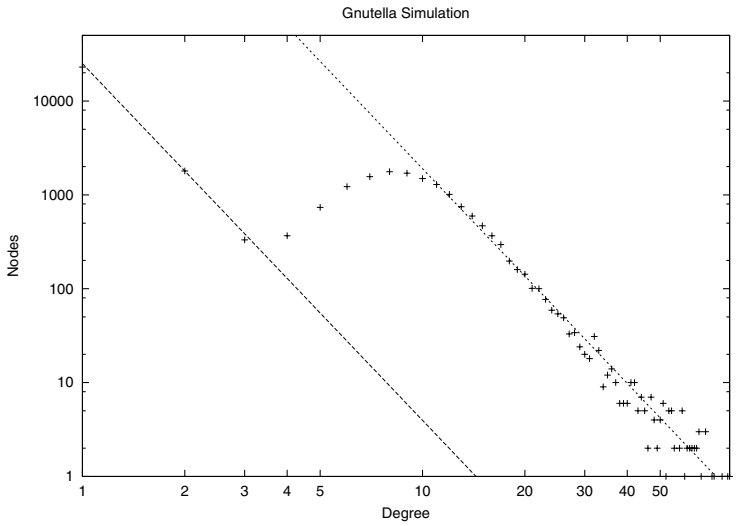


Fig. 2. Degree distribution from the simulation of the model for a Gnutella-like random graph (40 000 nodes of which 15 000 are persistent)

5 Curvature and Dimension of Random Graphs

So far, the discussion has only considered the degree distribution of the two network models. The term *small-world* network, however, addresses the question of distance, measured as hop count. It seems that both issues are not always clearly distinguished in the literature. Therefore, the following discussion will address the question of the *dimension* of a graph. This is the natural analogon of the concept of a dimension of a fractal set. It, too, obeys a power law. But this power law must not be confused with the power law of the degree distribution discussed in the previous section.

As described above, the dimension describes — on small scales — the increase in the number of reachable nodes when the time-to-live is increased. In order to sensibly assign a dimension to a graph this increase must follow a power law, $n \propto r^d$, where d is the dimension of the graph. Hence, of course, not all graphs have a dimension, e.g., a tree cannot be assigned a dimension.

Following the analogy from section 3, on larger scale, the increase in reachable nodes can be described as governed by the curvature of the graph. Again, not for all graphs this definition is mathematically sensibly possible. Even more, for a graph model that does not single out certain regions of the graph, the curvature must be expected to be constant, so that the graph needs to described as a *sphere*. (Mathematically, a surface of constant positive curvature is a sphere.) From this follows that the number of nodes reachable within time-to-live r is proportional to $sin(\frac{r}{2})^d$. Curvature, more exactly, constant curvature, is thus another important criteria to distinguish graphs.

In the following, we present the simulation results for the Barabasi-Albert model, the modified Barabasi-Albert model, and our Gnutella model. They will show that the latter model indeed leads to a sphere structure with fractional dimension. The two other models can still be assigned a dimension, but they fail to be describable by a constant curvature. Since both concepts, dimension and curvature, lead to important topological consequences for a network, this analysis is a valuable tool for the study of network models.

5.1 Gnutella-like Graphs

Figure 3 shows the comparision of the described simulation with the theoretical prediction from a graph with fractal dimension and constant curvature. Both viewgraphs show the same simulation of 40 000 nodes presented above.

Since for practical purposes, it is convenient to measure the rate of increase in the number of reachable nodes, the graphs do *not* show cumulated node counts, but the number of nodes that have exactly the given distance from a randomly selected node.

The upper graph demonstrates that, up to about 8 to 10 hops, the distribution is described by a power-law. The fitted line's slope directly yields the dimension of the graph, namely 4.8. The lower graph shows that the simulation outcome is also well described by the assumption of a constant curvature. The fit corresponds to a quadrant length of 11.9 hops, where quadrant length means half the maximum distance within the graph. (In graph theory the maximum distance within the graph is called *diameter*. In combination with the sphere analogon this term is misleading since the geometrical and graph theoretical diameter differ by a factor of $\frac{\pi}{2}$.)

Both, dimension and quadrant length, depend on the parameters of the simulation:

Persistent nodes	Total nodes	Dimension	Quadrant length
25000	80000	5.1	12.6
25000	60000	5.1	12.2
25000	40000	4.8	12.2
15000	40000	4.8	11.9
5000	40000	4.8	11.9
5000	25000	4.4	11.2
5000	10000	4.1	10.2

The nature of this dependence must be explored by further studies beyond the scope of this paper.

5.2 Barabasi-Albert Graphs

Doing the same analysis for the Barabasi-Albert models yields a different result. Figure 4 shows a simulation of 40 000 nodes. The graph has a dimension of 4.55, but there is a significant deviation from the constant curvature model. Up to

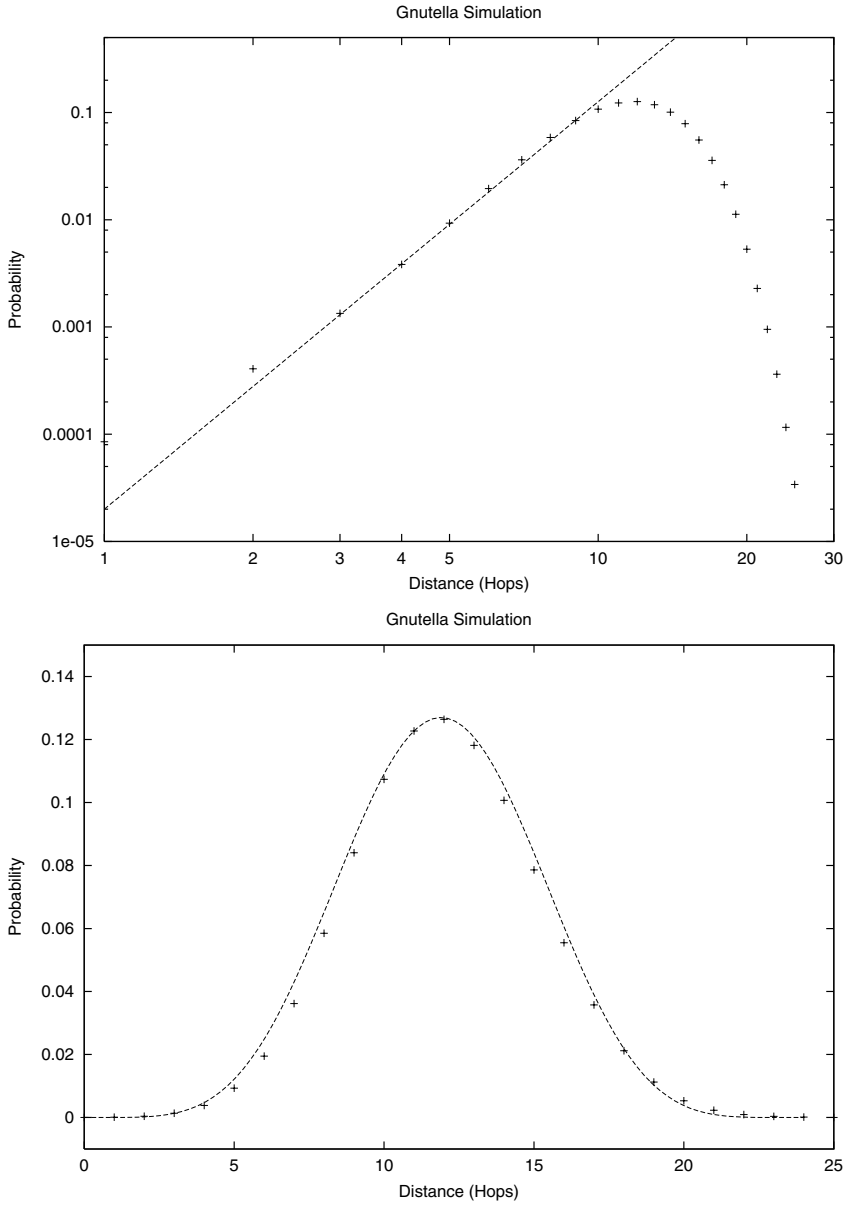


Fig. 3. Simulation of a Gnutella-like random graph: Probability distribution for distances between pairs of nodes as compared to the theoretical prediction. (Both graphs show the same data, see text.)

about 12 hops the model is still in good accordance with a constant curvature with quadrant length 11.0. But there is a significant higher fraction of nodes with a distance of 15 and more hops than could be expected based on the constant curvature model.

With the modified Barabasi-Albert model, the constant curvature assumption fails completely (cf. fig. 5). The resulting graph has a dimension of 7.0, but the distance distribution falls off rapidly at about 6 hops instead of showing a quadrant length of 7.3 as is indicated at small distances.

This analysis shows that the structure of an overlay-network has important consequences for the node distribution. If a network is known to have a sphere structure, the node distribution can be well predicted. This knowledge can then be used to choose the protocol parameters for optimized performance. If that knowledge lacks or, even worse, the wrong model is chosen, performance is greatly degraded. As can be seen from the latter example, small-world networks can be easily misjudged with regard to the expected node distance. As a consequence, the network is either unnecessarily flooded (distances overestimated) or, e.g., a search fails because too few nodes receive a message (distance underestimated).

6 Conclusion and Outlook

This paper has illustrated how the mathematical concepts of dimension and curvature can be applied to the study of random graphs and especially to small-world networks. These concepts yield a simple model for the distribution of the node-to-node distances found in random graphs. This property is important for the understanding of many peer-to-peer networks, especially Gnutella-like networks.

It was shown that these two concepts together with a newly proposed model for Gnutella-like networks lead to the picture of *Gnutella as a graph with fractal dimension and constant curvature*. The fact that the model's degree distribution well reflects the measurements by Ripeanu indicates that this model might actually describe the properties of the Gnutella network better than established models. However, before this conclusion can be drawn, further measurements of the dimension and curvature, as defined in this paper, are needed to confirm the simulation results presented in this paper.

Besides such measurements, more analytical and simulation work is needed, too. The properties that were here only studied by means of simulations need to be analyzed mathematically, and the dependence between the model's parameters needs to be further explored. This is even more desirable since these first results already indicate that small-world networks and especially Gnutella-like networks have very interesting properties that go beyond the well-known power-law in their degree distribution.

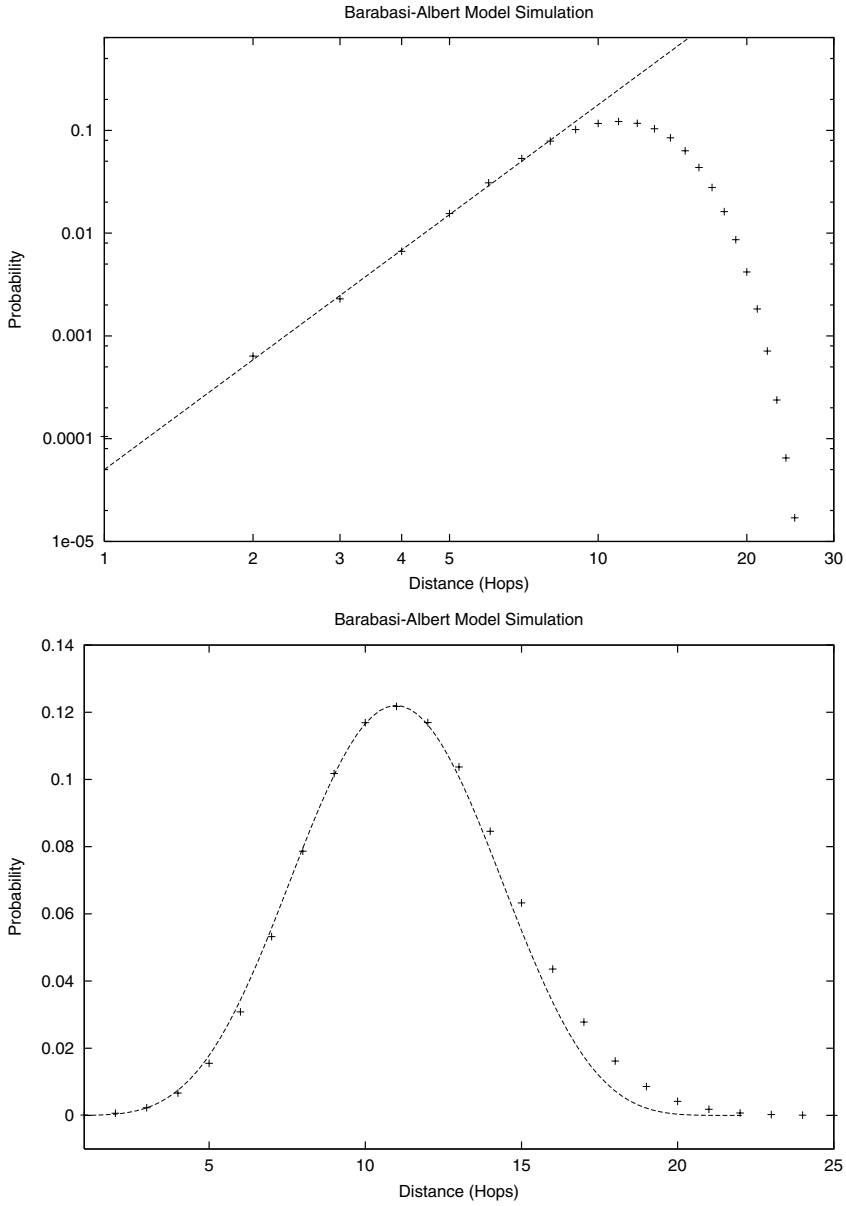


Fig. 4. Simulation of a Barabasi-Albert random graph: Probability distribution for distances between pairs of nodes as compared to the theoretical prediction. (Both graphs show the same data, see text.)

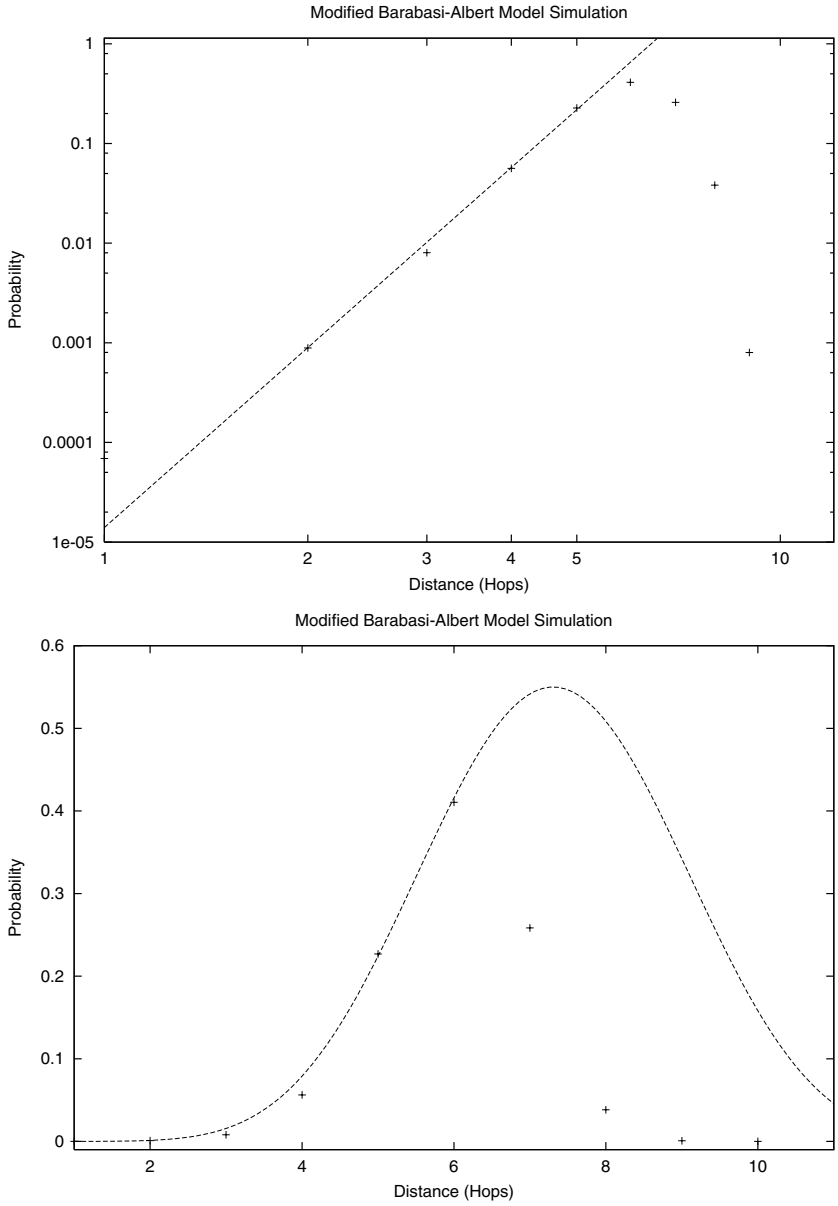


Fig. 5. Simulation of a modified Barabasi-Albert random graph with two initial links for each node (60 000 nodes). Clearly, the simulation outcome cannot be explained by the curvature assumption.

References

[1] Reka Albert and Albert-Laszlo Barabasi. Topology of evolving networks: Local events and universality. *Physical Review Letters*, 85(24), 2000.

[2] Reka Albert and Albert-Laszlo Barabasi. Statistical mechanics of complex networks. *Review of Modern Physics*, 74(47), 2002.

[3] Reka Albert, Hawoong Jeong, and Albert-Laszlo Barabasi. Diameter of the world wide web. *Nature*, 401:130–131, September 1999.

[4] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[5] Qian Chen, Hyunseok Chang, Ramesh Govindan, Sugih Jamin, Scott J. Shenker, and Walter Willinger. The origin of power laws in internet topologies revisited. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, volume 2, pages 608–617, 2002.

[6] Pal Erdös and Alfred Renyi. On random graphs. *Publ. Math. Debrecen*, 6:290–297, 1959.

[7] Pal Erdös and Alfred Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.*, 5:17–61, 1960.

[8] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999.

[9] Benoit B. Mandelbrot. *The Fractal Geometry of Nature*. H. W. Freeman and Company, New York, 1977.

[10] Stanley Milgram. The small world problem. *Psychol. Today*, 2:60–67, 1967.

[11] Matei Ripeanu and Ian Foster. Mapping gnutella network: Macroscopic properties of large-scale peer-to-peer systems. In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS'02)*, Cambridge, Massachusetts, March 2002.

[12] Matei Ripeanu, Ian Foster, and Adriana Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal (Special issue on peer-to-peer networking)*, 6(1), 2002.

[13] Rüdiger Schollmeier and Felix Hermann. Topology-analysis of pure peer-to-peer networks. In *Proceedings of Kommunikation in Verteilten Systemen*, pages 359–370, Leipzig, Germany, February 26-28 2003.

[14] Rüdiger Schollmeier and Gero Schollmeier. Why peer-to-peer does scale: An analysis of P2P traffic patterns. In *Proceedings of the Second International Conference on Peer-to-Peer Computing*, pages 112 –119, Linköping, Sweden, September 5-7 2002.

[15] Kurt Tutschku and Hermann deMeer. A measurement study on the dynamics of gnutella overlays. In *Proceedings of Kommunikation in Verteilten Systemen*, pages 295–306, Leipzig, Germany, February 26-28 2003.

[16] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.

Message-Oriented-Middleware in a Distributed Environment

Sushant Goel[1], Hema Sharda[1], and David Taniar[2]

[1] School of Electrical and Computer systems Engineering,
Royal Melbourne Institute of Technology, Australia
hema.sharda@rmit.edu.au
[2] School of Business Systems, Monash University, Australia
David.Taniar@infotech.monash.edu.au

Abstract. Middleware technologies have been facilitating the communication between the distributed applications. Traditional messaging systems are synchronous and have inherent weaknesses like limited client connections, poor performance due to lack of resource pooling, no store-and-forward mechanism or load balancing, lack of guaranteed messaging and security as well as static client and servers location dependent code. These weaknesses and increasing e-business requirements for the distributed systems motivated us to undertake this research. This paper proposes an asynchronous communication architecture Transfer of Messages in Distributed Systems. The advantage of the proposed architecture is that the sender of the message can continue processing after sending the message and need not wait for the reply from other application.

1 Introduction

Type of applications in the computing world has evolved rapidly from stand-alone architecture to mainframe architecture to two-tier client/server or three-tier (multi-tier) client/server architecture [6]. As the applications are becoming distributed, problems of information management on a large and distributed scale have become highly apparent.

This paper aims to enhance the features of messaging-architecture in distributed environment. Middleware is important in providing communication across heterogeneous platforms. Middleware technologies also plays important role in paradigm shift from mainframe to client/server architecture. Middleware is connectivity software that consists of a set of enabling services that allow multiple processes running on one or more machines to interact across a network [3].

Till now the industry had different commercial product for communication between components, but Java Message Service (JMS) is an effort towards standardization of the communication protocol. *JMS is an API provided by Sun Microsystems,* which is being supported by most of the messaging vendors. The *Transfer of Messages in Distributed Systems* (TMDS) architecture proposed in this paper, implements JMS specifications and enhances the features of existing

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 93-103, 2003.

middleware technologies. This is a step forward towards paradigm shift from synchronous to asynchronous messaging.

The basic aim of this paper to present a standard communication protocol in distributed environment and let the distributed components communicate in a more effective way. The proposed TMDS architecture puts together the benefits of synchronous and asynchronous communication (JMS). JMS was released in late 1998, and doesn't support XML, this paper enhances the feature of JMS and enable applications to communicate via XML message format.

The rest of this paper is organized as follows. Section 2 describes the background including messaging systems, middleware, JMS, and domain of messaging. Section 3 presents our proposed architecture. Section 4 presents a case study using our proposed architecture. Finally, Section 5 gives the conclusions and explains future work.

2 Background

We discuss various concepts of Message Oriented Middleware (MOM) in this section, including various domains of messaging like publish/subscribe and point-to-point messaging. Message is the package of business data, which contains the actual load and all the necessary routing information to travel in the network for delivery. Till late 90's there were couple of compan ies, who used to provide asynchronous communications between distributed components, like IBM's MQ Series.

2.1 Messaging and Message-Oriented-Middleware

Messaging is a peer-to-peer communication between software applications [5]. All the messaging clients are connected via an external agent and can send messages to any other client as well as can receive messages from any other messaging client [2]. The agent provides the way to communicate between the clients; it provides the facilities for creating, sending, receiving and reading the messages. Distributed applications can communicate in two ways:

In *synchronous communication* the application sends any message to another application and waits for the reply, the communication is typically known as synchronous communication. No further action could be done by the application till it receives the reply. But in *asynchronous communication* the application sends the message and continues processing without waiting for reply from another application [8].

Middleware in general could be defined as software that is designed for building large scale distributed systems [5]. Middleware is connectivity software that consists of a set of enabling services that allow multiple processes running on one or more machines to interact across a network [2,5].

To open up from the tightly synchronized hardware *Messaging-Oriented-Middleware (MOM)* provides the reliable data delivery mechanisms. MOM also lets the systems to be loosely coupled - not always operating at the same speed,

sometimes disconnected, and not having the recipient synchronously locked until the communication has completed [10].

If any new client has to be added in the messaging infrastructure then in highly coupled system the new client should know the location of all the existing clients [1], but in the MOM architecture the new client has to connect only to the middleware. If there are N numbers of client, then for a new client to be added in a tightly coupled architecture it must add N number of new connections but in MOM architecture only one new connection is added irrespective of the number of existing clients.

2.2 Java Message Service (JMS)

The JMS API provides the way to decouple the clients. JMS is a specification, which contains interfaces and abstract classes in itself needed by the messaging clients while communicating with messaging systems. If any of the components is down it does not hinder working of the system as a whole. JMS supports two major domains of messaging [8]:

Publish and subscribe domain of messaging (Pub/sub) is used when a group of users are to be informed about a particular event. The destination of the message is not the application component but the messages are delivered to the virtual destination called *'topic'* [8]. This model allows the publisher or message-producer to broadcast the message to one or more subscribers. *Point-to-Point* domain of messaging Point-to-point messaging domain communicates between clients using the concept of *'queue', 'sender'* and *'receiver'*. *Sender* sends all the messages addressed to a specific `queue`. All the messages are kept in the queue until the receiver fetches them or the message expires. There can be multiple senders to the queue but only single receiver.

JMS provides standard API that java developers can use to access the common features of the enterprise message systems. The design aim of JMS is to provide consistent set of interfaces that messaging clients can use independent of the underlying message system provider. The basic component of the JMS architecture is a message [7,8].

Major components to build up the application are:

Administered objects: JMS destinations and connection factories are maintained administratively and not programmatically. The messaging client lookup these administered object using JNDI API.

***Connection Factories*:** Connection factory encapsulates set of connection configuration parameters defined by the administrator. This object is used to create the connection.

***Destination**:* A destination is the object a client uses to specify the target of messages it produces and the source of message it consumes.

***Connections**:* Connection object provides resource allocation and management. Connections are created by the connection-factories and encapsulate a virtual connection with a JMS provider.

***Sessions**:* Sessions are objects, which provide context for producing and consuming messages. Session creates the message producers, message consumers and message itself.

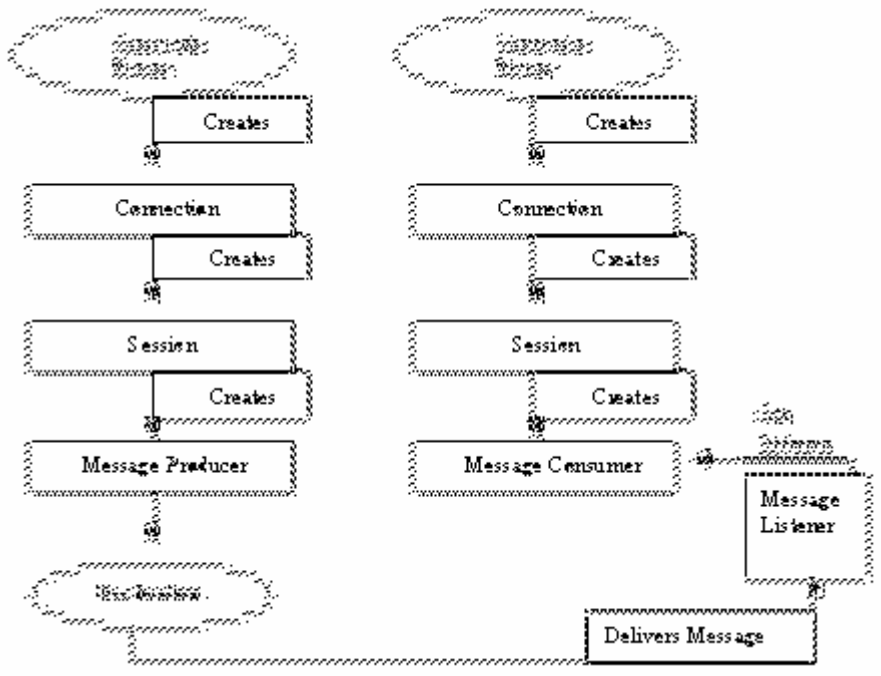


Fig. 1. Architecture of JMS application [8]

Message Producers: Session object creates message producers, to send or publish the message to the destination.

Message Consumers: Session object also creates message consumers, to receive the messages sent to a destination. The message consumer registers the interest in a destination with a JMS provider.

Messages: Message is the most important part of the messaging system, which is the point of communication between applications. The purpose of JMS application is to produce and consume messages.

JMS defines five types of message body [7]: *(i) Text Message (ii) Object Message (iii) Bytes Message (iv) Map Message (v) Stream Message*

The `Message-Listener` interface has a method called `onMessage()` The JMS provider invokes this method automatically when the message arrives. This is called as asynchronous message delivery.

3 Transfer of Messages in Distributed Systems

This section proposes the Transfer of Messages in Distributed Systems (TMDS) architecture. The proposed TMDS architecture is based on the extension of specifications provided by JMS. The application should be able to handle the messages not only when the consumer of the message is disconnected, but the

application should also be able to provide the acknowledgment of the received message, TMDS architecture takes care of these issues by implementing durable subscribers and different acknowledgment modes. Few applications cannot afford to have messages re-delivered, TMDS architecture takes care of this situation, by implementing the once-and-only once delivery mechanism.

TMDS architecture supports XML message formats. The benefit of using the XML message format is that, the industry has a unanimously agreed standard of communication and the messages can be shared between different vendors without any conflict. One more advantage of using XML data format is that, self-defined data formats can be used. The motivation to develop the TMDS architecture is to support both the models of messaging; Publish/Subscribe and Point-To-Point in the same architecture. This can be explained as:

- To develop architecture for electronic exchange.
- To provide the ability to transport XML data as a document.
- To ensure the delivery of important messages to the recipient and acknowledge the delivery.

A message is sent from one participant (the Sender) to a second participant (the Recipient). Additionally, it might be sent on behalf of a third participant (the Originator). Essentially, an interaction (message) between two participants might require the recipient to forward a similar message to some other participant. In this case, it is often necessary for the latter to know for whom the message is being sent. This can be modelled as shown in Figure 2.

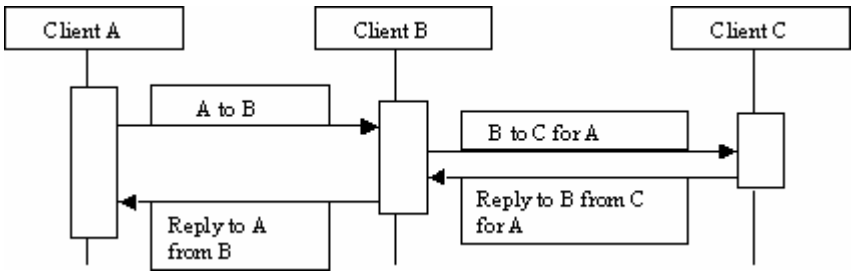


Fig. 2. XML data transfer between applications

3.1 Message Acknowledgment

Message acknowledgment protocol is most important in the guaranteed messaging domain. The JMS API provides message acknowledgment infrastructure. The successful message consumption takes place in three stages: (*i*) Message is received, (*ii*) Message is processed, and (*iii*) Message is acknowledged. Acknowledgment is set when the session is created:

```
TopicSession topicSession =topicConnection.createTopicSession
                              (false, Session.AUTO_ACKNOWLEDGE)
```

Different types of message acknowledgment that are supported: `AUTO_ACKNOWLEDGE`, `DUPS_OK_ ACKNOWLEDGE`, and `CLIENT_ ACKNOWLEDGE`.

`AUTO_ ACKNOWLEDGE`: The session automatically acknowledges a client's receipt of a message when the client has successfully executed the `receive()` method for the queue or when the `messageListener()` is successfully executed for the topic. `AUTO_ACKNOWLEDGE` mode can be viewed in three different perspectives: Message producer, Message server and Message consumer.

Publish() and send() methods of TopicPublisher and QueuerSender respectively are synchronous methods. These methods are responsible for sending the message and wait for an acknowledgment from the server. If the server is down or the message expires and the acknowledgment could not be sent the message is considered to be undelivered and the message is sent again.

From the server perspective, an acknowledgment is sent to the producer of the message means that the server has received the message and it takes the responsibility to deliver the message to the concerned recipient, but it has not yet reached the final destination. Messages are further classified as `PERSISTENT` and `NON-PERSISTENT`. For persistent message the server first writes the message to the disk (store-and-forward mechanism) and then sends the acknowledgment to the producer.

In case of non-persistent message the server may send the acknowledgment as soon as it receives the message and the message is kept in the memory of the server, and if the server dies before delivering the message the message is lost and can not be recovered.

The subscriber can also be divided into two categories namely: durable subscriber and non-durable subscriber. If any of the clients is of durable nature then the JMS server keeps the message in the persistent storage till it receives acknowledgment from all the clients. Certain clients cannot afford redelivered message. To prevent this the JMS server sets the flag for the `getJMSRedelivered()` method, and thus guards against re-delivery of message and thus ensures once-and-only-once delivery of messages.

`DUPS_OK_ACKNOWLEDGE`: This type of acknowledgment is used if any application can afford to receive duplicate messages. `AUTO_ACKNOWLEDGMENT` incurs in extra over-head and affects the performance.

`CLIENT_ACKNOWLEDGE`: A client acknowledges a message by calling the `acknowledge()` method of message. Acknowledging a consumed message automatically acknowledges the receipt of all messages that have been delivered in that session prior to the consumption of the acknowledged message.

3.2 Allowing Messages to Expire and Setting Message Priority

TMDS architecture provides the facility to expire the message after a certain amount of time to increase the performance of the application. TMDS also allows setting the priority level of the message for urgent messages. Both of these values can be set in the `publish()` method of `TopicPublisher` class.

```
TopicPublisher.publish(message, DeliveryMode.NON_PERSISTENT,
                       7, 5000);
```

The above line of code sets the priority level of 7 (0 - lowest, 10 - highest) for the message and the time to live the message is 5 seconds. By default the message never expires and its priority level is 4. If the time-to-live is set to 0 the message never expires.

3.3 Creating Durable Subscription

TMDS architecture uses the `PERSISTENT` messages and durable subscription for the subscriber to ensure that the messages are delivered to the client. A durable subscriber has a higher over-head. The durable subscriber registers a durable subscription with a unique identity that is retained by the messaging server. Non-durable subscribers receive the messages only when they are active but durable subscribers receive all the messages for their subscription period whether or not they are active. The messaging server keeps the message in the persistent storage till the message is delivered to all the durable subscribers or the message expires. A durable subscription can have only one active subscriber at a time. The client ID is set administratively for a client specific connection factory using the `j2eeadmin` command.

```
J2eeadmin –addJmsFactory DURABLE_TCF topic –props clientID=MYID
```

For the normal subscriber, the subscription is only when the subscriber is active, but for the durable subscriber the subscription is still active even if the subscriber is off-line and the subscription lasts till the `unsubscribe()` method is called.

3.4 Features of TMDS Architecture

The messages are not directly delivered to the recipient but the messages are delivered to the recipient via the virtual destinations called *'topic'* or *'queue'*. Destinations are the delivery labels in messaging rather than the place where the message is ultimately delivered. A destination is the commonly understood staging area for the message. The overview of TMDS Architecture distinguishes the two JMS messaging domains.

Point-to-Point (PTP): Produces messages to a named *'queue'*, which is the virtual destination of the message, placing new messages at the back of the queue. Prospective consumers of messages addressed to a queue can either receive the front-most message (thereby removing it from the queue) or browse through all the messages in the queue, causing no changes. Several clients can send messages to a *'queue'*, but only one client can receive th e message (one-to-one communication).

Publish and Subscribe (Pub/Sub): Produces messages to a *'topic'*, which is also a virtual destination for the message like *queue*. Prospective consumers of messages addressed to a topic simply subscribe to the topic. While a message can have many subscribers (one-to-many), the producer does not know how many subscribers, if any, exist for a topic.

Scalability: With TMDS Architecture, a B2B exchange can readily scale to more number of trading partners without requiring changes to the routing architecture or the trading applications. New e-business partners can subscribe to the existing *Topic*

and get the information, they also have the option to get a dedicated channel for communication from the e-broker.

Reliability: Messages can be guaranteed to persist when a message is sent to a queue. If the pub/sub domain of messaging is used, then the message property must be set to PERSISTENT to ensure guaranteed delivery of message. Mobile users, although connected to the network frequently, need not be concerned that they missed out on messages published when they were unable to receive them.

High Performance: TMDS architecture enables flexible programming models, both PTP and pub/sub domains of messaging have been implemented. If any message is to be directed only to the concerned e-business partner, it is delivered via the PTP model and saves the over-head of publishing the message. The second model (Pub/Sub) is used when the message is to be sent to a group of interested recipient.

Enterprise Application Integration: The basic purpose of TMDS architecture is to enable communication between enterprise applications. And most of the enterprise works on their legacy systems and wont be interested in changing their systems. If standard data format is used for communication, this could solve the purpose to some extend. XML was designed to describe the data and to focus on actual data. Users can define their own XML tags to describe the data.

4 TMDS Architecture with XML: A Case Study

The traditional way of exchanging data was through EDI, which uses proprietary data formats, which is defined by a specific company and cannot be used without their permission. XML offers a method to represent the data that is not proprietary. However XML does not have a reliable way of transporting critical business data over the intra-company communication environment. Server or network failure can occur during communication. Applications participating in the distributed environment can crash or have scheduled down time, thus a reliable transport mechanism is required to overcome these issues, e.g. if any client wants to communicate price changes to all the other parties, it must be ensured that message is delivered to all the disconnected clients as well. The RPC mechanism doesnt provide features like: persistence, verification and transactional support, so these features have to be embedded in the application logic.

4.1 Case Study

The TMDS Architecture enables to communicate between different trading partners. It has been simulated for a transport exchange but can be extended for any application where clients have to communicate with each other in a distributed environment.

There are four major components in this application: (*i*) Client, (*ii*) RMI Server, (*iii*) Message Server, and (*iv*) Transport Companies (could be any e-partner in the business).

The client has to be a registered user of the site. If the client is accessing the site for the first time he has to register himself with the e-broker and the details will be stored in the database. As the client enters the site, he has to mention the origin and

destination of the goods to be delivered along with the date of delivery. The client has two options for selecting the Transport Company. The client could either select a specific transport company to deliver the goods or they could select no choice, if the client is not sure of the company to be used. If the client opts for a specific transport company the request is sent to the transport company through a queue and if the client has no specific option the order is published to all the transport companies.

If the order is published to all the clients then the transport companies reply back their interest in the specific order to the Message Server via the queue.

```
public interface Interface extends Remote
{
     public void publishOrder (String cFrom, String cTo,
               String cDay, String cMonth, String cYr)
               throws java.rmi.RemoteException;
     public void queueOrder (String cName)
               throws java.rmi.RemoteException;
}
```

The sample code for the publisher to the Topic of message is as follows:

```
public void publishOrder(String cFrom, String cTo, String cDay,
   String cMonth, String  cYr)
{
     String   topicName = null;
     Context  jndiContext = null;
     TopicConnectionFactory
     topicConnectionFactory = null;
     TopicConnection  topicConnection = null;
     TopicSession  topicSession = null;
     Topic  topic = null;
     TopicPublisher  topicPublisher = null;
     TextMessage  message = null;
     final int  NUM_MSGS =1;
}
```

/* Creates a string with XML tags and publishes this message to the topic, which the subscribers receive and store it in the file and parses the file to get the data. */

```
String xmlString = null;
Date date = new Date ();
xmlString="<?xml version="+"\"1.0\""+" ?>
     <order><origin>"+cFrom+"</origin>
     <destination>"+ cTo +"</destination>
     <delivery_day>"+cDay +"</delivery_day>
     <delivery_month>"+cMonth+"</delivery_month>
     <delivery_yr>"+cYr+"</delivery_yr>
     </order>";
```

/* Create a JNDI InitialContext object if none exists yet.*/

```
topicName = "NewOrder";
try
{
     jndiContext = new InitialContext ();
}
catch(NamingException e){
     System.out.print("JNDI Error "+e.toString ());
```

```
            System.exit (1);
        }

        /* Look up connection factory and topic. If either does  not exist, exit. */

        try
        {
            topicConnectionFactory=(TopicConnectionFactory)
                jndiContext.lookup"TopicConnectionFactory"
            topic = (Topic)jndiContext.lookup (topicName);
        }
        catch(NamingException e){
            System.out.println("Lookup Fail"+e.toString());
            System.exit (1);
        }

        /* Create connection, Create session from connection; false means
        session is not transacted. */

        try
        {
            topicConnection =
            topicConnectionFactory.createTopicConnection();
            topicSession=topicConnection.createTopicSession
                (false, session.AUTO_ACKNOWLEDE);
            topicPublishr=topicSesion.createPublisher(topic)
            message = topicSession.createTextMessage ();

            /*  sets the message stream to xml message format.*/

            message.setText(xmlString)"Order Recived:"+date;
            System.out.print("New Order"+message.getText());
            topicPublisher.publish (message);
        }
        catch(JMSException e) {
            System.out.println("Exception:"+ e.toString ());
        }
        finally
        {
            if(topicConnection != null){
                try{
                    topicConnection.close();
                }
                catch(JMSException e){}
        }
    }
}
```

The message, which is published at the message server and then is send to the subscribers, is transferred in XML format. Similarly, there is another class namely `TopicSubscriber` which subscribes to any specific `topic` and receives all the published messages.

5 Conclusion and Future Work

TMDS architecture supports both the domain of messaging, pub/sub and Point-to-Point, thus enhancing the features of JMS specification. The basic aim of the architecture is to enhance the features of existing Middle-ware technologies (like RMI, DCOM etc.). TMDS architecture uses the advantages of existing Middle-ware technology and enhances the feature of distributed communication by adding asynchronous communication facility, prioritizing the message delivery as per the importance of message and a standard format of data transfer (XML format).

Aim of a new architecture shouldnt be to replace the existing one, but it should have the capability to be integrated with the existing system. TMDS architecture has the capability of integration with the existing messaging systems (supports EAI). There are a lot of commercial products for communication in distributed environment, but industry is still waiting for a standard architecture. TMDS architecture is a step, which implements and extends the features of communication standards proposed by Sun Micro-systems JMS. TMDS architecture uses the data-centric XML. Using the document-centric XML can still enhance the features of the architecture.

References

[1] Adler, R. M. "Distributed Coordination Models for Client/Sever Computing." *Computer 28,* 4, 14-22 April 1995.

[2] Bernstein, Philip A. "Middleware: A Model for Distributed Services." *Communications of the ACM 39*, 2, 86-97, February 1996.

[3] Eckerson, Wayne W., "Three Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications", *Open Information Systems 10*, 20, 1, January 1995.

[4] Newell D., Jones, O., and Machura, M. "Interoperable Object Models for Large Scale Distributed Systems," 30-31. *Proceedings. International Seminar on Client/Server Computing*. La Hulpe, Belgium, October 30-31, 1995. London, England: IEE, 1995.

[5] Rao, B.R. "Making the Most of Middleware." *Data Communications International 24*, 12, 89-96 September 1995.

[6] Schill, Alexander. "DCE-The OSF Distributed Computing Environment Client/Server Model and Beyond," 283. *Intl. DCE Workshop,* Germany, Springer-Verlag, Oct. 1993.

[7] Gopalan S, Grant S., Giotta P *Professional JMS*, 1st Edition, Wrox publicaton.

[8] Mark Hapner, Java Message Service Specification version 1.0.2b, Sun Microsystems, August, 2001, http://java.sun.com/jms/

[9] A. B. Arulanthu, C. ORyan, D. C. Schmidt, M. Kircher, and J. Parsons, The Design and Performance of a Scalable ORB Architecture for CORBA Asynchronous Messaging, *Proceedings of the Middleware2000 Conference*, ACM/IFIP, Apr. 2000.

[10] Birmen K., *Building Secure and Reliable Network applications"* Manning Publishing and Prentice Hall, December 1996.

Recursive Node Similarity in Networked Information Spaces

J.P. Grossman

Department of Mathematics and Statistics, Dalhousie University
Halifax, Nova Scotia, Canada, B3H 3J5
jpg@alum.mit.edu

Abstract. The link structure of a networked information space can be used to estimate similarity between nodes. A recursive definition of similarity arises naturally: two nodes are judged to be similar if they have similar neighbours. Quantifying similarity defined in this manner is challenging due to the tendency of the system to converge to a single point (i.e. all pairs of nodes are completely similar).

We present an embedding of undirected graphs into $\mathbf{R}^n$ based on recursive node similarity which solves this problem by defining an iterative procedure that converges to a non-singular embedding. We use the spectral decomposition of the normalized adjacency matrix to find an explicit expression for this embedding, then show how to compute the embedding efficiently by solving a sparse system of linear equations.

1 Introduction

In recent years the study of networked information spaces has grown in importance due to the availability of citation databases and the increasing prevalence of the internet as a source of information. A powerful and general tool that can be used to understand a networked information space is to find an embedding into $\mathbf{R}^n$ that respects the link structure of the space and/or the information content of its nodes. Content-based embeddings such as Term Frequency Inverse x Document Frequency [8] and Common Citation x Inverse Document Frequency [4] as well as link-based embeddings such as Recursive Neighbourhood Means [6] and Authority Vectors [5] serve several purposes:

Visualization. Embeddings into $\mathbf{R}^2$ or $\mathbf{R}^3$ lead to visualizations of the networked information space that can highlight its clusters and connectivity. Embeddings into higher dimensional spaces can be converted to embeddings into $\mathbf{R}^2$ or $\mathbf{R}^3$ using a variety of dimensionality-reducing techniques.

Clustering. Many clustering algorithms take as their input a set of data points in $\mathbf{R}^n$ ([2], [3]). An embedding can serve as the starting point for such an algorithm in order to search for clusters of related or highly interconnected nodes in the original networked information space.

T. Bh me, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 104-113, 2003.

Node Similarity. An embedding provides a natural measure of relatedness between two nodes given by the Euclidean distance between them.

In this paper we focus our attention on link-based embeddings, which are typically easier to compute as it is not necessary to process the information content of the nodes. Our specific motivation is to develop an embedding for the purpose of measuring node similarity based on the following intuitive and recursive definition of "similar": two nodes are similar if they have similar neighbours. We begin by treating a networked information space as an undirected graph. While the links of a networked information space are usually directed, a link between two nodes indicates some specific similarity between the nodes independent of the link orientation, so we ignore link directions.

The difficulty in defining an embedding based on recursive similarity lies in avoiding the singular embedding wherein all nodes are mapped to the same point. Because of this problem, previously reported graph embeddings have been iterative in nature ([6], [7]). The main contribution of this paper is an explicit (i.e. non-iterative) embedding of graphs into $\mathbf{R}^n$. Explicit embeddings have the advantages of being more amenable to mathematical analysis and potentially faster to compute. The central idea is to define an iterative procedure that converges to a non-singular embedding, then find an explicit formula for this embedding.

In this paper we present the theoretical aspects of the embedding and we show that it can be computed by solving a sparse system of linear equations. The following section reviews some properties of normalized adjacency matrices that will be required for our derivations. In Section 3 we develop the embedding by showing how to prevent the embedded vertices from collapsing to a single point. Finally, in Section 4 we conclude and outline directions for further work.

2 Background

We begin by reviewing some basic facts and definitions that will be used throughout this paper. Let G be a connected, undirected graph on n vertices with vertex set $\{v_i\}$, $1 \le i \le n$. Write $i \sim j$ if vertices v_i and v_j are connected in G. The *adjacency* matrix $\mathbf{A}$ is the $n \times n$ matrix defined by $a_{ij} = 1$ if $i \sim j$ and $a_{ij} = 0$ otherwise. The *degree* matrix $\mathbf{D}$ is the diagonal matrix defined by $d_{ii} = \deg(v_i)$ and $d_{ij} = 0$ $(i \ne j)$. The *normal* matrix $\mathbf{N}$ is defined by $\mathbf{N} = \mathbf{D}^{-1}\mathbf{A}$. Equivalently, $\mathbf{N}$ is the row-normalized adjacency matrix where the sum of the elements in each row is 1.

Because $\mathbf{N}$ is similar to the symmetric matrix $\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, it is diagonalizable. Let $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n$ be the eigenvalues of $\mathbf{N}$ with eigenvectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$, and let $\mathbf{Q}$ be the eigenvector matrix $\mathbf{Q} = [\, \mathbf{a}_1 \;\; \mathbf{a}_2 \;\; \cdots \;\; \mathbf{a}_n \,]$. Then:

$$\mathbf{N} = \mathbf{Q} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \mathbf{Q}^{-1} \,. \tag{1}$$

The following statements can be made regarding the spectrum of **N**:

Lemma 1 *For a connected, undirected graph G with* λ_i, $\mathbf{a}_i$ *as defined above:*

(i) $-1 \le \lambda_i \le 1$

(ii) $\lambda_1 = 1$ and we can take $\mathbf{a}_1 = \mathbf{1}$ (where **1** is the vector of all 1's)

(iii) $\lambda_2 < 1$ (i.e. the eigenvalue 1 has multiplicity 1)

(iv) $\lambda_n = -1$ if and only if *G* is bipartite

Proof See [1], in which equivalent statements are proven for the Laplacian matrix $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ which has eigenvalues $1 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. For the second part of (ii), we note that $\mathbf{N1} = \mathbf{1}$ since the rows of **N** sum to 1. ■

Now let $\mathbf{b}_i$ be the vectors such that $\mathbf{Q}^{-1} = [\, \mathbf{b}_1\ \mathbf{b}_2\ \ldots\ \mathbf{b}_n \,]^{\mathrm{T}}$, i.e. $\mathbf{a}_i \cdot \mathbf{b}_j = \delta_{ij}$. Then we have the following spectral decomposition of **N**, equivalent to (1):

$$\mathbf{N} = \sum_{i=1}^{n} \lambda_i \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \ . \tag{2}$$

The spectral decomposition is useful for working with powers of **N** as for any non-negative integer *k*,

$$\mathbf{N}^k = \sum_{i=1}^{n} \lambda_i^k \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \ . \tag{3}$$

If the eigenvalues λ_i are all non-zero then equation (3) is also valid for negative integers *k*. From Lemma 1 and Equation (3), we see that if *G* is not bipartite then $\mathbf{N}^k$ converges to a limit as $k \to \infty$, and we can define $\mathbf{N}^\infty = \lim_{k\to\infty} \mathbf{N}^k = \mathbf{a}_1\mathbf{b}_1^{\mathrm{T}} = \mathbf{1}\mathbf{b}_1^{\mathrm{T}}$. Note that for any $0 < \alpha < 1$ we have

$$\lim_{k\to\infty} (\alpha\mathbf{N} + (1-\alpha)\mathbf{I})^k = \lim_{k\to\infty} \sum_{i=1}^{n} (\alpha\lambda_i + 1 - \alpha)^k \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} = \mathbf{a}_1\mathbf{b}_1 = \mathbf{N}^\infty \ . \tag{4}$$

This alternate definition for $\mathbf{N}^\infty$ remains valid when *G* is bipartite. The following theorem allows us to easily compute $\mathbf{N}^\infty$:

Theorem 2 *The i^{th} coordinate of* $\mathbf{b}_1$ *is* $\dfrac{\deg(v_i)}{\mathrm{vol}(G)}$ *where* $\mathrm{vol}(G) = \sum_{i=1}^{n} \deg(v_i)$.

Proof Since $\mathbf{D}^{-\frac{1}{2}}\mathbf{N}\mathbf{D}^{-\frac{1}{2}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is symmetric, by definition (4) so is $\mathbf{D}^{-\frac{1}{2}}\mathbf{N}^{\infty}\mathbf{D}^{-\frac{1}{2}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{1}\mathbf{b}_1^{\mathrm{T}}\mathbf{D}^{-\frac{1}{2}}$. But if $\mathbf{b}_1 = [u_1\ u_2\ \cdots\ u_n]^{\mathrm{T}}$, then the $(i, j)^{\mathrm{th}}$ entry of $\mathbf{D}^{-\frac{1}{2}}\mathbf{1}\mathbf{b}_1^{\mathrm{T}}\mathbf{D}^{-\frac{1}{2}}$ is $u_j\sqrt{\dfrac{\deg(v_i)}{\deg(v_j)}}$, so for all i, j:

$$u_j\sqrt{\frac{\deg(v_i)}{\deg(v_j)}} = u_i\sqrt{\frac{\deg(v_j)}{\deg(v_i)}} \quad \Rightarrow \quad \frac{u_i}{\deg(v_i)} = \frac{u_j}{\deg(v_j)} \ .$$

It follows that $u_i = C\deg(v_i)$ for some constant C. But $1 = \mathbf{a}_1\cdot\mathbf{b}_1 = \mathbf{1}\cdot\mathbf{b}_1 = \sum u_i = C\mathrm{vol}(G)$ so $C = 1/\mathrm{vol}(G)$. ■

3 Recursive Node Similarity

The graph embedding presented in this paper was motivated by the desire to measure node similarity in a networked information space, and is based on the intuition that two nodes are likely to be similar if their neighbours are similar. This recursive definition of similarity naturally gives rise to the following general process of iterative refinement: given an embedding of a graph G on n vertices in $\mathbf{R}^m$ with vertex v_i at $\mathbf{x}_i$, we move v_i towards the mean $\mathbf{y}_i$ of its neighbours, where

$$\mathbf{y}_i = \frac{1}{\deg(v_i)}\sum_{i\sim j}\mathbf{x}_j \ . \tag{5}$$

We can express (5) in terms of the normal matrix $\mathbf{N}$ of G by taking the $\mathbf{x}_i$ and $\mathbf{y}_i$ to be row vectors of the $n\times m$ matrices $\mathbf{X}$ and $\mathbf{Y}$ respectively; then $\mathbf{Y} = \mathbf{N}\mathbf{X}$. Thus, our iterative procedure is to start with some initial embedding given by the rows of the $n\times m$ matrix $\mathbf{X}_0$, then define the k^{th} embedding $\mathbf{X}_k$ by

$$\mathbf{X}_k = (\alpha\mathbf{N} + (1-\alpha)\mathbf{I})\mathbf{X}_{k-1} = (\alpha\mathbf{N} + (1-\alpha)\mathbf{I})^k\mathbf{X}_0 \tag{6}$$

where $0 < \alpha \le 1$ is an implementation parameter which determines the extent to which a vertex is perturbed towards the mean of its neighbours at each iteration. When $\alpha = 1$, this is exactly the Recursive Neighbourhood Means algorithm presented in [6].

The difficulty with this algorithm is that $\mathbf{X}_k$ converges to $\mathbf{N}^{\infty}\mathbf{X}_0$ with all vertices embedded at the same point $\mathbf{b}_1^{\mathrm{T}}\mathbf{X}_0$ (unless $\alpha = 1$ and G is bipartite, in which case $\mathbf{X}_k$ does not converge at all). Thus, solving explicitly for the limiting embedding is

uninteresting, and an iterative approach is required. In [6] the initial embedding $\mathbf{X}_0$ is chosen randomly, then ~10 iterations are performed to obtain the final embedding. In [7] a slightly different approach is proposed. First, they use $\alpha = 0.5$ to eliminate oscillations due to negative eigenvalues, and they consider only $m = 1$. Then after each iteration the $\mathbf{x}_i$ are translated so that their mean is zero, and scaled so that the difference between the largest and smallest values is always 2.0. The system is allowed to converge, but solving for the limiting embedding explicitly is infeasible due to the data dependent translation/scale operation performed at each iteration.

3.1 Preventing Collapse

We can prevent the vertex embeddings from collapsing to a single point by slightly modifying the iteration equation (6). The idea is to treat $\mathbf{X}_0$ not only as an initial embedding but as a "scaffolding" which holds the vertices apart. We define the k^{th} embedding $\mathbf{X}_k$ by

$$\mathbf{X}_k = \alpha\mathbf{N}\mathbf{X}_{k-1} + (1 - \alpha)\mathbf{X}_0$$

where $0 < \alpha \leq 1$ is again an implementation parameter which determines the extent to which a vertex is affected by its neighbours. If $\alpha = 1$ then $\mathbf{X}_k = \mathbf{N}^k\mathbf{X}_0$ as before, but if $\alpha < 1$ then we can verify by induction that

$$\mathbf{X}_k = [\alpha^k\mathbf{N}^k + (1 - \alpha)(\mathbf{I} - \alpha^k\mathbf{N}^k)(\mathbf{I} - \alpha\mathbf{N})^{-1}]\mathbf{X}_0 \,. \tag{7}$$

Note that part (i) of Lemma 1 implies that $(\mathbf{I} - \alpha\mathbf{N})$ is invertible for $0 < \alpha < 1$. Using $\mathbf{X}_0$ as a scaffolding prevents the embedding from collapsing to a point, so it becomes useful to ask whether the system defined by (7) converges, and if so to find the limiting embedding. But part (i) of Lemma 1 tell us that the entries of $\mathbf{N}^k$ are bounded in absolute value, so $\alpha^k\mathbf{N}^k \to 0$ as $k \to \infty$, and thus

$$\lim_{k\to\infty} \mathbf{X}_k = (1-\alpha)(\mathbf{I} - \alpha\mathbf{N})^{-1}\mathbf{X}_0 \,. \tag{8}$$

Intuitively, α measures the degree of recursive similarity captured by the embedding (8). If α is close to 0 then $(\mathbf{I} - \alpha\mathbf{N})^{-1} \approx (\mathbf{I} + \alpha\mathbf{N})$, so to first order the final position of a vertex depends only on the initial positions of that vertex and its neighbours. In particular, we have the following:

Claim 3 *If α is sufficiently close to* 0 *and* $\mathbf{X}_0 = \mathbf{I}$*, then the vertices closest to v_i in the embedding* (8) *are exactly the neighbours of v_i.*

Proof Ignoring the constant factor of $(1 - \alpha)$, the embedding is given by the rows $\mathbf{x}_i$ of the matrix $(\mathbf{I} - \alpha\mathbf{N})^{-1} = \mathbf{I} + \alpha\mathbf{N} + o(\alpha^2)$. The i^{th} coordinate of $\mathbf{x}_i$ is $1 + o(\alpha^2)$. For $i \neq j$ the j^{th} coordinate is $\alpha/\deg(v_i) + o(\alpha^2)$ if $i \sim j$ and $o(\alpha^2)$ otherwise. Hence:

$$|\mathbf{x}_i - \mathbf{x}_j|^2 = \begin{cases} 2 - \dfrac{2\alpha}{\deg(v_i)} - \dfrac{2\alpha}{\deg(v_j)} + o(\alpha^2) & \text{if } i \sim j \\ 2 + o(\alpha^2) & \text{otherwise} \end{cases}$$

From this it follows that for sufficiently small α the closest vertices to v_i are its neighbours, and moreover that neighbours of low degree are closer than neighbours of high degree. ■

The larger α becomes, the greater the extent to which the embedding captures the recursive structure of the graph. As a concrete example, consider the graph shown in Figure 1. Vertices A and B are not connected, but they have the same neighbours. Taking $\mathbf{X}_0 = \mathbf{I}$, so that in the initial position all pairs of vertices are equally distant, gives us, for each choice of α, an embedding in $\mathbf{R}^{11}$. We find that for $\alpha < 0.75$, A is embedded closer to X, Y and Z than to B, corresponding to a non-recursive definition of similarity in which nodes are most similar to their neighbours. For $\alpha > 0.8$, A is embedded closer to B than to X, Y or Z, corresponding to a recursive definition of similarity in which nodes with similar neighbours are themselves similar.

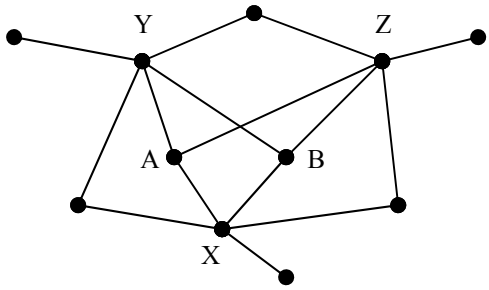


Fig. 1. When $\alpha > 0.8$, A is embedded closer to B than to X, Y or Z.

The preceding discussion provides motivation for taking α as close to 1 as possible. However, as $\alpha \to 1$ we again encounter the problem of the embedding collapsing to a single point. Using the spectral decomposition (2) of $\mathbf{N}$, we can rewrite (8) as:

$$(1-\alpha)(\mathbf{I} - \alpha\mathbf{N})^{-1}\mathbf{X}_0 = \left(\sum_{i=1}^{n} \frac{1-\alpha}{1-\alpha\lambda_i}\mathbf{a}_i\mathbf{b}_i^{\mathrm{T}}\right)\mathbf{X}_0 = \mathbf{N}^{\infty}\mathbf{X}_0 + \left(\sum_{i=2}^{n} \frac{1-\alpha}{1-\alpha\lambda_i}\mathbf{a}_i\mathbf{b}_i^{\mathrm{T}}\right)\mathbf{X}_0 \ . \quad (9)$$

Hence,

$$\lim_{\alpha\to 1}(1-\alpha)(\mathbf{I} - \alpha\mathbf{N})^{-1}\mathbf{X}_0 = \mathbf{N}^{\infty}\mathbf{X}_0 + \lim_{\alpha\to 1}\left(\sum_{i=2}^{n} \frac{1-\alpha}{1-\alpha\lambda_i}\mathbf{a}_i\mathbf{b}_i^{\mathrm{T}}\right)\mathbf{X}_0 = \mathbf{N}^{\infty}\mathbf{X}_0 \ .$$

In this case, however, there is an easy way to fix the problem. We note that an embedding is not qualitatively changed by translating or scaling the embedded vertices. Since each row of $\mathbf{N}^{\infty}\mathbf{X}_0$ is the m-dimensional row vector $\mathbf{b}_1^{\mathrm{T}}\mathbf{X}_0$, we can translate the embedding (9) by $-\mathbf{b}_1^{\mathrm{T}}\mathbf{X}_0$ and then scale by $1/(1-\alpha)$ to obtain the equivalent embedding:

$$\left(\sum_{i=2}^{n} \frac{1}{1-\alpha\lambda_i} \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \right) \mathbf{X}_0 \ . \tag{10}$$

We are now free to set $\alpha = 1$, obtaining

$$\left(\sum_{i=2}^{n} \frac{1}{1-\lambda_i} \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \right) \mathbf{X}_0 \ . \tag{11}$$

We observe that if $f(\alpha)$ denotes the original embedding as a function of α, then Equation (10) is $(f(\alpha) - f(1))/(1-\alpha)$, and thus Equation (11) is in fact $-f'(1)$.

3.2 Computing the Embedding

We can compute the embedding (11) without explicitly finding each $\mathbf{a}_i$, $\mathbf{b}_i$, λ_i by rewriting the summation as follows:

$$\begin{aligned}
\sum_{i=2}^{n} \frac{1}{1-\lambda_i} \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} &= \left(\mathbf{a}_1 \mathbf{b}_1^{\mathrm{T}} + \sum_{i=2}^{n} \frac{1}{1-\lambda_i} \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \right) - \mathbf{N}^{\infty} \\
&= \left(\mathbf{a}_1 \mathbf{b}_1^{\mathrm{T}} + \sum_{i=2}^{n} (1-\lambda_i) \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \right)^{-1} - \mathbf{N}^{\infty} \\
&= \left(\mathbf{N}^{\infty} + \sum_{i=1}^{n} (1-\lambda_i) \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \right)^{-1} - \mathbf{N}^{\infty} \\
&= (\mathbf{N}^{\infty} + \mathbf{I} - \mathbf{N})^{-1} - \mathbf{N}^{\infty} \ .
\end{aligned}$$

The embedding therefore becomes $(\mathbf{N}^{\infty} + \mathbf{I} - \mathbf{N})^{-1}\mathbf{X}_0 - \mathbf{N}^{\infty}\mathbf{X}_0$ which we translate by $\mathbf{b}_1^{\mathrm{T}}\mathbf{X}_0$ to obtain the simpler, equivalent embedding

$$\mathbf{X} = (\mathbf{N}^{\infty} + \mathbf{I} - \mathbf{N})^{-1}\mathbf{X}_0 \ . \tag{12}$$

This can be viewed as an initial embedding $(\mathbf{N}^{\infty} + \mathbf{I} \square \mathbf{N})^{-1}$ into $\mathbf{R}^n$ composed with a linear map $\mathbf{X}_0^{\mathrm{T}}:\mathbf{R}^n \rightarrow \mathbf{R}^m$. For small n it is possible to directly compute the inverse $(\mathbf{N}^{\infty} + \mathbf{I} \square \mathbf{N})^{-1}$. When n is large and in particular when $n >> m$, it is likely more efficient to solve the m linear systems of equations, each in n unknowns, defined by rewriting (12) as:

$$(\mathbf{N}^{\infty} + \mathbf{I} \square \mathbf{N})\mathbf{X} = \mathbf{X}_0 \,. \tag{13}$$

The $n \times n$ matrix $\mathbf{N}^{\infty} + \mathbf{I} \square \mathbf{N}$ is dense; indeed, Theorem 2.2 tells us that $\mathbf{N}^{\infty}$ has no non-zero entries. However, all rows of $\mathbf{N}^{\infty}$ are the same, so if we take $\mathbf{S}$ to be the sparse matrix with 1□s along the diagonal and -1□s along the subdiagonal, then $\mathbf{S}\mathbf{N}^{\infty}$ is zero everywhere but the first row. We can therefore solve the equivalent m systems of sparse linear equations defined by multiplying both sides of (13) by $\mathbf{S}$:

$$\mathbf{S}(\mathbf{N}^{\infty} + \mathbf{I} \square \mathbf{N})\mathbf{X} = \mathbf{S}\mathbf{X}_0 \,. \tag{14}$$

3.3 Relationship with Recursive Neighbourhood Means

The Recursive Neighbourhood Means embedding [6] computes $\mathbf{N}^k\mathbf{X}_0$ where $k \approx 10$. A tradeoff exists in choosing k as larger values of k tend to produce higher-quality embeddings, but as k increases the embedding converges to a single point, and analysis becomes more susceptible to rounding errors. A relationship between Recursive Neighbourhood Means and the embedding (11) is revealed if we approximate the summation as follows:

$$\begin{aligned} \sum_{i=2}^{n} \frac{1}{1-\lambda_i} \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} &\approx \sum_{i=2}^{n} \sum_{j=0}^{k} \lambda_i^j \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \\ &= \sum_{j=0}^{k} \sum_{i=2}^{n} \lambda_i^j \mathbf{a}_i \mathbf{b}_i^{\mathrm{T}} \\ &= \sum_{j=0}^{k} \mathbf{N}^j - (k+1)\mathbf{N}^{\infty} \,. \end{aligned} \tag{15}$$

Translating by $(k+1)\mathbf{b}_1^{\mathrm{T}}\mathbf{X}_0$, this gives us the equivalent approximation embedding

$$\sum_{j=0}^{k} \mathbf{N}^j \mathbf{X}_0 \,.$$

which is simply the sum of the first (k + 1) terms obtained from the Recursive Neighbourhood Means iteration. We emphasize that the approximation (15) is only valid if G is not bipartite. If G is bipartite then λ_n = -1 and $\mathbf{N}^k\mathbf{X}_0$ does not converge as $k \to \infty$.

4 Conclusion

Embedding graphs into $\mathbf{R}^n$ is an important technique for visualizing and analyzing networked information spaces. We have presented an explicit embedding (11) based on recursive node similarity that avoids the problem of all nodes collapsing to a single point. The embedding can be expressed concisely as the inverse of a matrix (12), and computed efficiently by solving sparse systems of linear equations (13).

In this paper we have focused on the theoretical aspects of the embedding; an important direction for future research is to evaluate its practical aspects. Two questions are of particular interest. First, the time required to solve the linear equations in (14) is directly related to the number of non-zero entries in the LU factorization of the sparse matrix $\mathbf{S}(\mathbf{N}^\infty + \mathbf{I} - \mathbf{N})$. It is therefore desirable to characterize this number for the types of graphs which typically arise from networked information spaces. Second, a natural application for the embedding is information retrieval: the embedding can be used to find the N most similar documents to a query document. We can evaluate the embedding in this context by comparing the quality of its search results to those of other methods, such as the algorithms described in [4] and [5].

5 Acknowledgements

We would like to thank Jeannette Janssen and the anonymous reviewers for their helpful comments and suggestions. This work was funded in part by a grant from MITACS (Mathematics of Information Technology and Complex Systems), Canada.

References

1. Fan R. K. Chung, Spectral Graph Theory, CBMS Lecture Notes, Regional Conference Series in Mathematics No. 92, American Mathematical Society, 1995, 207pp.
2. Daniel Fasulo, "An Analysis of Recent Work on Clustering Algorithms", *Technical Report #01-03-02*, Department of Computer Science and Engineering, University of Washington, Seattle, WA, April 26, 1999.
3. C. Fraley, A. E. Raftery, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis", *Computer Journal*, 41(1998):578-588, 1998.
4. C. Lee Giles, Kurt D. Bollacker, Steve Lawrence, "Citeseer: An Automatic Citation Indexing System", In *Digital Libraries 98 – Third ACM Conference on Digital Libraries 98*, pp. 89-98, 1998.

5. Wangzhong Lu, Jeannette Janssen, Evangelos Milios, Nathalie Japkowicz, "Node Similarity in Networked Information Spaces", *Proc. CASCON 2001*, Toronto, Ontario, November 5-7, 2001.
6. James Moody, "Peer influence groups: identifying dense clusters in large networks", *Social Networks 23 (2001), 261-283*, 2001.
7. William D. Richards, Jr., Andrew J. Seary, "Convergence Analysis of Communication Networks", http://www.sfu.ca/~richards/Pages/converge.pdf, 1999, 36 pp.
8. Gerard Salton, C.S. Yang, "On the specification of term values in automatic indexing", *Journal of Documentation,* 29:351–372, 1973.

Mobile Business Processes

Volker Gruhn, Matthias Book

Chair of Applied Telematics/e-Business[1], Dept. of Computer Science, University of Leipzig
Klostergasse 3, 04109 Leipzig, Germany
{volker.gruhn, matthias.book}@informatik.uni-leipzig.de

Abstract. Todays global mark ets demand global processes. Increasingly, these processes are not only distributed, but also contain mobile aspects. We discuss two challenges brought about by these mobile business processes: Firstly, the need to specify the distribution of processes across several sites, and secondly, the need to specify the dialog flows of the applications implementing those processes on mobile devices. To remedy the first challenge, we give an overview of the Process Landscaping method with its support for refining processes across multiple abstraction layers and associating their activities and objects with distinguished locations. Next, we present a Dialog Flow Notation and Dialog Control Framework for the specification and management of complex hypertext-based dialog flows. These tools allow developers to build user interfaces for mobile client devices with different input/output capabilities, which all access the same application logic on a central server.

1 Introduction

The market reach of goods and services is ever increasing today both in the business-to-consumer (B2C) and business-to-business (B2B) sector, transactions are performed on a regional, national or even global scope [22]. The global markets demand global business processes in order to handle those transactions efficiently. However, when looking at global markets, it would be a costly over-abstraction to consider the associated business processes as centralized entities [19]. Rather, they involve distributed teams, distributed service provisioning, and distributed repositories. This environment places higher demands on the infrastructure, coordination, communication and cooperation of the involved parties, all of which affect the suitable process models substantially ([10], [20]). As illustrated in the examples of the Iridium software process and housing industry processes, distribution affects both processes and data.

Recently, an additional challenge has been developing: As working environments are becoming more mobile, we are not just dealing with *distributed* processes anymore. In addition, we need to consider *mobile* processes: All sales-oriented processes tend to become more mobile, and the same is true for processes spreading over various sales channels. Also, processes delivering services to customers lo cations tend to encompass mobile aspects. These mobile processes require flexible support for coordination and communication among the involved parties, as well as controlled remote

[1] The Chair of Applied Telematics/e-Business is endowed by Deutsche Telekom AG.

T. Bh me, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 114-133, 2003.

data access. Under these conditions, a central question is whether the mobility requirements affect the process models themselves or just their execution support [12].

1.1 Mobile Process Landscaping

When modeling the processes of a project, there are some key issues that developers need to resolve: After identifying the core processes, they need to determine a suitable order for modeling them and establish the interfaces between them. With regard to distributed and mobile processes, two especially vital questions are where process parts or activities are to be executed, and which data are needed in which location.

To support the specification, optimization and implementation of distributed processes, the Process Landscaping methodology was developed [13]. It comprises a number of activities that are also suitable for handling mobile business processes. The first step consists of identifying the high-level process clusters, positioning them in a process landscape and establishing their mutual interfaces (Fig. 1).

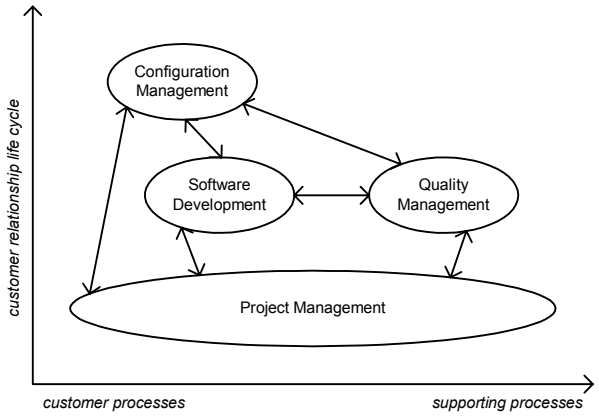


Fig. 1. Process landscape of a software development project (example)

In the following steps, different aspects of the process model are refined in whichever order is most natural in a concrete project: To refine interfaces, the data exchanged between the clusters is specified in combination with the direction of the data flow. Clusters can be refined in two different ways: The developer can either specify a set of sub-clusters that make up a super-cluster, or define a concrete process model that defines the activities performed and deliverables produced in a cluster. Activities in the process model may again be refined by sub-process models. This way, developers can move from a very coarse to a highly detailed definition of processes in a structured way – the overall process landscape serves as an orientation, with refinements being added on lower levels of abstraction as needed. This facilitates easy analysis of the model and discussion of the process.

Besides the structure imposed on the process model by the relationships between super- and sub-clusters, super- and sub-process models, further structural information is specified by assigning locations on the process landscape to objects and activities: Each activity must be assigned to one or more execution locations, and every object type must be assigned to a storage site. Furthermore, the interfaces are first-class entities in Process Landscaping to allow early identification of process relationships [14].

1.2 Mobile Process Implementation Characteristics

Despite the support by modeling methodologies such as Process Landscaping, the implementation of mobile processes is still hindered by a number of major obstacles today: Firstly, the required telecommunications infrastructure might be unavailable or unable to provide the necessary bandwidth. With network availability being less of a problem today (except for some isolated areas) and the introduction of high-volume transmission technologies such as UMTS imminent, this obstacle is starting to fade – however, slow deployment of the network equipment and mobile devices, combined with potentially high introductory prices, will likely limit the speed of adoption of mobile applications for some time to come.

Secondly, the currently employed legacy systems may be too inflexible for immediate integration with mobile processes, and difficult to open up to new access patterns. While not impossible, building suitable interfaces to integrate legacy systems with mobile processes and application front-ends is likely to be a complex and costly task. Similarly, organizational issues and traditional processes may not be compatible with mobile business processes and need to be adapted carefully to realize the full potential of mobile applications.

Finally, among the variety of mobile devices available today, only few mainstream conventions or *de facto* standards have developed yet. Since devices differ widely in aspects such as screen size, input/output interfaces, networking, programming and dialog capabilities, mobile applications either have to cater to the lowest common denominator, or be modified to fit different mobile devices. This becomes most obvious (and challenging) in the area of mobile dialog design.

One approach to solving these problems seems to be the use of hypertext-based user interfaces (UIs) for mobile applications, where the UI consists of web pages presented in a browser. Compared to window-based user interfaces, they require only modest client capabilities, making them especially suitable for mobile devices with their strict energy, memory, input and output limitations [9]. Furthermore, the simple information elements and interaction techniques of hypertext-based UIs can be rendered on various presentation channels, ranging from desktop to mobile devices [3]. This multi-channel thin client scenario requires the application logic to be implemented presentation channel-independently on a central server, while the UI is rendered individually on various client devices [23].

However, when developing applications with hypertext-based UIs, software engineers need to be aware that their implementation differs in some important characteristics from applications with window-based UIs ([21], [26]):

Firstly, the devices different input and output capabilities restrict the amount of information users can work with at a time. Consequently, presentation channel-independent applications must not only implement different UIs, but also be able to

handle different interaction patterns – for example, a task that may be completed in one interaction step with a desktop browser may take three steps on a mobile device and a dozen over an interactive voice response (IVR) system (Fig. 2).

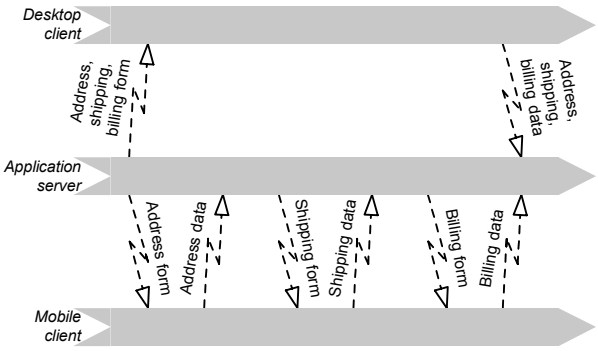


Fig. 2. Different dialog flows on different devices

Secondly, hypertext-based UIs present information on pages instead of in windows. Consequently, interactions that would be performed without involving the application logic in a window-based UI (e.g. closing a window) require the generation of a new page in a page-based UI and thus involve the application logic for every interaction step. Thirdly, hypertext-based UIs employ a request-response mechanism to pull data from the server. Since the application logic cannot push data to the client, it can only react passively to user actions (like clicking on a link) instead of actively initiating dialog steps (like opening a new window). Finally, the Hypertext Transfer Protocol (HTTP) is stateless: The protocol only transports data, but does not maintain any information on the state of the dialog system. Consequently, the application itself has to manage the dialog state for each user session, which requires complicated logic for more complex dialog structures.

Regarding the impact of these characteristics on the user experience, one of the most notable effects is the limitation to simple dialog structures in many hypertext-based applications today: Linear and branched dialog sequences can be easily implemented and are therefore commonplace, but already simple nested structures (e.g. an authorization form inserted at the beginning of a sensitive transaction) require a lot of dialog control logic, and no application that the authors are aware of is capable of nesting arbitrary dialogs on multiple levels.

Since users have a long-established conceptual model of nested dialogs from window-based applications, they will likely transfer that model to hypertext-based applications. However, because of insufficient dialog control logic, many applications still violate users' expectations today when they send them to other pages than they intended to reach (in some web applications, for example, login forms return users to the homepage after a successful login instead of sending them to the area that required authorization, forcing them to navigate manually to the desired area). This violation

of the ISO dialog principles of controllability and conformity with user expectations [17] imposes a high cognitive and memory load on the user.

Since these challenges are independent of a specific application, a desirable solution would be a notation and a framework that can be used for the specification and implementation of any hypertext-based application. After giving an overview of the related work (section 2), we will therefore introduce a Dialog Flow Notation for specifying complex dialog flows (section 3), and present the architecture of a Dialog Control Framework for managing those dialog flows on different devices (section 4).

2 Related Work

A number of notations for the specification of interactive systems' user interfaces have been proposed over time. However, many were developed for traditional window-based applications and are therefore not suitable for the task of modeling the special characteristics of hypertext-based applications presented in section 1.2: While they can model direct manipulation techniques and multiple windows, which hypertext applications lack, they do not provide means for specifying request-response interaction patterns on page-based media.

Other approaches that were explicitly designed to describe hypertext systems mostly focus on *data-intensive* information systems, but not *interaction-intensive* applications [8]: For example, the RMM development process [16] allows the definition of navigable relationships between data entities, and the OOHDM [24] process provides classes like node, link and index to represent different forms of navigation; however, the resulting structures remain flat and cannot be nested arbitrarily. The same is true for the HDM-lite notation used by the Autoweb tool [7], which supports the automatic generation of database schemas and application pages from a conceptual model; and the modeling language DoDL [6], which allows mapping of structured database content to static hypertext pages, but does not support dynamic features. Finally, while the language WebML [5] is capable of modeling simple dynamic features of a data-intensive web application by providing operation units for creating, deleting and modifying entities, it does not support more complex structures such as modular, nestable dialog sequences.

For the implementation of hypertext-based applications, several frameworks exist that separate the user interface from the application logic to facilitate easier dialog control, as suggested by the Model-View-Controller (MVC) design pattern [18]. The Apache Jakarta Project's Struts framework [1] is one of the most popular solutions today. However, Struts forces developers to combine dialog control logic and application logic in the Model implementation, since the Controller does not implement any actual dialog control logic, but merely maps action names to class names (a more thorough discussion of the Struts approach vs. the one suggested in this paper will be presented in section 4).

The challenges of device-independent design are addressed in the Sisl (Several Interfaces, Single Logic) approach [2]. It inserts a so-called 'service monitor' between the central application logic and the presentation logic for each device type to coordinate the events that the interface can generate with the events that the application logic can currently handle. This allows Sisl to support a wide spectrum of devices, in-

cluding speech recognition systems, and handle the partial or unordered input that they may produce. However, since Sisl uses acyclic graphs for modeling dialogs, it seems more suitable for simple prompt- or menu-based interaction scenarios than for highly interactive applications with complex (i.e. nested or cyclic) dialog structures.

We are still missing a solution that controls the dialog structure of a hypertext-based application independently of the implementation of the Model and View tiers, supports different interaction patterns on different devices, and allows developers to work with complex dialog constructs like dialog modules nested on multiple levels. The Dialog Flow Notation and Dialog Control Framework introduced in the following sections are designed to address this need.

3 Dialog Flow Notation

To define the concept of a "dialog flow" and develop the elements of the Dialog Flow Notation (DFN), we first examine the client-server communication taking place when users work with a hypertext-based application. As Fig. 3 shows, a page *A'* displayed on the client is rendered from source code (e.g. HTML) that was first generated by an entity *A* (e.g. a JavaServer Page) on the server and then transmitted to the client. When the user follows a link or submits a form on this page, the resulting data *a* is transmitted to the server. The application logic may now process the data in a number of steps (here: *1* and *2*), which each generate data (*b* and *c*) that is processed in the next step. Finally, the source code for the following page is generated (*B*), transmitted to the client and rendered there (*B'*). Alternatively, user-submitted data (such as *d*) may not require any application logic processing, but directly lead to the generation and rendering of a new page (*C* and *C'*).

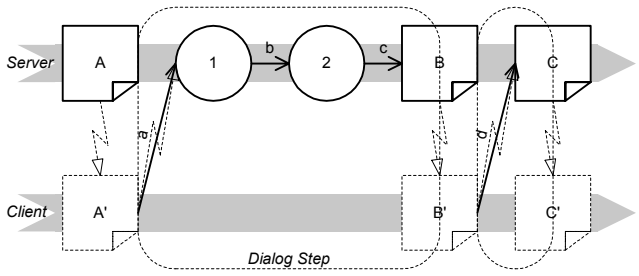


Fig. 3. Client-server communication in HTTP

We call the server activity happening between the submission of a request and the receipt of a response by the client a *dialog step* (in an online shop, for example, a dialog step might begin with submission of the user's billing information, comprise the validation of his credit card data by the application logic, and end with the generation of a confirmation page). Multiple consecutive dialog steps form a *dialog sequence* (for example, an online shop's checkout dialog sequence might be composed of several

dialog steps for collecting the user's address, shipping options, and billing information. Finally, all possible dialog sequences that can be performed on a certain presentation channel of an application constitute that channel's *dialog flow*. An online shop's dialog flow might for example comprise searching for products, looking at detailed product information, putting products into the cart, checking out, etc.

3.1 Notation Elements

Looking back at the communication model in Fig. 3, we realize that the client-server communication and thus the distinction between generating (*A*) and rendering pages (*A'*) is irrelevant for the purpose of modeling dialog flows: When specifying how the user interacts with the application logic via the UI pages, the dialog flow designer does not need to know about technical details such as pages' source code being generated on the server and transmitted to the client prior to rendering.

The DFN therefore only specifies the order of the UI pages and processing steps, and the data exchanged between them. It models the dialog flow as a transition network, i.e. a directed graph of states connected by transitions called a *dialog graph* (Fig. 4).[2] As illustrated in the communication model above, dialog graphs do not need to be bipartite.

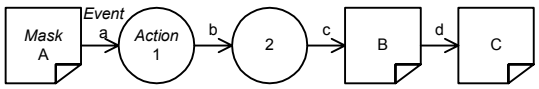


Fig. 4. Dialog graph

The DFN refers to the transitions as *events* and to the states as *dialog elements*, discerning atomic and compound elements. *Atomic dialog elements* are hypertext pages (symbolized by dog-eared sheets and referred to by the more generic term *masks* here) and application logic operations (symbolized by circles and called *actions* from now on). Every dialog element can generate and receive multiple events, enabling the developer to specify much more complex dialog graphs than the linear succession of elements shown above. Which element will receive an event depends both on the event and the generating element (e.g., an event *e* may be received by action *3* if it was generated by mask *D*, but be received by action *4* if generated by mask *E*). Events can carry parameters, i.e. application-specific information such as form input submitted from a mask, and thus facilitate communication between dialog elements.

Theoretically, the complete dialog flow of an application could be described using only atomic elements. However, the resulting specification would be much too complicated to understand, and the flat structure does not support reuse of often-needed dialog graphs. The DFN therefore provides *compound dialog elements* (compounds) which encapsulate dialog graphs and realize the key requirement of nested dialog

[2] The basic concepts and symbols of this notation were inspired by Harel's Statecharts [15], but their semantics have been adapted for the context of hypertext dialog flow specification.

structures: A compound's *interior dialog graph* can contain sub-compounds, and the compound itself can be embedded in the *exterior dialog graphs* of super-compounds. We discern two types of compound dialog elements: *Dialog modules* (symbolized by rectangles with rounded corners) contain an interior dialog graph with one entry point and one or more exit points, while *dialog containers* (symbolized by rectangles) contain an interior dialog graph with one entry point, but no exit points.

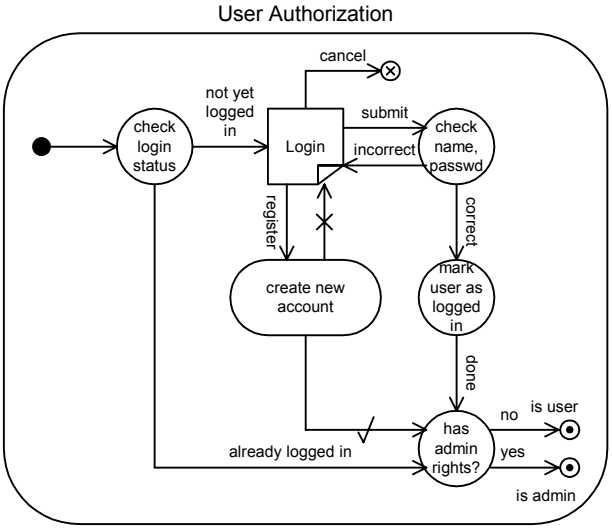
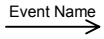


Fig. 5. *User Authorization* dialog module

We will introduce the features of dialog modules using the *User Authorization* module in Fig. 5 as an example. This module checks if the user is already logged in and shows a *Login* mask to prompt for his user name and password, if necessary. If the user's credentials are correct, the module marks him as logged in, checks his access rights and terminates, notifying the super-compound of the user's status. If the user does not yet have an account, he can register using the embedded *create new account* sub-module. Note that by splitting the application logic into relatively fine-grained operations instead of implementing them all in one action, the module can react flexibly to different situations, like bypassing the credential check when the user is already logged in.

Initial and Terminal Events. When a compound receives an event from the exterior dialog graph that it is embedded in, traversal of its interior dialog graph starts with the *initial event*. When the interior dialog graph terminates, it generates a *terminal event* that is propagated to the super-compound and continues the traversal of the exterior dialog graph. Depending on the semantics of the termination, developers can choose between three kinds of terminal events (Table 1):

Table 1. Event types and notation symbols

Event type	Interior dialog graph symbol	Exterior dialog graph symbol
Initial event		*n/a*
Regular terminal event		
Done terminal event		
Cancelled terminal event		
Abort event		*n/a*

Regular terminal events are intended to communicate application-specific information to the terminating modules ex terior dialog graph, such as the result of an operation or decision (for example, the *User Authorization* module generates an *is user* or *is admin* terminal event, depending on the users rights). Often, however, modules do not need to notify their calling super-compound about some application-specific state, but should simply indicate if they completed their task successfully or not. The DFN provides the *done* and *cancelled* terminal events to model these situations (for example, the *create new account* module may terminate with a *done* or *cancelled* event, depending on the success of the registration process). In contrast to regular terminal events, *done* and *cancelled* events are unnamed and cannot carry parameters. Their application-independent semantics enable the dialog control logic to handle them automatically in certain situations, as we will see soon.

Compound Events and Return Mechanism. Complex dialog structures will usually contain a certain amount of redundancy, since some dialog elements may be linked from many other elements in the application. If we had to specify all the respective events explicitly, our dialog graph diagrams would soon become cluttered with redundant information. In his Statecharts notation, Harel introduced a special construct to counter the combinatory explosion of transitions that often plagues state machines: a transition leading from a contour to a state [15].

The DFN uses a similar construct, albeit adapted for dialog flow specification: A so-called *compound event*, symbolized in dialog graph diagrams by an arrow leading from the compounds contour to a certain element, indicates that this event may be generated by every element in the compound. As an example, consider the dialog graph of a simple online shop in Fig. 6:[3] The shops homepage, list of items in each category, detailed description of each item, shopping cart and checkout process shall be linked from every mask in the system. If all events connecting the elements had been specified explicitly, a tangled event web would have been the result. Using compound events, however, we can express the relationships in a much clearer diagram.

[3] The shop was modeled as a dialog container instead of a module since it does not have a natural terminal state.

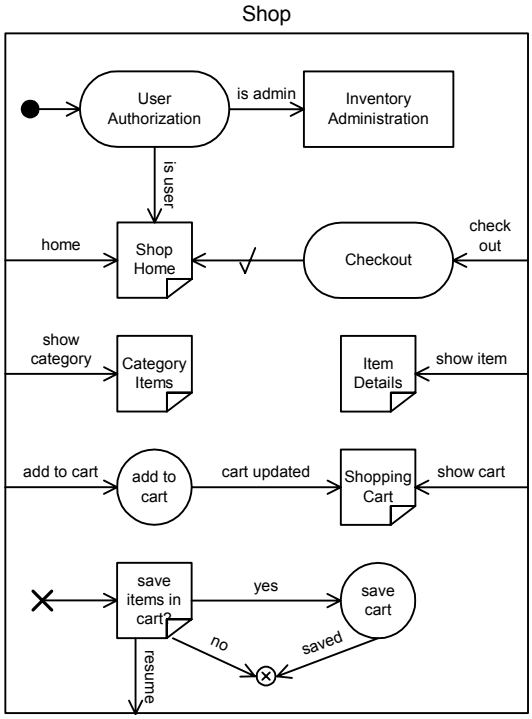


Fig. 6. *Shop* dialog container

Note that the above dialog graph does not explicitly specify what happens when a user does not complete the *Checkout* module with a *done* event, but cancels its execution. For usability reasons, we would not want to return the user to the shop's home page in this case, but to the mask from which he had entered the *Checkout* module (in the same way that window-based applications return the focus to the parent window after the user closed a child window). However, since we do not know at specification time where to return the user, we cannot specify the receiver of the *cancelled* event. The Dialog Control Framework introduced in section 4 solves this apparent dilemma by using the *cancelled* event's application-independent semantics described earlier: If the framework intercepts a *done* or *cancelled* event without a specified receiver, the *return mechanism* automatically leads the event to the dialog mask from which the terminated module was activated, creating the familiar nesting effect for the user. Fig. 7 shows a sample dialog sequence employing this mechanism (the gray arrows indicate the compounds' nesting levels).

The scope of compound events only encompasses the compound that they are specified in, but not its super- or sub-compounds. For example, while the *show item* event leads to the *Item Details* mask from any other mask in the *Shop* container, such a connection does not exist for any masks inside the *Checkout* sub-module.

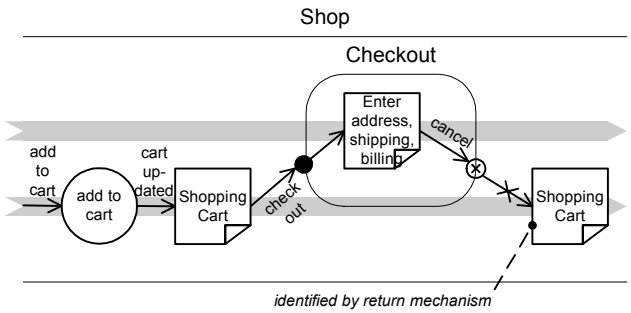


Fig. 7. Return mechanism

Common Events and Abort Mechanism. In some situations, however, it may actually be desirable that certain events can be handled even if their receiver is not specified in the compound that they are generated in – for example, the *create new account* module may be reachable from anywhere within a hypertext-based application, not just from the *Login* mask. To model these relationships, the DFN provides the *common event*. Similar to the compound event, it is symbolized by an arrow leading away from the compound's contour, but outward to another compound element (and only to a compound – it may not lead to an atomic element or into a dialog graph). This so-called *common compound* is nested into the user's dialog sequence wherever he generates the respective common event, independently of his current position in the application.

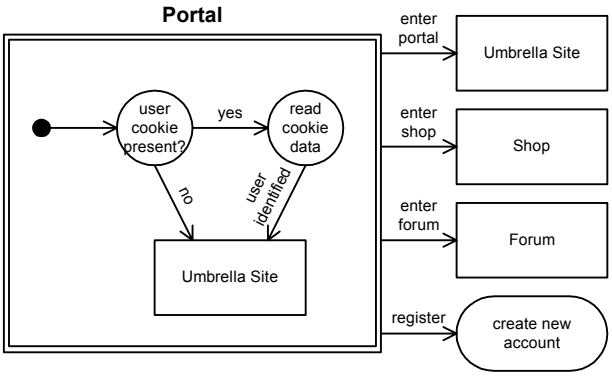


Fig. 8. *Portal* application container

As an example, consider the *Portal* application container in Fig. 8 (the *application container*, symbolized by a double-line box, is the root of the compound's nesting hierarchy, where every user's dialog sequence starts when he enters the application).

The various parts of this portal system are modeled as common compounds so they can be reached from anywhere within the application.

As with compound events, we need to consider how to return from a common compound. For common modules, we can simply use the return mechanism that leads the user back to the dialog mask that called the module. However, common containers pose more of a challenge. Since they do not terminate by themselves, and nesting them deeper and deeper into each other as the user navigates between them would gradually lock up memory, the only option is to abort a common container before another one can be activated at the same nesting level. For example, if the user is currently in the *Shop* container and generates an *enter portal* event, traversal of the *Shop* container's interior dialog graph (and of all compounds nested into it at the time) has to be aborted before the *Umbrella Site* container's initial event can be handled.

In order to abort a compound in a controlled way, a special *abort dialog graph* can be specified for it, which might ask the user if he really wants to abort (also giving him a chance to resume the original dialog graph where he left off), or if he wants to save any unsaved data before aborting. Traversal of the abort dialog graph, which may not contain any sub-compounds and must not be connected to the compound's regular dialog graph, starts at the *abort event* (see symbol in Table 1) and ends at a *cancelled* terminal event. For example, in the *Shop* container's abort dialog graph (Fig. 6), the system prompts the user if he wants to save the items in his cart before leaving, or if he wants to resume shopping. Fig. 9 shows a dialog sequence using the abort mechanism to switch from the *Shop* to the *Umbrella Site* container.

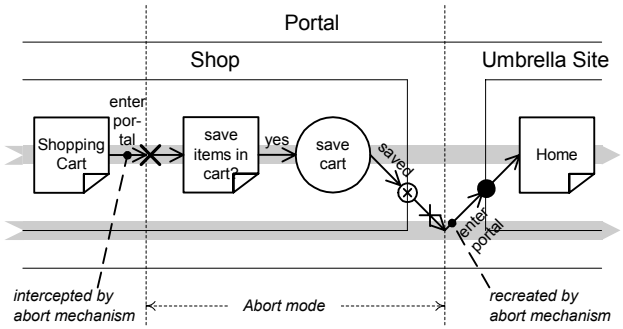


Fig. 9. Abort mechanism

In case the user decides not to switch containers, he can generate a *resume event* (symbolized in dialog graph diagrams by an arrow leading towards the compound's contour), which invokes the *resume mechanism*. Using an algorithm similar to the return mechanism, it leads the user back to the dialog mask in the regular dialog graph that was last displayed before the abort sequence started.

Presentation Channels. The notation constructs introduced so far allow developers to specify complex, hierarchical dialog flows. However, we still need a way to specify

the presentation channel-dependent dialog flows required for different client devices, as illustrated in Fig. 2. In the DFN, this can be achieved by specifying the dialog flows for different media in separate dialog compounds and adding the *channel labels* of the respective presentation channels in square brackets after the compounds' name.

For example, Fig. 10 specifies the dialog flows for a *Checkout* module on the HTML and WML presentation channel. Note that while the channels employ different dialog masks according to the clients' input/output capabilities, they use the same actions for processing the user's input, as indicated by the shading. This enables developers to implement the device-independent application logic only once and then reuse it for multiple presentation channels. Provided that the actions were designed with sufficient granularity, further channels can be added to an application just by implementing the respective masks and specifying the new channels' dialog flows.

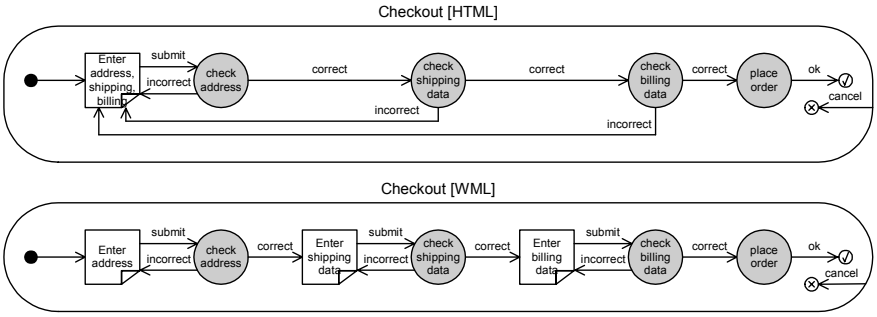


Fig. 10. *Checkout* module on HTML and WML presentation channel

This concludes our presentation of all notation elements. While their semantics were not described formally here, the implementation of the Dialog Control Framework (section 4) defines operational semantics for all constructs.

3.2 Dialog Flow Specification Language

After the dialog flows of an application have been specified in dialog graph diagrams, an efficient transition from specification to implementation is desirable: The dialog graph diagrams should not just visualize the dialog flow, still requiring developers to implement the appropriate dialog control manually, but should rather serve as direct input for the dialog control logic, instructing it how to handle events.

To achieve this, the graphical specifications must first be transformed into a machine-readable representation that can be parsed by the dialog control logic. We therefore introduce the *Dialog Flow Specification Language (DFSL)*, an XML-based language consisting of elements that correspond to the DFN's dialog elements, events and constructs. A complete dialog flow specification consists of two documents – a *dialog flows document* containing a textual representation of the dialog graphs, and a *dialog elements document* mapping dialog elements to their implementation (Fig. 11).

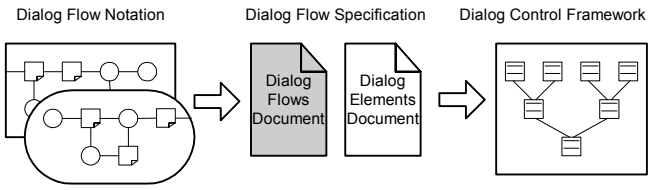


Fig. 11. Transition from specification to implementation

4 Dialog Control Framework

The dialog control logic that reads the DFSL documents and manages the dialog flow accordingly is application-independent. Therefore, we implemented it in a Dialog Control Framework that can be reused for any hypertext-based application and presentation channel. Hypertext-based applications are usually designed according to the Model-View-Controller (MVC) paradigm [18], which suggests the separation of user interface, application logic and control logic. While user interface and application logic can be distinguished quite naturally ("what the user sees" vs. "what the system does"), the distinction between application logic and dialog control logic is much more subtle ("what the system does" vs. "what it should do next, based on the user's input"). Therefore, it is easy to mix up the implementation of application and dialog control logic, even if both are separated well from the presentation logic.

4.1 Struts: Decentralized Dialog Control

For example, in the Apache Jakarta Struts framework [1], the dialog flow is controlled by so-called `Action` objects. Fig. 12 shows how these handle each request:

1. A request comes in from the client.
2. The Controller dispatches the request to the responsible `Action` object, as defined by the action mappings read earlier from a configuration file.
3. The `Action` performs some application logic, either by itself or by calling a subsystem that does the actual work. In the process, the Model data is updated.
4. Based on the outcome of the application logic operation, the `Action` object determines how to proceed in the dialog flow and indicates to the Controller which View should generate the response.
5. The Controller forwards the request to the View indicated by the `Action`.
6. The View generates the response using application data extracted from the Model.
7. The response is sent back to the client.

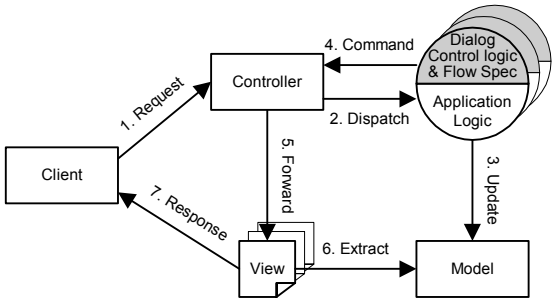


Fig. 12. Coarse architecture of the Struts framework

As indicated by the shading in the figure, the dialog control logic is distributed over all actions in the Struts approach, i.e. the dialog flow is not specified outside the application, but actually implemented in the Java code of the `Action` objects.

This allows the actions to make only relatively isolated dialog flow decisions, and hampers the implementation of more complex dialog structures with constructs like nested dialog modules. To raise the actions awareness of the bi g picture and enable them to control more complex constructs, still more control logic would have to be implemented in them, exacerbating the problem. Also, the hard-coded decentralized implementation of the dialog control logic is relatively inflexible, almost unsuitable for reuse and hard to maintain. Finally, achieving presentation channel independence would require additional effort and possibly redundant work: Since the dialog flow depends on the presentation channel, while the application logic does not, their close coupling prevents the reuse of actions on multiple presentation channels. Instead, each presentation channel would require its own set of `Action` objects to implement the individual dialog flow for the respective devices.

4.2 DCF: Centralized Dialog Control

In contrast, the Dialog Control Framework (DCF) presented in this paper features a very strict implementation of the MVC pattern, completely separating not only the application logic and user interface, but also the dialog flow specification and dialog control logic: The controller decides where to forward requests by using a central dialog flow model to look up the receivers of events generated by masks and actions [25]. This dialog flow model is an object structure that is not hard-coded anywhere, but constructed automatically from the parsed DFSL documents upon initialization of the framework (Fig. 13).

As the coarse architecture shows, the actions are relatively lightweight here since they contain only application logic, while all dialog control logic has been moved to the *dialog controller*. This controller does not receive requests from the clients directly anymore. Instead, on each presentation channel, it receives events that have been extracted from the requests by *channel servlets*. The dialog controller looks up the receivers for these events in the *dialog flow model* a collection of objects representing dialog elements that hold references to each other to mirror the dialog flow.

This dialog flow model is built upon initialization of the framework by parsing the DFSL documents containing the dialog flow specification (the shaded parts of the diagram emphasize that the dialog control logic and the flow specification are decoupled from the application logic and from each other in this approach). Depending on the receiver that the controller retrieved from the model for an event, it may call an action, forward the request to a mask, nest or terminate compounds. The latter operations are performed on *compound stacks*, which store the nested compounds that constitute the state of the dialog system for each user. We refer to this design pattern as MVC+D (Model-View-Controller plus Dialog Flows).

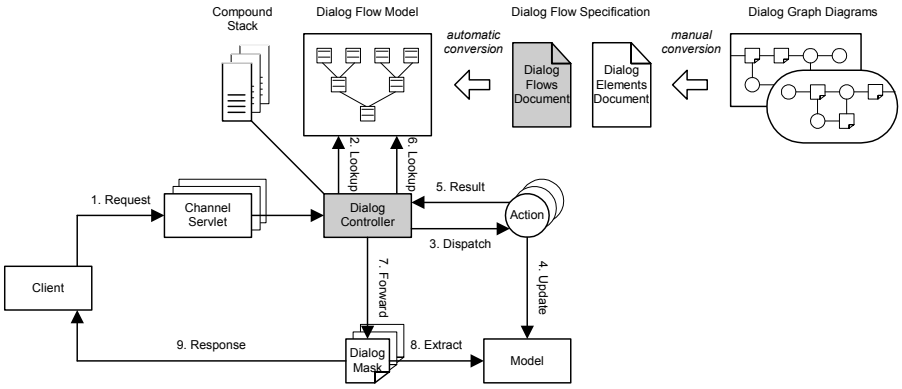


Fig. 13. Coarse architecture of the Dialog Control Framework

In each dialog step, these components work together as follows:

1. A client request with an encoded event is received by a channel servlet, which decodes the event and sends it to the dialog controller.
2. The dialog controller refers to the dialog flow model to look up how to handle this event in the current dialog system state, as stored on the user's compound stack.
3. If an action shall handle the event, it is invoked and the event passed on to it (if a mask shall handle the event, the system proceeds with step 7 instead).
4. The action performs some application logic, either by itself or by calling a subsystem that does the actual work. In the process, the Model data is updated.
5. Based on the outcome of the application logic operation, the action generates a new event and returns it to the dialog controller.
6. The dialog controller refers to the dialog flow model to look up how to handle this event in the current dialog system state, as stored on the user's compound stack.
7. If a mask shall handle the event, the request is forwarded to it (if another action shall handle the event, the system proceeds with step 3 instead).
8. The mask generates the response using application data extracted from the Model.
9. The response is sent back to the client.

For easier comparison with the Struts approach, events involving compounds were not shown in the above sequence. If compounds have to be activated or terminated, the dialog controller would push them onto or retrieve them from the user's compound stack and then look up the next event in the dialog flow model.

This centralized dialog control solution has three advantages over the previously discussed approach:

- The strict separation between application logic implementation, user interface design, dialog flow specification and dialog control logic enables a high degree of flexibility, reusability and maintainability for the components of all four tiers.
- Due to this clean separation, presentation channel-independent applications can be built with minimal redundancy: Only the dialog masks and the dialog flow specifications for the different channels have to be adapted, while the application logic is implemented only once.
- Finally, since the central dialog control logic is aware of the whole dialog flow specified for a channel (it knows the "big picture"), it can provide mechanisms for the realization of complex dialog constructs. Thus, the application developer can use context-independent dialog modules that may be nested, aborted and resumed without having to deal with states, stacks and resume point identification.

To build an application with this framework, the developer does not need to know about the inner structure or implementation of the framework. He only needs to provide subclasses of an `ActionImpl` class implementing the actions, JavaServer Pages implementing the dialog masks, DFSL documents specifying the dialog flow and mapping elements to their implementing entities, and if required, channel servlets for various presentation channels (the prototype framework we implemented already provides `HTMLChannel` and `WMLChannel` servlets). Since these deliverables are completely application-specific, the framework is suitable for black box reuse, giving developers a high degree of flexibility and convenience in building their application.

The authors implemented a prototype of the Dialog Control Framework employing the Java 2 Enterprise Edition. The Dialog Flow Notation elements, events and dialog graph constructs were modeled in a class structure making heavy use of generalization, overwriting and overloading techniques to achieve modularity, extensibility and device independence. To validate the suitability of the Dialog Flow Notation, Dialog Flow Specification Language and Dialog Control Framework for practical use, a demo application that employs all dialog control features was developed at the Chair of Applied Telematics / Mobile Technology Lab. The "Travel Planner" application provides users a front-end for scheduling trips (including reservations for transport and accommodation) that can be accessed via a desktop web browser or a WAP-enabled mobile device. Its development covered all phases from the specification of the dialog flows via their translation into DFSL documents to the framework-based implementation of the application.

5 Conclusions

This paper discussed two challenges brought about by mobile business processes: Firstly, the need to specify the distribution of processes across several sites, and sec-

ondly, the need to specify the dialog flows of the applications implementing those processes on mobile devices. It then gave an overview of the Process Landscaping method with its support for refining processes across multiple abstraction layers and associating their activities and objects with distinguished locations. Next, it presented a Dialog Flow Notation and Dialog Control Framework for the specification and management of complex dialog flows in hypertext-based applications.

Introducing the MVC+D pattern, the framework not only strictly distinguishes application logic, user interface and dialog control, but also separates the dialog control logic from the dialog flow specification. The associated notation is essential for providing the specification of the dialog flow to the framework. Since it does not require a detailed knowledge of the underlying protocols and technologies, but instead works with three relatively intuitively understandable concepts (masks contain what the user sees, actions contain what the system does, and compounds contain transactions the user can perform), it can also be used by people without programming experience, such as representatives of the applications target audience, usability experts and user interface designers. Therefore, the notations dialog graph diagrams can be used as a communication tool throughout the software development process. The graphical specifications can be transformed into DFSL documents according to simple rules, allowing for an efficient transition from specification to implementation.

A weak point of the notation might be the fine granularity of actions that is required to employ them flexibly on different presentation channels (this especially concerns actions responsible for processing user input submitted through forms): The finer the actions are grained, the easier it is to adapt to different interaction patterns – however, very fine granularity also results in quite high specification, implementation and performance overhead. When specifying an application, the developer therefore needs to find a balance between the desired flexibility and the required granularity, while being aware that if the granularity is not fine enough, it may be difficult to add more presentation channels to an existing application in the future. Research on solutions to this dilemma is currently in progress.

Another issue that is a current focus of our research is the frameworks robustness and error tolerance. When encountering events that cannot be handled, a graceful degradation is the minimum requirement. There are a number of ways in which an event-driven system might react in this case [11], for example by ignoring the event or by reestablishing a clearly defined state. In some situations, however, a more user-friendly reaction would be desirable – most importantly, when the user employs the clients backtracking feature. On the Web, clicking the browsers back button is the second most frequent user activity after clicking on a link [4]. It should therefore not be dismissed as a rare and exceptional activity that can be neglected by the dialog control logic, but rather be regarded as a normal interaction pattern that the application must be able to handle as well as regular clicks on links. Backtracking in a hypertext-based application is similar, but not equivalent to the *undo* feature of traditional applications: While a traditional *undo* aims to reverse a previous application operation, backtracking aims to revisit a previous dialog mask without changing the applications data model. This is a challenge – since the user events that are recreated through backtracking often lead to actions, which perform application-logic operations before the dialog step finally completes with displaying a mask.

Finally, more empiric research is needed to see how the Dialog Flow Notation and Dialog Control Framework can be integrated into the software development process

for hypertext-based applications. Experiences gained from larger projects should also yield insights into possible limitations of both tools in certain application domains or on certain presentation channels.

References

1. Apache Jakarta Project: Struts. *http://jakarta.apache.org/struts/*
2. Ball, T., Colby, C., Danielsen, P., Jagadeesan, L.J., Jagadeesan, R., Lūfer, K., Mataga, P., Rehor, K.: Sisl: Several Interfaces, Single Logic. *International Journal of Speech Technology 3,* 2 (June 2000), 91-106. Kluwer Academic Publishers, 2000
3. Butler, M., Giannetti, F., Gimson, R., Wiley, T.: Device Independence and the Web. *IEEE Computing 6,* 5 (Sep.-Oct. 2002), 81-86
4. Catledge, L.D., Pitkow, J.E.: Characterizing Browsing Strategies in the World Wide Web. *Computer Networks and ISDN Systems 27,* 1065-1073. Elsevier Science, 1995
5. Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. *Computer Networks 33* (June 2000), 137-157
6. Doberkat, E.E.: A Language for Specifying Hyperdocuments. *Software - Concepts and Tools 17,* 1996, 163-172
7. Fraternali, P., Paolini, P.: Model-Driven Development of Web Applications: The Autoweb System. *ACM Transactions on Information Systems 28,* 4 (Oct. 2000), 323-382
8. Fraternali, P.: Tools and Approaches for Developing Data-Intensive Web Applications: A Survey. *ACM Computing Surveys 31,* 3 (Sep. 1999), 227-263
9. Gaedke, M., Beigl, M., Gellersen, H.-W., Segor, C.: Web Content Delivery to Heterogeneous Mobile Platforms. Advances in Database Technologies, *Lecture Notes in Computer Science 1552.* Springer, 1998
10. Graw, G., Gruhn, V.: Process Management In-the-Many. 4th European Workshop on Software Process Technology (EWSPT 95), *Lecture Notes in Computer Science 913,* 163-178. Springer, 1995
11. Green, M.: A Survey of Three Dialogue Models. *ACM Transactions on Graphics 5,* 3 (July 1986), 244-275
12. Gruhn, V., Wellen, U.: Support for Distributed Business Processes. *Asia-Pacific Software Engineering Conference (APSEC99),* Takamatsu, Japan, December 7-10, 1999
13. Gruhn, V., Wellen, U.: Structuring Complex Software Processes by Process Landscaping. European Workshop on Software Process Technology (EWSPT 2000), *Lecture Notes in Computer Science 1780.* Springer, 2000
14. Gruhn, V., Wellen, U.: Process Landscaping: Modelling Distributed Processes and Proving Properties of Distributed Process Models. *Lecture Notes in Computer Science 2128,* 103ff. Springer, 2001
15. Harel, D.: Statecharts: A visual formalism for complex systems. *Scientific Computer Programming 8,* 3, 231-274
16. Isakowitz, T., Stohr, E. A., Balasubramanian, P.: RMM: a methodology for structured hypermedia design. *Communications of the ACM 38,* 8 (Aug. 1995), 34-44
17. International Organization for Standardization: Ergonomic requirements for office work with visual display terminals (VDTs) Part 10: Dialogue principles. *ISO 9241-10,* 1996
18. Krasner, G.E.: A Cookbook for using the Model-View-Controller User Interface Paradigm in Smalltalk. *Journal of Object-Oriented Programming 1,* 3 (1988), 26-49
19. Lupu, E.C., Sloman, M.: Conflicts in Policy-Based Distributed Systems Management. *IEEE Transactions on Software Engineering 25,* 6, 852-869. IEEE Computer Society Press, 1999

20. Nuseibeh, B., Kramer, J., Finkelstein, A., Leonhardt, U.: Decentralized Process Modelling. 4th European Workshop on Software Process Technology (EWSPT 95), *Lecture Notes in Computer Science 913,* 185-188. Springer, 1995
21. Rice, J., Farquhar, A., Piernot, P, Gruber, T.: Using the web instead of a window system. *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '96).* ACM Press, 1996
22. Shaw, M.J.: Electronic Commerce: State of the Art. Shaw, M., Blanning, R., Stader, T., Whinston, A. (eds.): *Handbook on Electronic Commerce,* 3-24. Springer, 2000
23. Sinha, A.: Client-Server Computing. *Communications of the ACM 35,* 7 (July 1992), 77-98
24. Schwabe, D., Rossi, G.: The object-oriented hypermedia design model. *Communications of the ACM 38,* 8 (Aug. 1995), 45-46
25. Singh, I., Stearns, B., Johnson, M., et al.: *Designing Enterprise Applications with the J2EE Platform, 2nd Edition.* Addison-Wesley, 2002
26. Zhao, W., Kearney, D., Gioiosa, G.: Architectures for Web Based Applications. 4th Australasian Workshop on Software and Systems Architectures (AWSA 2002), Feb. 2002, *http://www.dstc.monash.edu.au/awsa2002/papers/Zhao.pdf*

A Distributed Communication System for Integration with Logistics Software

V. Gruhn[1], M. Hülder[1], R. Ijioui[1], and L. Schöpe[2]

[1] Chair of Applied Telematics / e-Business*, Computer Science Faculty, University of Leipzig, Germany
{gruhn, huelder, ijioui}@ebus.informatik.uni-leipzig.de
[2] Informatik Centrum Dortmund e.V., Abt. Softwaretechnik,
lothar.schoepe@icd.de

Abstract. Truckage companies need continuous and up-to-date information about their business processes in order to respond quickly to customers' needs and problems emerging during transport processes. Therefore a reliable and user-friendly communication system is required, which improves the relationship between drivers and dispatchers. The project "Mobile Spedition im Web (SpiW)**" presented here, develops a mobile communication system, which focuses on the driver/dispatcher interaction. The main goals are integration with legacy logistics software and the possible use of new telematics and communication techniques. To achieve these goals, a component based architecture allows the later change and extension of components, making it possible to add new features to the system as they become available. A distributed workflow server supports the adjustment of business processes to individual needs.

1 Introduction

Truckage companies take advantage in business, when they can perform transports fast, securely, economically, and in time. "Time more and more becomes a critical component in freight transportation" [EW97]. This advantage is even more important, as due to globalisation and the extension of the European Union the number of truckage companies rises. Truckage companies that can achieve the named goals more efficiently due to the use of mobile communication can gain more trust as well as a better customer relationship.

1.1 Communication Problems

According to [EK01] the following problems can appear during communication and cooperation between the different roles within a truckage company (drivers, dispatchers, customers) and thus stand in the way of reaching the required goals.

Problems for the dispatcher:

* The chair for Applied Telematics / e-Business is endowed by Deutsche Telekom AG.
** Supported by the German Ministry of Education and Research (reference no. 01HT0143)

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 134–143, 2003.

- Discontinuous, oral information interchange between dispatcher and drivers leads to delays and mistakes.
- From the company's point of view drivers are a main source for information, but the information pooled at a driver cannot be transmitted into the logistics software without further manual work.
- Because of missing knowledge abut the transport progress the dispatcher can hardly reschedule transports.
- Calculations of transport costs can only be performed with a great delay.

Problems for drivers:

- Drivers can communicate problems only by mobile phone most of the time.
- Drivers have little influence on the scheduled tours and possible rescheduling.
- Data transmitted by the papers is often incomplete or even wrong.
- Dispatchers may not be available for questions.

Problems for customers:

- Transport progress is not known to the customer.
- Delays are not to be calculated.

By developing from location based acquisition towards mobile acquisition and transmission of information and data, important information can be made available in time. This information may provide solutions to the named problems, and these solutions may not only help for the dispatching processes but also for the strategic fleet management.
Ideally such a communication system ought to provide a generic interface to logistics software systems and thus provide the advantage of being integrated into different logistics software systems. That way a clear cut between logistics software and communication system is made, proving the opportunity for truckage companies to extend their logistics software system rather than investing into a new monolithic system.

1.2 Usage Scenario

In order to plan the transports the dispatcher makes use of a logistics software system. To communicate with the drivers, the dispatcher usually uses paper forms or telephones. Using these means of communication exclusively can lead to loss of information or delays which prevent the dispatcher from reacting appropriately to events that happen to the drivers. The communication system introduced in this paper helps to overcome these problems. The communication system described in the following chapters is based on the following scenario (see Fig. 1): The dispatcher asks the driver to perform a transport. The driver then loads the freight and delivers it to the customer. This scenario differs from other parcel or express services in so far as there are very few private people but rather business customers involved who typically get freight with several tons of weight delivered.

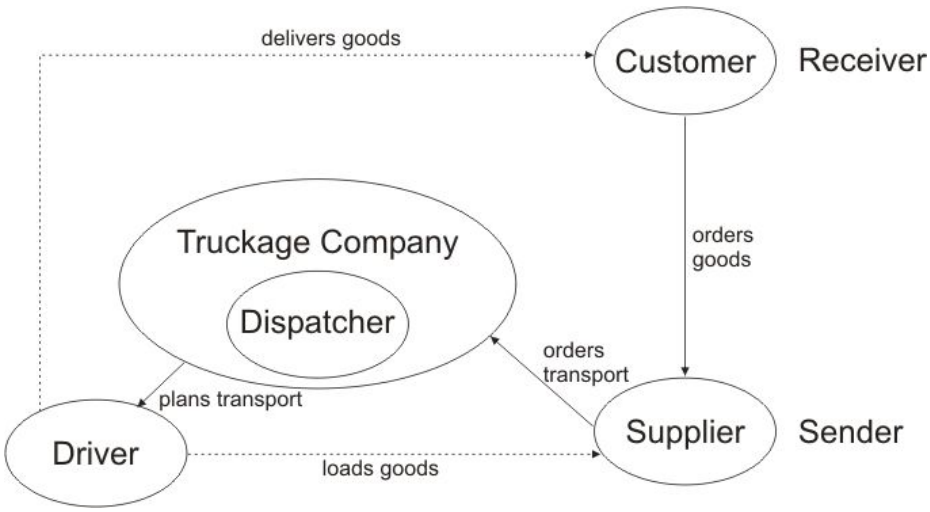


Fig. 1. Roles in the usage scenario

To support the communication between drivers and dispatchers, drivers are provided with mobile devices (e.g. PDAs). On one hand these devices can inform the drivers about the scheduled transports and on the other hand they enable the drivers to report back about each transport's status and problems that might appear during the delivery.

2 Architecture of the Communication System

The communication system's architecture consists of three components: mobile devices (e.g. PDAs), stationary devices (e.g. PCs), and an application server. The mobile devices connect to the application server via a wireless telecommunication system (GSM, GPRS, HSCSD, EDGE, UMTS), whereas the stationary devices use a wired connection (Ethernet, FastEthernet) to connect to the server.
The software architecture of the communication system follows the client/server paradigm [Lew98]. The business logic for working on business objects is provided by an application server [BG98], which itself takes advantage of other server components: a workflow server, a communication server, and a database server (see Fig. 2). The services provided by the server according to the business processes are used by the clients to deliver data to the targeted user.
The kind of data that is supplied by different clients may differ to a great extend, according to the users' roles and needs. While a driver mainly needs transport data, the dispatcher needs to have not only transport data but also the appropriate managing data as well. Depending on the kind of device used and its particular technical possibilities (e.g. display size) the different clients do not

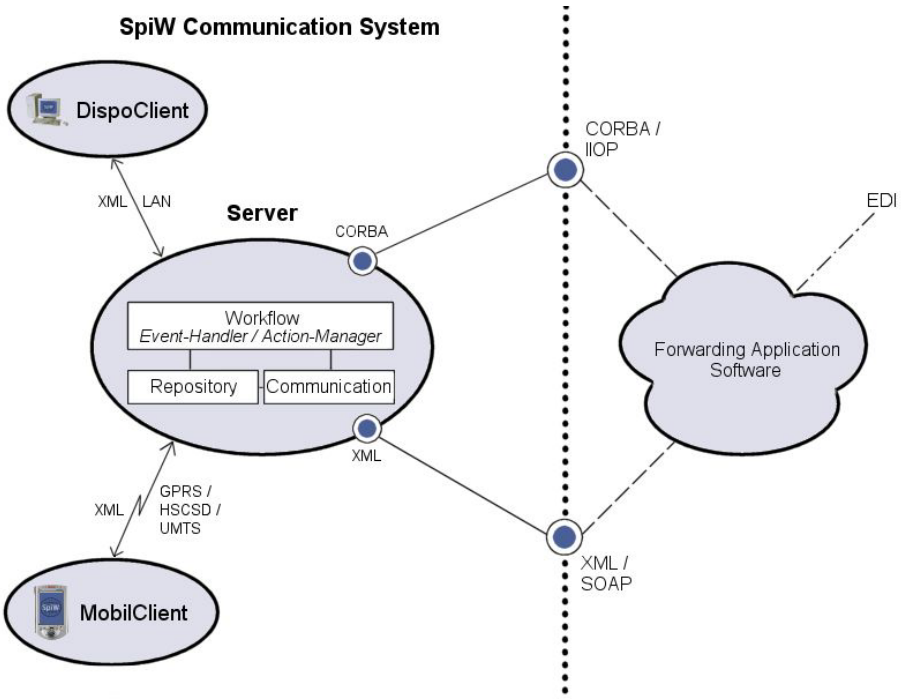


Fig. 2. Architecture

only render the data differently but also the amount of data displayed is adjusted. Similarly the usage of the user interface is realised in a different way (e.g. preconfigured hardware keys on PDA or mouse control on a PC).

2.1 Thin Vs. Thick Clients

If business logic is not only supplied by an application server, but parts of the business logic are realised on a client, clients are called "thick clients" [Lew98]. On the other hand so-called "thin clients" (e.g. web browsers) do not have business logic on their own, but exclusively use services provided by an application server. Although thin clients generally need fewer resources, and therefore appear to be especially suited for devices with limited memory and processing power, thick clients are used on the mobile devices. This is due to the fact that a wireless connection cannot be guaranteed to be available or even stable. But to take care of the requirements mentioned in the introduction, parts of the business logic need to be executed even when the communication link is temporarily unavailable. After re-establishing the connection, data synchronization has to take place.

It is rather unlikely that the communication link over the wired medium in a local area network (LAN) breaks down for a long time, so that this line of argument is not valid for stationary devices. Still, a thick client is used for the stationary devices, for this client being the logistics management software which contains the complete business logic for rendering and working with the data. For more information on the advantages and disadvantages of thin and thick clients [Lew98] and [OE96] can be named.

2.2 Data Exchange with Legacy Software

The communication system defines interfaces for exchanging data with logistics management software. The data structure for the data exchange between clients and application server is described by a Document Type Definition (DTD) [Tol99, BUvE99] and transmitted according to the XML-format. For this purpose attributes of objects need to be transformed into XML data and then sent over the communication link. During the transmission additional compression and cryptographic techniques may be used. The receiver then has to search the transmitted XML-document (parsing) and reproduce copies of the original data from the structure and the contents of the document.
This process is necessary because the development of application server and clients is based on a component model [GT98, Szy98] used in conjunction with an object-oriented programming language. Due to this, a later extension of the system does not require the data transmission part to be developed again, because only the required classes and the extended DTD have to be deployed on the system. According to the object-oriented paradigm the parser itself can use the objects' methods to produce an XML-code representation of the objects and vice versa reconstruct an object from the XML representation.

3 Distributed Workflow Support

Business processes that are to be supported by the communication system are described by so-called workflows. A workflow consists of a number of single actions, which can be simple or complex in themselves again. The actions of a workflow can be carried out in either a sequential, or parallel way, or as alternatives to each other. Workflows can be connected, i.e. initiate or depend on one another.
For any component in the architecture (see Fig. 3) there are one or more workflows which deal with the creation, manipulation, and visualisation of business objects that are connected with it.
According to dependencies between components, workflows of different components may also be dependent on each other. Such dependencies are not always fixed but can evolve after creation or modification of data. Any class of a component contains a so-called `display`-method which is used to visualize data of that class according to a style-guide and the user interface. This way it is made sure that rendering of data is consistently done in the same way and the user is directed to his goal in the same manner.

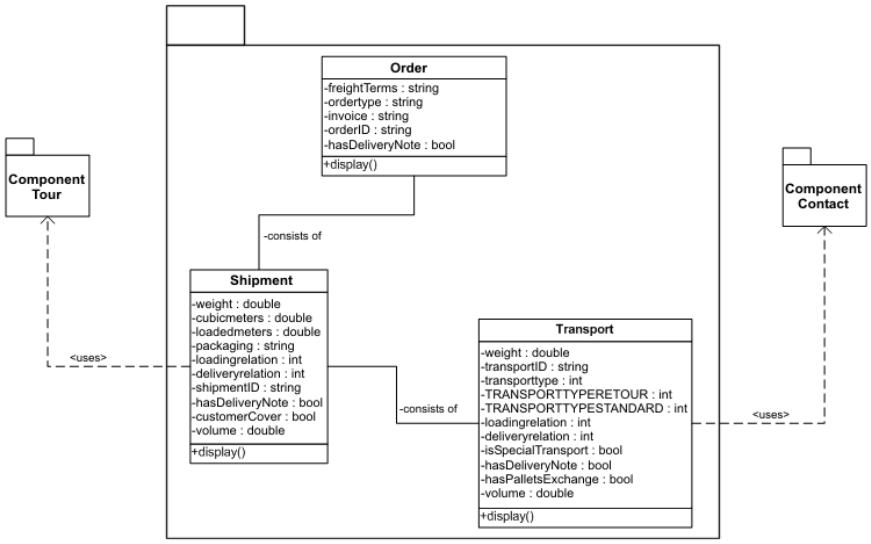


Fig. 3. Components of the communication system

In addition to the global workflow which describes business processes, there is local workflow which describes method collaboration within a class. This local workflow can be realised by implementing the display-method of a class.
Any component of the architecture consists of several classes. The components themselves possess a specified interface. The components shield data and store it permanently in a database. Realisation of components is done using an object-oriented programming language (such as Java or C#). Components are associated with actions of a workflow (Fig. 4). In general there should only be one component associated with each action, but in exceptional cases there can be an association with several components. Such a complex association may only be allowed if changing the workflow is not possible or does not solve the problem. In this case workflow between associated components has to be described explicitly.
If there is a decision to be made depending on the data during the course of an action which defines the following course of the action, there has to be a decision table (Fig. 5). The course of workflows is controlled by a workflow server which is part of the application server. Due to the fact that parts of the business processes need to happen in a mobile and therefore distributed environment on the clients as well as on the application server the workflow server has to support the distributed execution and is responsible for data consistency and integrity. Distributed execution may be achieved by either a centralised or a decentralised approach. Considering the wireless connection between mobile clients and application server, and the problems originating from this (loss of connection, network

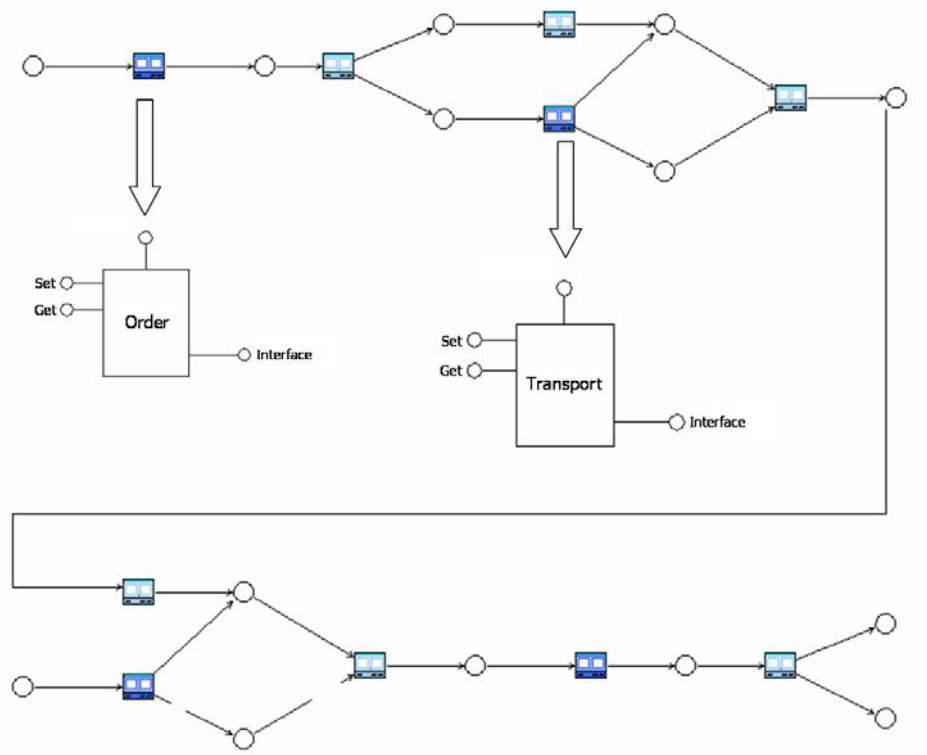


Fig. 4. Realised and associated components

availability, etc.), a decentralised solution appears to be suitable. That means that there is a central workflow server available as well as local workflow servers on each of the clients, although they may differ in the amount of functionality they provide.

Depending on the environment and equipment of the mobile devices the mobile workflows have to be adjustable. For example it is possible to attach a barcode scanner to the mobile device. In this case, the workflow has to handle scanner and read barcodes appropriately, e.g. when delivering goods. In another case, when there is no barcode scanner installed, the delivery process has to leave out the scanning and proceed accordingly, e.g. prompting the driver to confirm that he has delivered the appropriate goods. This mechanism of modelling different workflows for different user and device profiles, and then executing the workflows in the workflow server, allows for fine grained adjustment of the software without changing the source-code.

Whether components may be loaded dynamically at runtime from the server according to the workflow executed or whether all components have to be deployed statically on the clients, depends on the communication techniques (HSCSD, GPRS, EDGE, UMTS) available.

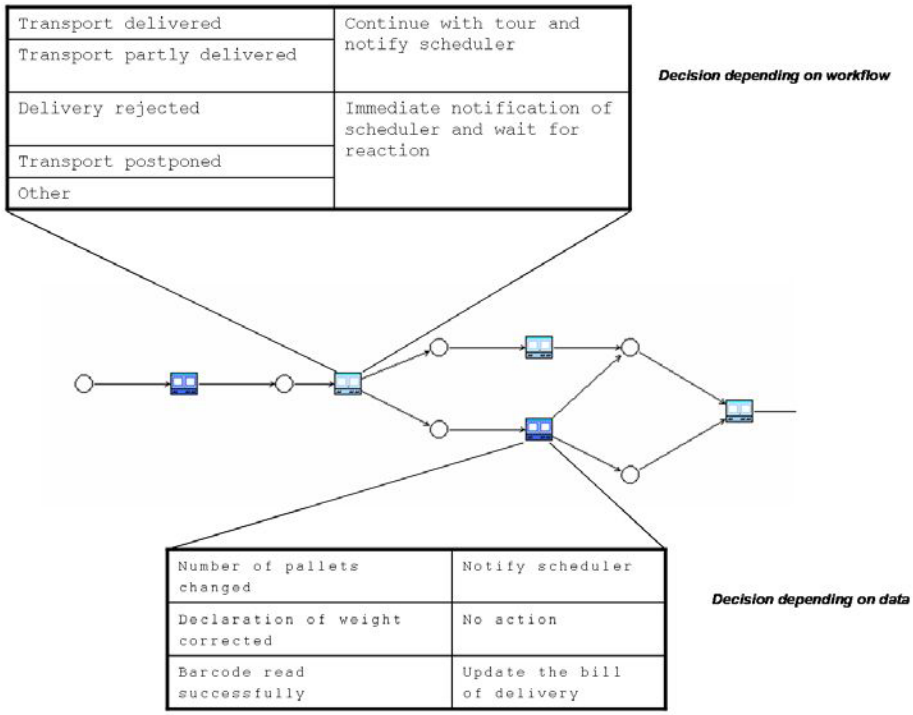


Fig. 5. Decision table for actions

4 Related Work

The complexity of the transportation field is reflected by the richness of research areas, methods, and software. The future of truckage companies, however, lies in providing a more efficient and cost effective transport service. The objective is to use modern technology in order to exploit the full potential of saving costs. More and more goods have to be moved efficiently, quickly and cost effectively by service and transport companies. In order to survive, transport companies must respond quickly to their customers' needs and focus on cost control. Continuous and up-to-date understanding of business processes is essential and requires a reliable and user-friendly system for mobile data communication [Fre03].
Researchers have already developed a number of techniques to solve the communication and fleet management problems of truckage companies. For example, researchers at the University of Bremen proposed a concept for controlling truck fleets using cellphones and the Internet [EK01, SK03]. Economists at the University of Koblenz are also working on a prototype for supporting logistics processes [Jun01]. The main goal of these research efforts is to provide flexible fleet management and introduce new kinds of services that truckage companies could offer in future (e.g. location-based services).

There is no universal solution that is suitable for any truckage company, and the systems available today still suffer from a few weaknesses.

5 Conclusion

Electronic support of transport execution is barely integrated with conveyance systems. Communication during the transport process and afterwards is done either on paper documents or by telephone, both of which require additional work to receive an electronic representation later on. This additional work is not only time consuming but also error prone. The scheduler is not informed well enough about the progress of the transport, and needs to acquire additional information actively himself.
Therefore we see the need for bidirectional communication to exchange transport information in time on one hand, and on the other hand unidirectional communication between mobile devices and other backend software systems to exchange data over a wireless yet stable medium.
Most of the systems available today do not focus on the need of small conveyance companies. Those companies often cannot invest into a completely new software system. Employees would have to get used to the new software and the market situation is so unclear that no one can say which systems are going to last for enough time to secure the investments.
Systems that are based on application service providing concepts put up another barrier, because they require hosting vital company data about customers and lorries on the service provider's side. Thus the company becomes dependent on a third party's accessibility and reliability.
The project "Mobile Spedition im Web (SpiW)" which is supported by the German Ministry of Education and Research, aims to reach the two main goals of integration with legacy logistics software and the possible use of new telematics and communication techniques. The component based architecture of the communication system allows the later change and extension of components, making it possible to add new features to the system, as they become available (such as transmission of video data or data gained from board sensors).
Within the project consortium it is not possible to reach these goals completely, especially for the integration with legacy software the interface defined has to be supported by the legacy software developers.
Although the benefit of such a communication system is obvious, it also depends on the costs of acquisition and operation which is even more important. The industrial partners in the project consortium are to ensure that the benefit exceeds those costs.

References

[BG98] S. Baker and R. Geraghty. Java For Business Objects. In Andy Carmichel, editor, *SIGS*, pages 225–237. Cambridge University Press, 1998.

[BUvE99] U. Böhnlein and A. Ulbrich vom Ende. XML Extensible Markup Language. *Wirtschaftsinformatik*, 41(3):275–277, 1999.

[EK01] E. Erkens and H. Kopfer. WAP-LOG: Ein System zur mobilen Fahrzeugeinsatzsteuerung und Auftragsfortschrittkontrolle. pages 293–303. Teubner Verlag, Stuttgart, 2001.

[EW97] M. Ernst and D. Walpukis. *Telekommunikation und Verkehr*. Verlag Franz Vahlen, München, 1997.

[Fre03] J. Frediani. fleet management via the internet, 2003. http://www.daimlerchrysler.de/environ/report2001/pdf/fleet_e.pdf (07.03.2003).

[GT98] V. Gruhn and A. Thiel. *Komponentenmodelle*. Addison Wesley, München, 1998.

[Jun01] J. S. Jung. Flottenmanagement im Handwerk durch integrierte Telematikdienste, 2001. http://www.uni-koblenz.de/ flotthit/Project/overview.html (07.03.2003).

[Lew98] S. Lewandowski. Frameworks for Computer-Based Client/Server Computing. *ACM Computing Surveys*, 30(1):3–27, 1998.

[OE96] Harkey D. Orfali, R. and J. Edwards. *The Essential Client/Server Survival Guide*. Wiley Publ., 1996.

[SK03] Erkens E. Siek, K. and H. Kopfer. Marktübersicht über Systeme zur Fahrzeugkommunikation im Stra(ë)engüterverkehr. *ACM Computing Surveys*, 5(1), 2003.

[Szy98] C. Szyperski. *Component Software Beyond Object-Oriented Programming*. Addison-Wesley, Reading, MA, 1998.

[Tol99] R. Tolksdorf. XML und darauf basierende Standards. *Informatik Spektrum*, 22(6):407–421, 1999.

Fault Tolerant Geographical Addressing

Dominic Heutelbeck, Reinhold Räth, and Claus Unger

University of Hagen, Department of Computer Science, 58084 Hagen, Germany
{Dominic.Heutelbeck,Reinhold.Raeth,Claus.Unger}@fernuni-hagen.de

Abstract. Geographical addressing and resource discovery are important building blocks for creating mobile location-aware systems. Mobile systems rely on wireless network technologies, which are prone to frequent failures. In this paper, we present a protocol for maintaining a self-organising routing backbone that supports geographical addressing and resource discovery. The routing backbone detects node failures and automatically repairs its infrastructure, while maintaining the operability of the network at least in partitions an thus maximising service availability. The system operates in a peer-to-peer fashion, i.e. the administrative tasks are distributed between all participating nodes, taking into account their capabilites, location, and mobility. In this paper we explore maintenance aspects of the network that are caused by node failures.

1 Introduction

Today mobile networked devices are part of our everyday life and we can access information and computational power anytime and anywhere. In this scenario, location based services like navigation tools, geographic data bases [4], or even location aware games like Pirates! [1] come into play. Location-aware applications denote an important subset of the more general context-aware applications. A user of such an application may want to discover computational or real-world services in spatial proximity, e.g. may be interested in the next free cab or printer. Other applications may use geographical messaging to send emergency warnings to people located in a region that is about to be flooded. Finding entities in a given region, or sending messages to them is easily identified as a key building block for such applications. We speak about *geographical addressing*, when we are communicating based on location instead of network addresses. When we want to create real world location based applications, we have to deal with an increasing number of technologies connecting mobile devices to the Internet, like WLAN, Bluetooth, GSM, UMTS etc. Additionally, each single technology can be accessed through a large number of different service providers. Wireless connected mobile devices are likely to fail, caused by, e.g. a loss of reception, or, because the user turns them off to preserve battery power. In the described scenario, ad-hoc routing protocols [7] can not directly be applied, as they assume a direct communication channel between nearby nodes. In our scenario, nodes are connected to the Internet through their individual providers. This results in a network, where entities in spatial proximity can be far away from each other in terms of physical network topology.

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 144–155, 2003.

We present a protocol that creates a self-structuring logical network that supports geographical addressing without any central administration. All participating nodes collaborate in a peer-to-peer fashion. The protocol distributes the workload in respect to the context and resources of each participating node. In this paper we emphasise the strategies for detecting and handling failures of participating nodes.

2 Related Work

Imielinski and Navas [6] propose three possible solutions for geographic addressing, routing, and resource discovery. The *geographic routing method* proposes the installation of an infrastructure of *geographic routers*, the *geographic-multicast routing method* hierarchically maps geographical regions and additional context information to multicast groups from the IPV6 address space, and the *domain name server solution* proposes the introduction of a new top-level domain called ".geo" where special name servers map addresses to a set of IP addresses or a temporary multicast groups.

The *content-based networking* scheme proposed by Carzaniga and Wolf [2] presents a more general approach to the problem. It relies on an infrastructure of routers like in the geographic routing method. Instead of using geographical regions, more general predicates are used for addressing, similar to the use of the multicast addresses in the geographic-multicast routing method, but without the limitations of an IP address space. These solutions rely on the installation of a complex infrastructure, or they use a large portion of the multicast address space together with an adjusted multicasting protocol.

Another solution of the resource discovery problem provide central spatio-temporal databases as proposed by Harter et al. [4]. Challenges for such databases are formulated by Sistla et al. [8]. For applications with complex spatio-temporal data sets, these approaches are the right choice and they provide a wide range of possible operations, at the cost of high maintenance costs and a central point of failure.

Resource discovery in highly distributed, heterogeneous networks is one of the core problems addressed in peer-to-peer research. To locate a given resource with an unique identifier on arbitrary hosts, the Gnutella protocol [9] connects arbitrary nodes in a pseudo-random network, where queries are flooded to the hosts. More sophisticated systems like Chord [10] apply the abstraction of a distributed hash table (DHT). Each nodes is assigned to a random identifier and stores the locations of objects whose hash values are close to this identifier. Queries are routed based on these identifiers. These systems address problems different from geographical addressing. In file-sharing systems, queries are used to look up data identified by a *fixed* key, e.g. filename or hash. In our case, the key denotes the constantly changing geographical location of the data. Because a node will most likely query regions in close proximity to itself, the system should be most responsive to such queries. Geographical addressing is more complex than file-sharing.

3 Context Space and Presences

We want to create a system with no central administration that supports two basic services: *Contextual messaging*, i.e. sending a message m to all mobile entities in the target region T. We call such a message a *ContextCast message*. *Contextual resource discovery*, i.e. finding all mobile entities in the region T.

All operation in our system occur in a so-called *global context space* S which defines the set of all possible locations. Let S be the cross product of the bounding intervals $I_i = [l_i, u_i]$, $i \in \{0, \ldots, n-1\}$, $l_i, u_i \in \mathbb{Z}$ and $l_i \leq u_i$, $n \in \mathbb{N}$. One can easily map geographical coordinates to a global context space, which we call a 3-dimensional geographical context space. Services are provided by the *ContextCast network*, where all nodes cooperate in a P2P fashion.

We call mobile entities *presences*. Each presence is an application running on a mobile device with location sensors, e.g. a GPS receiver. Each presence has a name d, a location $l \in S$, a maximal bandwidth α in kB/s, a mobility measure $\beta \in \mathbb{N}$, and a network address consisting of the IP address of the host and a port number. A more general discussion of the concept of context spaces and presences can be found in [5].

4 Structure of a ContextCast Network

We build a distributed network of presences with the following properties:

- It supports contextual messaging and resource discovery.
- It operates without central administration.
- Each presence actively supports the system with its resources.
- It remains consistent under insertion, removal and movement of presences.
- It is robust to node failures.

By assuming that communication between presences in close vicinity are more likely than between presences that are located far apart from each other, our protocol minimises the utilisation of presence resources for messages that are not relevant for the presence. A presence should mostly forward messages that are targeted to regions nearby.

The ContextCast network is based on a recursive decomposition of the context space into a hierarchical structure of clusters. In an n-dimensional context space, each cluster is recursively decomposed into 2^n uniquely labelled subclusters of equal size. Subclusters are addressed via access paths represented by number sequences. For the empty sequence a we set $S_a := S$.

Presences taking over administrational tasks for certain clusters are called *clusterheads*. We aim to assign administrational tasks, i.e. clusterhead roles, to presences for which such a particular task is of high relevance. Therefore a presence may only become clusterhead of clusters in which it is located, but other parameters play an important role as well.

The network is built from successively joining presences. An initial presence takes over the role of the clusterhead in the complete space and registers itself

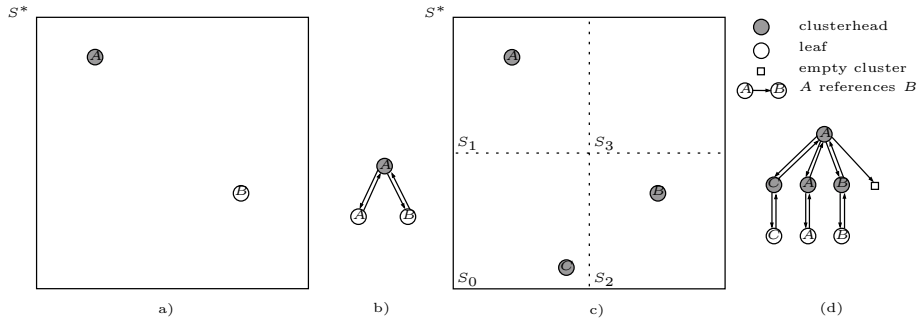


Fig. 1. a) A context space S with two presences A and B. b) The resulting topology of the ContextCast network. c) S after the accedence of a third presence C. d) The network topology after the accedence of C.

as a leaf in the cluster. The topology of the created network is tree shaped, and every presence is at least represented by a leaf in the tree. Each successive presence has to register itself at the clusterhead, and afterwards becomes a leaf in the corresponding cluster. A Clusterhead forwards ContextCast messages and acts as a cache for resource discovery. When the traffic caused by these tasks exceeds the bandwidth of the clusterhead, the cluster is split into subclusters of equal size. The original clusterhead remains the clusterhead of the complete context space, but appoints new clusterheads for the newly created subclusters. Figure 1 shows a possible ContextCast network before and after a split caused by a new presence.

The splitting is continued recursively when new presences are added to the network. The clusterheads act as a routing backbone for ContextCast messages by forwarding messages along the edges of the tree. A presence is at least a leaf in the network and may be clusterhead of several nested clusters. The network is connected by references between topological neighbours in the tree.

5 Maintenance During Normal Operation

In this section we give a brief overview of the ContextCast protocol before we take a closer look at strategies for discovering and recovering from node failures. For a more in depth description of the protocol basics see [5].

5.1 Routing

Figure 2 shows a possible route for a ContextCast message. If a presence F wants to send a message to a target region T, the message is propagated bottom up from the clusterlevel of F. As soon as one of the subclusters of the reached clusterhead intersects with T, the message is propagated down that branch of the network. If that branch does not cover T entirely, the message is passed up again to its parent.

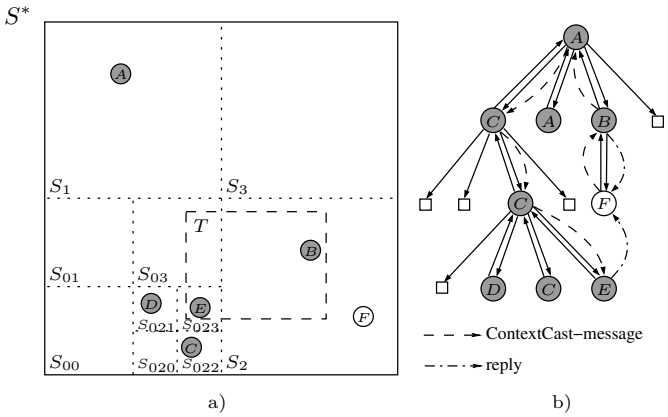


Fig. 2. a) The presence F wishes to send a message to all nodes in the region T. b) The route of the message and replies of the target node. Some leaf nodes are omitted in the graphical representation.

By adding the address of the source presence to the message, other presences may reply directly to the source presence. Contextual resource discovery can be implemented in this way.

5.2 Creating the Network

We use TCP to exchange messages because we need safe communication at some points. This also aids us in detecting node failures. Each presence manages a single queue of incoming messages, which are processed strictly sequentially. Every message contains the address of its original sender.

The context space is instantiated by the first presence. It assumes the clusterhead role for S and waits for other presences to join. If Several presences want to instantiate the context space concurrently, we have to perform a leader election to ensure that we end up with a unique root clusterhead. Classical election algorithms [3] assume that all participating nodes are connected by some topology and at least one other participant of the election is known. In our case, the candidates initially do not know each other.

When a presence A assumes the role of the root clusterhead, it listens to a well known multicast group, and regularly announces itself on this multicast group. In this way, a presence can detect the existence of other root clusterheads for the network. If A detects the existence of another root clusterhead B, and B is better suited for the role, A gives up the root clusterhead role and tells its children to rejoin the network via B. Whether some presence suits better for the root clusterhead role is decided by a heuristic algorithm based on the connectivity and mobility of the presence. Ties are broken by comparing the addresses of the presences. All presences have to use the same algorithm. This

strategy for bootstrapping also aids in merging partitioned ContextCast network, and to handle root clusterhead failures.

In the global Internet, multicast IP is not always available. So for operating in a very large scale, a set of well-known servers that coordinate bootstrapping cannot be avoided. Alternatively, hybrid approaches may be best suited for a bootstrapping problem.

A presence that wants to join an existing ContextCast network needs to know at least one other presence that already is a member of the network. It may get this information by listening to the multicast group or by querying the bootstrapping server.

If A knows such a presence B, it sends a JOIN message to it, containing the location of A. When B receives the JOIN message, the message is hierarchically routed to the clusterhead of the smallest non empty cluster containing the location of A. If a clusterhead on this route notices, that no such cluster exists, A becomes the new clusterhead of that cluster and is notified about it.

When the JOIN message reaches a fitting cluster, the clusterhead H has to decide if its bandwidth is sufficient to handle the additional load. If so, H acknowledges the success of the join operation with a JOINACK message. Otherwise, H splits the cluster into subclusters until the load is distributed over a number of presences or it is not possible to decompose the space any further. The involved presences are notified about the change by a SPLIT message. The split operation maintains two important invariants. Every presence may only be a clusterhead of clusters in which it is located, and it may only be a leaf in such a cluster. A main challenge in the protocol is to sustain these invariants, while the presences are moving arbitrarily.

5.3 Movement

The registered location l of a presence may differ from its actual geographical location. The latter is only updated after the previous update has been completed. The update rate is chosen to fit the application and the jitter of the location sensors. One can identify several basic operations that are necessary to update the network, reflecting the movement of the participating presences.

Movement Inside a Cluster A clusterhead stores the current location of its leafs to be able to send ContextCast messages to the right recipients, to act as a cache for resource discovery, and to split clusters correctly. Therefore, when moving, presences have to inform their clusterhead, i.e. the clusterhead that has registered the presence as a leaf, via a simple handshake consisting of a MOVE and a MOVEACK message. The acknowledgement is necessary for handling split operations during a movement.

Moving from One Cluster to Another To move from one cluster to another, a presence has to leave its original cluster. Then it hands over or closes all clusters it controls that do not contain the target location. Finally it joins a new cluster

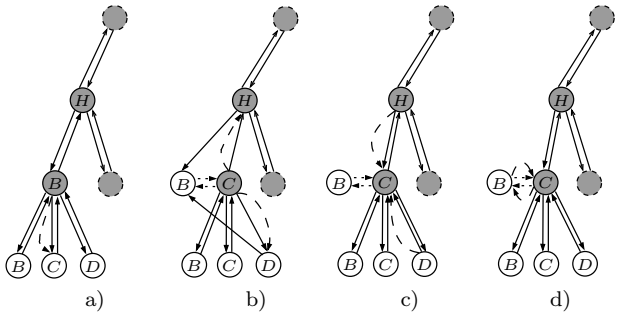


Fig. 3. Topology and message passing in a handover. Sending messages is represented by dashed lines. Dotted lines represent temporary references. a) B sends a HANDOVER message to C. b) C accepts the clusterhead role and sends NEWCLUSTERHEAD messages to the presences that still reference B as the clusterhead. c) The presences H and D register C as the new clusterhead and reply with NEWCLUSTERHEADACK messages. d) C sends a HANDOVERACK message to B, which replies with HANDOVERFIN. C successfully took over the role as the clusterhead.

covering its new location. A presence has to perform similar steps, when it leaves the network completely. In this case, it does not join again.

Leaving or Closing a Cluster To leave a cluster, we use a three way handshake between the presence and its clusterhead. The leaving presence sends a LEAVE message and the clusterhead acknowledges this with a LEAVEACK message. The clusterhead remains in a waiting state, expecting a JOIN message from the presence. The presence sends the JOIN message and sends a LEAVEFIN message to release the clusterhead from its waiting sate. This scheme is chosen to handle race conditions like a concurrent split operation, and to have a suitable presence to send the JOIN message to.

When the presence leaves a cluster in which it is the only presence left, i.e. is the clusterhead, it has to inform its parent clusterhead, that the cluster is now empty. Like while leaving a cluster, three way handshake is used.

Handing over a Cluster The most complex operation is handing over a clusterhead role to a different presence. Remember that for each clusterhead or leaf role a presence stores references to its neighbours in the network. To hand over a cluster, a presence has to select a new clusterhead and to make sure it accepted the clusterhead role and all neighbour presences have become aware of this change.

Figure 3 illustrates how the topology of a ContextCast network changes during a handover. First the presence B sends a HANDOVER message to the new clusterhead C. At this moment, B gives up the clusterhead role and forwards all messages, reaching it as the clusterhead, to C. With the informations embedded

in the HANDOVER message, C takes over the clusterhead role. Now C sends a NEWCLUSTERHEAD message to all neighbour nodes, that are not yet aware of the handover, announcing its new role. These nodes register the new clusterhead and reply with an NEWCLUSTERHEADACK message. After collecting all acknowledgements, C sends a HANDOVERACK message to B. The presence B can now be sure that it will get no more messages in its former clusterhead role. As in the previous operations, B now sends a JOIN message. Then B finishes the handover with a HANDOVERFIN message to C. Since many handover operations may occur simultaneously, we have to schedule the sequence of the handover operations carefully to avoid endless loops of handovers between the same presences. The handover operations a presence has to perform in a movement operation are executed sequentially. The presence starts with the smallest cluster and continues bottom-up with the remaining clusters.

With these basic operations we completed the description of the protocol basics for operation without node failures.

6 Failure Detection

In our mobile scenario, the failure of presences can be considered as a regular event. The heuristics we apply aim at delegating less responsibilities to presences with unreliable connections and thus minimises the probability of clusterhead failures.

Our protocol provides mechanisms for detecting failures and for recovering automatically while still ensuring operability in affected areas of the network.

To recover from failures, we first have to discover them. By using TCP, a presence A is able to detect the failure of another presence B whenever it tries to send a message to B. To ensure that failures of neighbouring nodes in the topology are discovered quickly, we additionally implemented a timeout policy. A presence A has to exchange a message with each neighbour at least once during a timeout t_{msg}. If no regular communication happens during this time, A probes the corresponding neighbour by sending a PING message to it. Thus disconnected presences can easily be detected.

The timeout t_{msg} is chosen carefully depending on the characteristics of the underlying network and the needs of the application. Whenever we detect the failure of a presence, we trigger the recovery protocol.

In our protocol we do not only consider permanent failures, but also short temporary failures. In weakly connected wireless networks such failures may occur when the user passes through an area without reception. The protocol must handle situations in which presences disagree about their mutual states.

7 The Recovery Protocol

To introduce fault tolerance into the protocol, we have to consider a trade-off between maintenance costs and recovery costs. In distributed systems, replication

of central components is often used for fault tolerance. Introducing replicated clusterhead structures into the ContextCast network would implicate a high complexity in maintaining the replicated entities consistent. Additionally, the required protocols would introduce a lot of message traffic for the clusterheads.

Since the failure of a clusterhead should not affect the whole network, our approach rebuilds the network locally in the affected region. This limits the service availability only locally, and the entire system is much simpler to maintain than a replicated clusterhead approach.

As described above, the recovery protocol is triggered when a presence A detects the failure of a presence B. Depending on the actual roles of A and B, the presence A reacts differently.

Failure of the Clusterhead of a Leaf Presence A detects the failure of the parent clusterhead B of its leaf representation. In this case, A can drop any information concerning B and simply try to rejoin the network. If A is a clusterhead at some point, it can process its own JOIN message as the clusterhead of the smallest cluster it manages.

Failure of a Leaf Presence Presence A is a clusterhead and detects the failure of a leaf presence B. In this case A removes B from the local list of leaf presences.

Failure of a Parent Clusterhead The clusterhead A detects a failure of its parent clusterhead B. This is the most interesting case. It may happen that while A still has not noticed the failure of B, the parent of B already detected the failure and declared the corresponding cluster as empty. Still before A detects and handles the failure, other presences may have moved into the cluster. Since for B the cluster is empty, these presences build a new subtree in the network. It is possible to handle this case in a trivial way, where A tells all children to completely reset and rejoin the network. As a consequence, the service in the affected region breaks down completely while the network is being rebuild. We try to avoid such a situation by reinserting the whole subtree of A or its subtrees into the network and thus keeping the subtrees and the services intact for the region covered by the subtree. When a parent clusterhead fails, we can distinguish two major cases. In the first case, a presence detects the failure of a clusterhead, that is not the root clusterhead. Here, it can rely on higher level clusterheads coordinating the reorganisation of the network. In the second case, when the failure of the root clusterhead is detected, the bootstrapping protocol supports reconstructing the network.

Failure of a Non-root Clusterhead To reinsert its subtree into the network, A sends a REINSERT message to the net like a JOIN message. The REINSERT message contains the cluster S_a covering the subtree of A that is supposed to be reinserted. The message is routed to the clusterhead C of the smallest cluster S_c containing S_a.

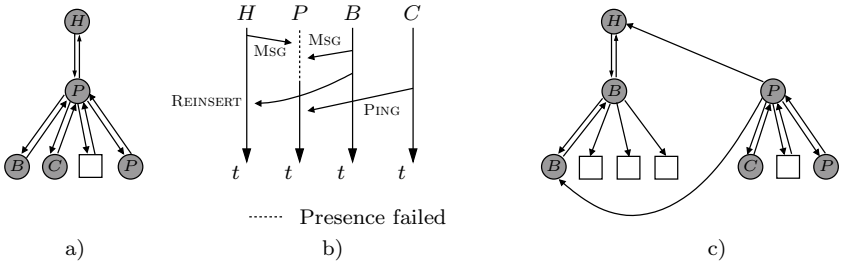


Fig. 4. a) Initial situation b) Sent messages c) Resulting topology

If C knows that S_a is currently not managed by another clusterhead, it assigns A the clusterhead role of S_a and of all clusters on the path between S_c and S_a. The presence A is notified with a ReinsertACK message.

If S_a is already or still managed by another presence, C sends a negative acknowledgement ReinsertNACK to A. Upon receiving the negative acknowledgement, A can either fall back to the trivial strategy to rebuild the whole subtree from scratch, or tell its children to try to reinsert their subtrees. In both cases A itself has to give up its clusterhead role of the corresponding cluster.

If this protocol times out, A assumes that the root clusterhead also failed and continues with the corresponding protocol.

Failure of the Root Clusterhead When the root clusterhead fails, A takes over the role of the root clusterhead and follows the bootstrapping protocol described in section 5.2.

Failure of a Subclusterhead A clusterhead A detects the failure of a subclusterhead B. In this case, it is sufficient to remove the reference to B. If new presences join the now empty cluster before possible subtrees where reinserted, the following reinsertion attempts will fail. To avoid this, A can buffer incoming Join messages for the affected region until a timeout t_{buf} or a buffer overflow occurs, or an appropriate subtree is reinserted. At the cost of a possible delay for some join attempts, this strategy can considerably increase the stability of the topology. The timeout is selected according to the timeouts for failure detection.

Handling Failure Disagreements As noted before, short temporary failures of a clusterhead P can lead to a disagreement about its state. An example is shown in figure 4.

Different types of messages may only be exchanged between neighbours in the topology, e.g. Ping messages. A disagreement about the state of a presence leads to messages that do not follow these rules. Therefore each presence checks incoming messages for such cases and sends a ForceReinsert message to the source presence. When the source presence receives this message, it initiates the same protocol as if the corresponding clusterhead failed.

8 Future Work

In the ContextCast protocol a presence does not have a spatial expansion. We only support point spaces. There are numerous applications, where presences with spatial expansions are useful. Supporting such presences requires different clustering strategies.

In the current protocol, we use a fixed decomposition rule for clusters, where size and shape of the resulting clusters do not depend on the location and density of the presences in the original cluster. It may be interesting to compare this approach with more adaptive decomposition rules.

It should further be investigated how privacy concerns of different presences in the network can be addressed.

The present version of the protocol mainly considers a one to one mapping of peers to presences. It may be an interesting to investigate protocols for peers that manage a set of presences.

9 Conclusions

With the ContextCast protocol, we created a self-organising network that supports contextual messaging and resource discovery. It takes into account context information and the capabilities of the individual presences. The created routing backbone is kept consistent with the context of the presences, even when presences are moving. We introduced recovery strategies that provide a robustness to the ContextCast network that makes it ready for practical application. All aspects of the protocol have been successfully implemented in Java and rigorously been tested under realistic real time conditions.

The core idea of the ContextCast protocol is to build a structure whose topology is not predetermined by the physical connections between the nodes, but by their context. The ContextCast protocol brings semantically close presences closer together. Doing so, it makes it easy for applications to become aware of other presences in close proximity. This awareness can spawn spontaneous collaboration between presences with similar interest in a heterogeneous network environment. Without supporting context information in a heterogeneous network environment, one may pass up many valuable chances for collaboration.

References

[1] Staffan Björk, Jennica Falk, Rebecca Hansson, and Peter Ljungstrand. Pirates! using the physical world as a game board. In *Interact 2001, IFIP TC.13 Conference on Human-Computer Interaction*, 2001.

[2] Antonio Carzaniga and Alexander L. Wolf. Content-based networking: A new communication infrastructure. In *NSF Workshop on an Infrastructure for Mobile and Wireless Systems*, Scottsdale, Arizona, October 2001.

[3] Hector Garcia-Molina. Elections in a Distributed Computing System. *IEEE Transactions on Computers*, C-31(1):48–59, January 1982.

[4] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The anatomy of a context-aware application. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MOBICOM'99*, pages 59–68, August 1999.

[5] Dominic Heutelbeck. Context spaces - self-structuring distributed networks for contextual messaging and resource discovery. In *CoopIS'02, Proc. of Tenth International Conference on Cooperative Information Systems.*, pages 248–265, University of California Irvine, USA, 2002. Springer Verlag.

[6] Tomasz Imielinski and Julio C. Navas. Gps-based geographic addressing, routing, and resource discovery. *Communications of the ACM*, 42(4):86–92, April 1999.

[7] Charles Perkins, editor. *Ad Hoc Networking*. Addison Wesley, 2001.

[8] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain, and Son Dao. Modeling and querying moving objects. In Alex Gray and Per-Åke Larson, editors, *Proceedings of the Thirteenth International Conference on Data Engineering, April 7-11, 1997 Birmingham U.K*, pages 422–432. IEEE Computer Society, 1997.

[9] Clip2 Distributed Search Solutions. The gnutella protocol specification v0.4. http://www.clip2.com/GnutellaProtocol04.pdf.

[10] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, pages 149–160, 2001.

Improving Web Server Performance by Distributing Web Applications

Mital Kakaiya[1], Sushant Goel[1], Hema Sharda[1], David Taniar[2]

[1] RMIT University, School of Electrical and Computer Engineering, Vic-3000, Australia
hemas@rmit.edu.au
[2] Monash University, School of Business Systems, Vic 3800, Australia
David.Taniar@infotech.monash.edu.au

Abstract. Distributed interNet Application (DNA) covers a wide range of topics. DNA is a methodology that specifies how to distribute Internet application on various Web servers. DNA helps to generate scalable, reliable enterprise applications. It provides load-balancing techniques to distribute load on multiple Web servers. This paper describes DNA methodology for a distributed application, which enables better performance, availability and service to clients. This paper also provides comparison of applications' performance and scalability between DNA and non-DNA application. The comparison clearly indicates web server performance improvement using DNA methodology. CPU usage improvement statistics are also provided in this paper. Choosing optimized technology is one of the major criteria in a distributed system to achieve best result. Current major industries are moving towards distributed Internet application solution for global market strategies.

1 Introduction

Distributed computing is a technique, which converts a huge software problem into smaller parts and distributes the smaller segments among several computers. It is a complicated job to develop a large application, which is distributed among several servers. Distributed interNet Application (DNA) provides a methodology for such applications, which are easy to understand and implement onto multiple servers. DNA architecture is not a solution but rather a methodology to solve a complex distributed problems. In other words, DNA is just an abstract pattern. It is a software application engineering design, which generates a solution to a set of common generic problems.

The 2-tier architecture works well up to a medium size application requirement. If application is huge, the single server cannot process all user requests. Some application might require lots of memory and processing work. The server needs to process lots of instructions and data to generate final required output for clients. In major cases, increase in hardware speed is not a solution for applications' performance, scalability and reliability. To overcome these difficulties, the single servers' load could be dist ributed among multiple servers. In distributed computing, multiple servers are connected together to perform a specific task in a distributed

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 156-167, 2003.

environment. The multiple servers can be connected in horizontal as well as vertical hierarchy. DNA helps to develop enterprise applications as a scalable, secure, robust and reliable manners [1]. The goal of the distributed architecture is to distribute the processing load across as many resources as necessary; it doesn't mean to distribute the data within the system [1]. DNA is an abstract idea, which helps to understand design of multi-tier client/server application. There are no coding practices, special notation or even restrictions on the technologies to use. Developer can develop and deploy DNA architecture applications without any restriction of using DNA methodology.

Due to distributed development and communication demand, new network-based technologies were invented, which enabled faster, secure and reliable communication protocols and standards between computers within a network. The Internet and web based applications with open Internet standards have great ability to communicate across machine boundaries and provide information in a reliable, secure and efficient way.

This paper discusses how to distribute an Internet application on multiple web servers, which provides scalability, reliability, availability and better performance. This paper also describes performance tests between websites based on DNA methodology and without DNA methodology. It provides clear understanding of a performance improvement and necessary web server load balancing.

2 Distributed Component Technology: A Background

Traditional applications are hardly distributed among various servers due to limited resources, difficulty in developing and managing. Development of distributed application cost is very high and also contains high risk [9]. There are several technologies available to develop robust and reliable distributed development environment, like Component Object Model, Distributed COM, Transaction servers etc.

2.1 Component Object Model (COM)

The traditional applications are made of single monolithic binary file. Once the application file is compiled and published, it does not change until next version of the application is developed and shipped. If there are any changes into the application customers have to wait for the next rebuild. To find out bugs into monolithic application is critical, because incorrect functionality at one point might affect other parts of the application's functionality. It is difficult to find out exact incorrect part in a huge application.

COM [11] is a specification, which specifies binary standard. COM is a platform independent, distributed; object-oriented system for creating binary software that can interact with applications. It defines a standard for components' interoperability and it is available on multiple platforms like Windows, Macintosh, and Unix. Virtually, any programming language can be used to develop a component. This standard is helpful

when different people at different locations develop different parts of the application. COM is easily extendable and contains robust architecture. COM consists of a binary code, which is distributed as a dynamic link library (DLL) or an executable (EXE). The COM is not only specification. The COM has the COM library, which called COM API. It provides components management services that are useful for all components. The COM component provides various advantages to an application like: Dynamic linking, increased performance, scalability, language independence, version compatibility etc. COM+ is an extension of COM and it provides additional helpful services like manage transaction, Just-In-Time (JIT) activation, advanced security, object pooling, queued components, loosely coupled events, basic interception services, deployment and administration. COM+ services help to develop fast, powerful and robust component for an enterprise application [10].

2.2 Distributed COM

The Distributed Component Object Model (DCOM) is an extension of COM with additional functionality of communication across machine boundaries. This protocol enables software components to communicate directly over a network in a reliable, secure and efficient manner. The concept of DCOM was introduced in 1992 by developing Dynamic Data Exchange (DDE) protocol. The DCOM also helps to reschedule one machine's components to another machine's components.

2.3 Load Balancing

A distributed application design consists of various components, which interact with each other and provides reliable required output. The component distribution task requires careful planning and analysis to distribute application on web servers. Load balancing, performance and scalability become key aspects of the design process in a distributed application [7]. Components' runtime load, architecture including logical packaging, physical deployment, remote server workload analysis, and available network bandwidth needs to be considered [3].

Web server receives requests from clients randomly. Server needs to respond to the client's request, so it creates many instances of component within distributed architecture. Due to uncertain request interval, some servers are heavily loaded, while others are lightly loaded [5]. Uneven distribution of the load disturbs performance of the distributed application. Load balancing algorithms helps to reduce uneven load, which improves performance by distributing the component more evenly on various servers. The performance of web server directly reflects with load balance. The over loaded server or unbalanced system provides poor performance result.

The number of variety of applications using the distributed architecture is increasing and the expectancy of customers is also increasing. As the number of users of any application increases the response time also increases [6]. The over-loaded server may not response to all clients, which can result in the timeout of the request. There are two ways to meet the ever-increasing demand of the Internet server performance.

The Modern Approach: The better way of solving the problem is to use a cluster of servers serving clients with one and the same service using synchronized contents. When the requests for the Internet service increases new servers are added to the cluster to meet the increased traffic requirements. The traffic is distributed among the individual servers to balance the load on each server.

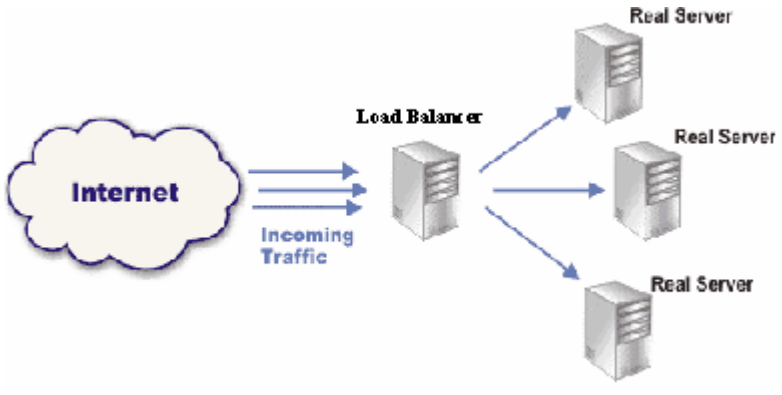


Fig. 1. Modern approach of load balancing

The Traditional Approach: The first way is the single server solution in which the server is upgraded to a higher performance. There is a problem in this approach that soon this server can be overloaded again and a next upgrade will be required. The whole process of upgrading is complex, time consuming and expensive.

2.4 Load Balancing Algorithms

The load balance algorithms are helpful to share load on various servers. Major load balancing algorithms add load state information to existing client requests. There are two types of useful load balancing techniques, which are static and dynamic algorithm [2]. The static algorithm does not distribute request based on current load on web server. The dynamic load-balancing algorithm calculates current load on web servers and forwards request to minimum load server [4].

Round-Robin Algorithm: Round-Robin algorithm is the simplest form of load balancing algorithm. The round-robin scheduling algorithm sends each incoming request to the next server in it s list. Thus in a three server cluster (servers A, B and C) request 1 would go to server A, request 2 would go to server B, request 3 would go to server C, and request 4 would go to server A, thus completing the cycling or round-robin of servers.

Weighted Round-Robin Algorithm: The weighted round-robin scheduling is better than the round-robin scheduling, when the processing capacity of real servers are different. The weighted round-robin algorithm assigns each server hidden weight load based on processing power. However, it may lead to dynamic load imbalance among the real servers if the load of the requests varies highly.

Least-Connection Algorithm: The least-connection scheduling algorithm directs network connections to the server with the least number of established connections. This is one of the dynamic scheduling algorithms because it needs to count live connections for each server dynamically. The least-connection scheduling cannot get load well balanced among servers with various processing capacities. The faster server can process thousands of requests and keep them in the TCP's TIME_WAIT state.

3 Performance Enhancement Using DNA

With the single server Internet application the chance of the server being unavailable is high. Adding new servers can increase availability significantly. If one server has 95-percent availability that would mean it's not available for an average of 1.2 hours a day. The probability of the server failing at a given moment is 0.1. Adding one additional server decreases the probability that both servers will fail at once to 0.1*0.1= 0.01. The likelihood of one of the servers being available is increased to a much-improved 99 percent. The algorithms discussed in the previous section helps to distribute the load to different servers. But the load on the servers is not evenly distributed, as the work to be allocated to the servers is not determined dynamically.

Optimized Weighted Round-Robin Algorithm: This algorithm is similar with the Weighted Round Robin algorithm. But the server can be dynamically assigned a weight depending on its current availability and current load.

Weighted Least Connections: As we know from the previous section, least connection algorithm cannot balance the load effectively among the servers if the processing capacity of the servers is different. In weighted least connection algorithm a performance weight is assigned to each real server. Larger percentage of live connections is assigned to the server with the higher weight value. If number of connections are `C1, C2, C3,...,Cn` and the performance weight assigned to the servers are `W1, W2, W3,...,Wn`, then as per the weighted least connection any new connection will be assigned to the server with minimum `Ci/Wi` value (where `i = 1,2,3,...,n`). The advantage of using this algorithm is any new connection will be allocated to the least loaded server.

Any cluster of servers or a server farm not properly balanced; may reject client requests because some of the servers may be at their performance level threshold but others may be still well under the threshold. But using the proper load-balancing algorithm like the Weighted Least Connection may distribute the load evenly in all the servers in the server farm. The following figures (2 & 3) show the comparison between the two:

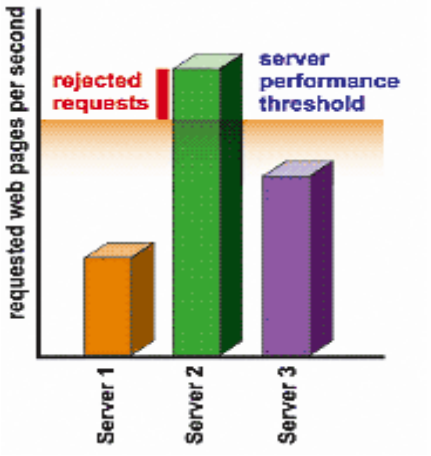


Fig. 2. Improperly balanced server-farm

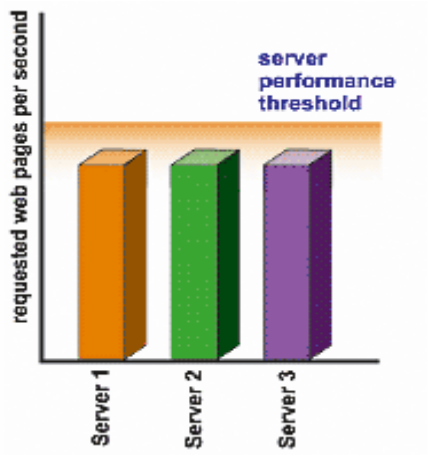


Fig. 3. Properly balanced server-farm

3.1 Distributed interNet Application

Distributed interNet Application (DNA) describes architecture for building multi-tier distributed computing solutions. The major tier of the DNA is Presentation tier, Business tier and Data tier. Each tier represents own services to the application. Each layer or tier usually resides on a different virtual machine. The presentation layer only communicates with the application or middle layer, which contains the business objects. The middle layer handles the applications processing logic and in turn communicates with the data access layer, such as SQL server. A three-tier application allows the implementation of thin client and is much more flexible and easy to maintain than a two or one-tier. For example, the data storage layer can be substituted completely without having to change any code at the presentation layer.

There could be more than one web server and application logic server depending on the traffic. Hence there could be a **Load-Balancing Layer** in addition to the Presentation-Layer, Business-Layer and Data-Layer. Load-balancing solutions present a single system image to clients in the form of a virtual host name, and distribute client requests across multiple application servers.

Presentation-Layer: The presentation layer handles the basic user input and output. It is responsible for providing the graphical user interface. This layer collects input from client and sends user input to business services for further processing. Some of the presentation layer tools are: DHTML, VBScript, Jscript, Browsers, activeX controls.

Business-layer: Business layer is the core of the application. The business service receives user input from presentation service tier, performs business operation automated by an application, interfaces with data service as necessary, and returns result to presentation service. Some business layer tools are: COM+, IIS Server, ASP, ADO.

Data-Layer: Data service receives requests from business service, retrieve data from database, and check data integrity and returns result to business services. Some Data layer tools are: Exchange server, OLE DB providers, SQL Servers and other DB servers.

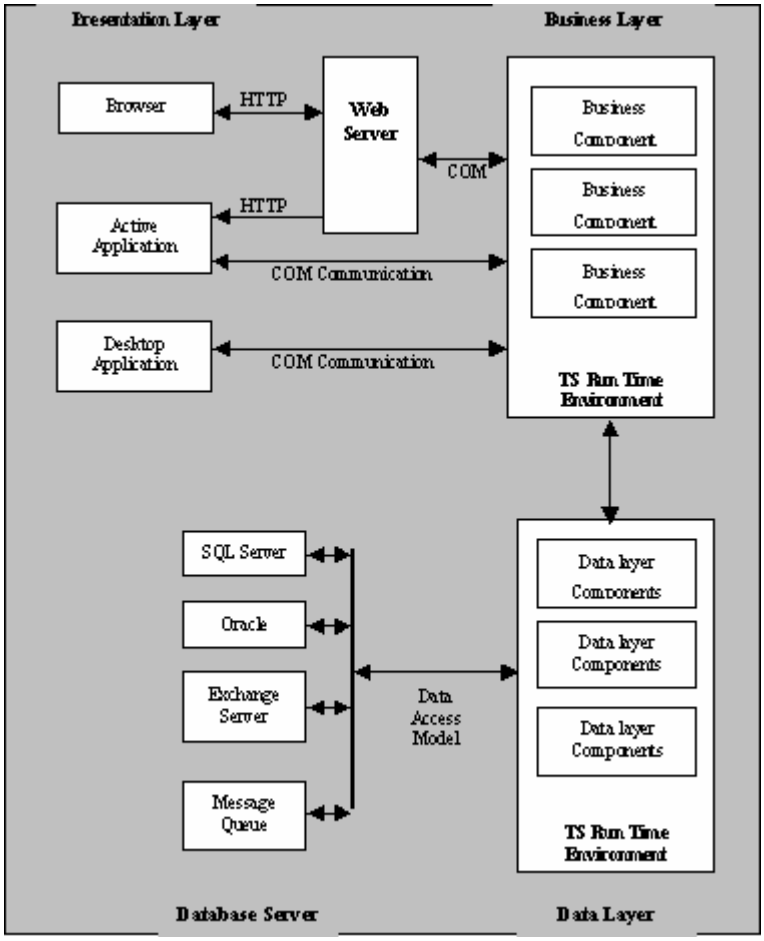


Fig. 4. DNA architecture overview

4 Performance Evaluation

To study the behavior of the Distributed interNet application and the application which is not distributed we performed the test on the following test bed:

- Processor: Intel P-III, 850 MHz.
- SD-RAM: 128 MB

- Hard Disk: 5 GB
- Network Card: Intel 8255X based PCI Ethernet Adapter (10/100)
- Windows 2000 server
- IIS 5.0
- MS-SQL Server 2000
- MS- Web Application stress (WAS) tool Stress Version: 1.1.293.1

The WAS tool generates arbitrary requests and is used to measure the performance level of the server. The test is done over two terminals. One terminal is dedicated to run the application, the other terminal is used to run the WAS tool and any activity other than running the application server. The test is conducted over two terminals to ensure that the developed application gets full attention of the processor in which the application is running so that the results are correct.

The tested application is a very standard e-commerce web site with members login page, new clients registration pa ge, product display page, shopping cart page, payment page etc. the WAS tool generates random requests for a specific time and measures the response from the server. The testing is done for two different cases:

I. The application is developed using Distributed interNet Application technology, three separate layers are created as discussed in the previous section.
II. The application is developed without using the component technology.

4.1 Sample Code at Different Layer (Using DNA Technology)

The **Presentation Layer** for the application consists of HTML and DHTML coding. The HTML coding is an important interface for users to communicate with website. The following lines show an example of user interface for web browser:

```
<html>
<head> <title>Home Page</title> ..... </head>
<body> ..... </body>
</html>
```

For presentation of a website, It needs information from database and various resources. The application is not able to communicate directly with database. It needs to pass through business layer component to maintain applications security.

The presentation layer creates an instance of a business layer component. The business layer component is registered into MTS environment. It will create object in MTS runtime boundary. The following lines shows create instance of `BusinessCOM.cbCart`, which is defined as an `objCart` Object.

```
Dim objCart
'Create instance of a component
Set objCart = Server.CreateObject ("BusinessCOM.cbCart")
```

Business Layer shows component based development code using Microsoft Visual Basic development language. Class initialize and terminate events are defined which is called automatically based on object creation and deletion.

```
Private Sub Class_Initialize()
  ' Component Initialized variables here
End Sub

Private Sub Class_Terminate()
  ' Component Terminate variables here
End Sub
```

All required variable declaration has been defined inside InitializeVariables and TerminateVariables functions. These functions contain general required initialization and termination declaration for components, so it needs to call from all components. It saves time and cost of development process. Modification at one function reflects changes into all components. The business component defined in this application has a `Get property` that returns the value of specified variable. Some of the methods defined in the business Layer are `AddCart(adds new product in the shopping cart), UpdateQty, ClearCart, TotalProductPrice, IsValidUser` etc. Following is the sample code for `ClearCart` function.

```
' Name of function: ClearCart
' Purpose          : Clear all products from a Cart.
' Returns        : true indicates Successful.

Public Function ClearCart() As Varient

' Assume successfully not clear products from cart

  ClearCart = False
  Dim intCountCart As Integer

   'loop until all products are clear.

  For intCountCart = 0 To MAX_CART
    TCart(intCountCart).m_intProductID = 0
    TCart(intCountCart).m_strProductDesc = ""
    TCart(intCountCart).m_intProductQty = 0
    TCart(intCountCart).m_curProductPrice = 0
  Next intCountCart

  'Successfully clear products from cart

  ClearCart = True
 End Function
```

The cart retains in memory up to users session time. It is automatically destroyed when users session is destroyed. It does not need to connect with database and store values. This component is not required to connect with data layer component for shopping cart operation. This component has no data layer functionality. It still uses data from database by accessing other data layer components. If username and password is provided non-zero length and valid values, the business layer component creates an object of a data layer components customer class.

```
'Variable Declaration
Dim objCustomerData As DataCOM.cdCustomer
'Create Object of Business Layer Customer class
SetobjCustomerData = CtxCreateObject
                                (TProgID.Data_cdCustomer)
```

The object is created using `CtxCreateObject` method. This method enables to create a new instance inside MTS runtime. The MTS handles to minimize impact of memory allocation and resources. This helps to improve overall performance.

Data layer is useful to communicate with database. The data layer is developed on Microsoft Visual Basic environment. The data layer always check the data validity before modify any permanent changes into a database. If data validity is correct and it could not destroy any current information from database, it sends login request to database objects. The data layer does not directly communicate with database table objects. The database forwards request to stored procedure of SQL Server and passes all required information to it. The following code creates ActiveX Data Object (ADO) Connection and Command objects. The object is running inside MTS environment by calling CtxCreateObject method.

```
 Dim objCmd As ADODB.Command
'Create ADO Command Object inside MTS
 SetobjADOCommand=CtxCreateObject
                            (TProgID.Data_clsADOCommandC)
'open the connection object
 Set objConn = objADOCommand.OpenDB()
'Command Object calls Stored Procedure of a database
 Set objCmd = objADOCommand.LoadProc(objConn,
             TDB.TStoredProcName.prc_tblCustomer_login)
```

The Command object executes stored procedure, which returns a result based on arguments. The store procedure helps to execute faster query and data access.

Test Results

Two website test is taken using same hardware and test configuration: DNA and NODNA. The DNA indicates test data, based on DNA methodology web application. The NODNA indicates test data information using non-DNA methodology. The paper describes both website test reports and performance comparison. The DNA website is developed based on DNA methodology and sample DNA code described in the above section. NODNA website does not follow DNA methodology rules. NODNA website connects directly to database without interfere with business layer or data layer. NODNA website has no business layer or data layer.

Test results for DNA application:

```
DNA Overview
=============================================
Report name                          : DNA
Run length                           : 00:01:00
Web Application Stress Tool Version: 1.1.293.1
Number of test clients               : 1
Number of hits                       : 3989
Requests per Second                  : 66.39
%processor time                      : 83 %
Connection attempts/Sec.             : 71
Request handeled/Sec.                : 43

Socket Statistics
-----------------------------------------
Socket Connects                       : 4594
Total Bytes Sent (in KB)              : 1673.44
Bytes Sent Rate (in KB/s)             : 27.85
Total Bytes Recv (in KB)              : 23171.77
Bytes Recv Rate (in KB/s)             : 385.63
```

Test result for NONDNA application:

```
NODNA Overview
=============================================
Report name                          : NODNA
Run length                           : 00:01:00
Web Application Stress Tool Version:1.1.293.1
Number of test clients               : 1
Number of hits                       : 1222
Requests per Second                  : 20.36
%processor time                      : 90 %
Connection attempts/Sec.             : 20
Request handeled/Sec.                : 13

Socket Statistics
-------------------------------------------
Socket Connects                      : 1367
Total Bytes Sent (in KB)             : 604.34
Bytes Sent Rate (in KB/s)            : 10.07
Total Bytes Recv (in KB)             : 20758.11
Bytes Recv Rate (in KB/s)            : 345.89
```

4.2 Result Discussion

From the above mentioned test results it is very obvious that the number of requests handled by the DNA application is much higher than the NONDNA application. The %processor time is an important criteri on during performance test. If %processor time goes above 90% of total time, it may result in delay in response. The incoming

request might wait into a request queue. The usage of processor time of the DNA application is less compared to NODNA web application. It increases performance of the website by reducing processor time and increasing response time. The performance test shows clearly that DNA website provides high performance with low CPU usage. It helps to handle more requests with less processing power because of caching various business layer and data layer objects into memory.

5 Conclusions and Future Work

As a result of growing interest and need for more powerful enterprise solutions, vast amount of research is undertaken in this field. This field depends on major research areas such as networking, programming language improvements, hardware and software technology changes, and database speed improvements. The improvement of any area may provide significant change in overall performance of the application. The DNA is not restricted with any programming language. It helps to develop Internet based applications by choosing any programming language. This provides benefits of using latest programming techniques and language such as Microsoft .Net framework, Microsoft asp, Sun jsp, php etc to develop a web application.

The test result of a real time application is also important during comparison between test results. The real time configuration and real time application output indicates exact performance capability of the web servers. The real time application response is an important factor to achieve better performance result.

References

1 Joseph M., *Enterprise Application Architecture With VB, ASP and MTS*, Wrox Press Ltd. 1999,

2 Colajanni M., Yu P., Diad D., "Analysis of task assignment policies in scalable distributed wed-server systems," *IEEE Trans. Parallel And Distributed systems,* 9(6):585-600, June 1998

3 Ezhilchelvan, Palmer D., Khayyambashi R., Morgan G. "Measuring the Cost of Scalability and Reliability for Internet-based, server-centered applications," 6 th *Intl. Workshop on Object-oriented Real-time Dependable Systems (WORDS01)*, Rome, 01.

4 Petra B., Tom F., Ernst W. "Parallel Continuos RandomizedLoad Balancing," *Proc. of the 10th Annual ACP Symposium on Parallel Algorithms and Arch.*, pp. 192201, 98.

5 Thomas L., Jon G. "Analysis of Three Dyna mic Distributed Load-Balancing Strategies with Varying Global Info. Requirements" *IEEE Computer*, pp 185192, August 87.

6 Xu C. and Lau F., *Load Balancing in Parallel Computers,* Kluwer AcademicPublishers, Boston,MA, 1997.

7 Andresen D.,Yang T., Ibarra O. H., Smith T. R., "Scalability issues for high performance digital libraries on the world wide web," *Proceedings of IEEE ADL, Forum on research and technology advances in digital libraries*, Washingoton, 1996.

8 Clements, Paul, From Subroutines to Subsystems: Component-Based Software Development," *The American Programmer*, Vol 8, No 11, Nov 1995

9 Clements P. E., "Requirements for software systems which enable the flexible enterprise," 1997, *MSI Research Institute.*

10 Microsoft COM Technologies" http://www.microsoft.com/com/

On How to Model Content Engineering in a Semantic Web Environment

Lutz Maicher

Department of Information Science, University of Leipzig
maicher@informatik.uni-leipzig.de
http://www.informatik.uni-leipzig.de/~maicher

Abstract. The technologies of the Semantic Web demand complex publications which themselves are the result of complex production processes. The complexity of these publications is cause and effect of more sophisticated communication processes possible through the Semantic Web. We introduce a model which briefly describes these processes on a solution-independent level by using a market perspective. The model is based on the assumption of a market interaction between content offer and demand, but is independent of the existence of real content markets with financial transactions. We emphasise the need of structured guidelines for Semantic Web Content Engineering Processes. Furthermore, the model represents a foundation for their development.

Introduction

The Semantic Web will provoke more and more complex publications, which are context-sensitive and readable by man and machines. The results of Semantic Web research will drastically contribute to the complexity of the production processes of the publications themselves. Such processes are called Content Engineering Processes (CEP). Publications will be "enriched" with sophisticated metadata to improve the information retrieval process. Furthermore, each reader will use provided languages to describe the actual delivery context[1] and its desire for specific contents to personalize the communication process. In this paper, the more technical delivery contexts and the user's desire for specific contents will be merged into the term of "demands". These demands can be described in sophisticated ways in the semantic web, which has strong effects to the production processes of the publications. It is important to understand these processes from an economic viewpoint because of the strong impacts on a lot of different business processes in organisations.

Today, there are only few publications available for Semantic Web technologies. On the one hand, this is caused by immature technologies but, on the other hand, by the absence of structured guidelines for publication production in the Semantic Web

[1] Delivery contexts can be characterized in terms of specific user preferences and abilities, capabilities of the access device and available network resources [Osse+02].

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 168-179, 2003.

environment. Unfortunately, the adjacent research communities Content Engineering (CE) and Semantic Web are still working relatively independently [Osse+02]. A closer cooperation in future will be necessary due to the importance of the publication production processes in enterprises.

In this paper, we introduce a model which is based on a market perspective. Publications are seen as a collection of offers for personalised content supply. The model is solution-independent and regardless of existing marketing buzz words. The model should serve as an interface between research in CE, Semantic Web and other related research areas like information retrieval, software engineering, business process engineering and service engineering.

A short summary of the paper's structure will conclude the introductory chapter. First, the idea of the Semantic Web and the term of Content Engineering is discussed in detail. Then, the model is described as being based on four assumptions which initiate its further development in the paper. Subsequently, the modelling of publications, which form the basis of the communication process in the Semantic Web is explained. Then, the communication processes are described in the introduced communication space, whereas the production space produces the publications for the communication space. Because the processes in the production space have to be realised in organisations they will be discussed in more detail. To conclude the paper applications of the model are discussed.

The Semantic Web

Berners-Lee et al. describe the Semantic Web as "an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" ([Bern+01]). Here, it should be emphasised that opportunities for cooperation should be improved for computers *and* man ([Osse+01]). The Semantic Web technologies not only can be deployed to improve information gathering and brokerage in the web, but also to present information most appropriate to each consumer ([Osse+02]).

The term Semantic Web pools an enormous amount of different approaches, which is illustrated by the vision of Berners-Lee et al. Thus, the Semantic Web does not define itself, but it only exists in communities which have made prior agreements concerning their approach to it.. At our abstract level we refer to the Semantic Web as a whole knowing that its realisations are only community depended occurrences of a research pool with own specific characteristics. See [Fens+03], [Hyv+02] for a good introduction in aims, technologies, languages and applications of the Semantic Web research.

The task of the CEP in a Semantic Web environment is the construction and the maintenance of a system supplying sophisticated, goal-oriented communication processes of agents. These agents are located in different points of space and time (see [Romh98], p. 147; [Maic02], pp. 24). Figure 1 describes the basic characteristics of such a communication process in the Semantic Web.

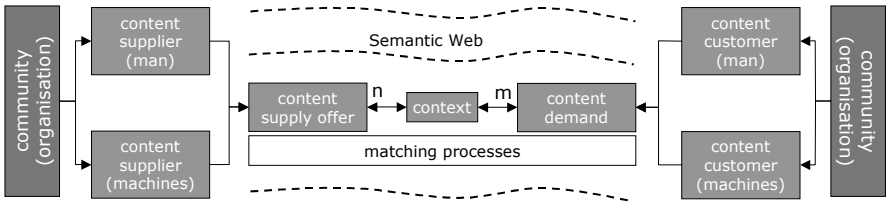


Fig. 1. The communication process in the Semantic Web

The Semantic Web develops its full potential in communication processes with the following characteristics:

- *a large number of content suppliers* (man and machines),
- *a large number of content customers* (man and machines),
- the communication partners are *not necessarily known* in advance,
- the communication is placed in *different contexts*,
- a *heterogeneous* and *decentralised* environment,
- based on *mass-data*,
- *trust and strategy* in and between *communities* are important.

Especially the cooperation of automatic information suppliers and customers leads to new challenges for the CEP. These opportunities will be leveraged by an appropriate application of context-sensitive publications. Besides, the communication in communities can be improved if the CEP is able to realise the benefits of the Semantic Web. However, communication based on Semantic Web technologies will only be able in communities which have made prior agreements about the technological parameters.

The Content Engineering Process

In the knowledge based economy content can be seen as a preliminary product, like screws and joints in an industrial process of manufacture. It is produced from the raw material knowledge and have to be refined in publications for the end-user's consumption. For a sustainable usage of the limited resources of organisations a reuse of knowledge, content and publications is necessary and is therefore to be integrated in the CEP.

We will regard Content Engineering as the industrial manufacture of publications (see figure 2) and use the term CEP as the goal-oriented and process-based *generation* (collection, production, storage), *transformation*, *aggregation* and *representation* of content in publications.

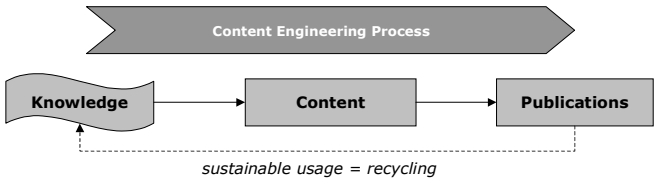


Fig. 2. The Content Engineering Process as an industrial process of manufacture

Modelling Content Engineering

The basis of the model from a market perspective is described by following assumptions:

1. Existence of offered content supply and (anticipated) content demand.
2. Each interaction between a content suppliers and a content customers is the result of a market matching process
3. Content suppliers and content customers try to maximise their utility.
4. The market perspective is independent from the existence of real, financial transactions.

We propose to distinguish between a *production space* and a *communication space*. The production space generates publications for the communication space. This process is modelled and referred to as CEP. Subsequently, the communication space is produced by the production space. Its inherent communication processes are to be analysed separately.

The Model of Publications

Before starting the development of the models of the production and the communication space publications as their connector have to be introduced. A publication will be called **m**. The characteristics of such a publication are (see [Roth+01], pp. 134):

- it is made for an (anticipated) demand formulated by the customer,
- it is produced for man and/or machine consumption and
- it is not necessarily persistent.

The discussion in research about adaptive hypermedia systems suggesting a separation of content and links ([Bech+01], [Staf97], [Wild+00]). We extend this approach by introducing a separation of the publication's content which is made for consumption, and a set of content offers which is made for the maintenance of the

communication process. Only these offers describe the possible content supply provided by the producer.

In the current web contents are produced for human consumption, like texts about enterprises or people, and are primarily written in HTML[2]. In the Semantic Web information will be described by sophisticated meta-data by using different languages (RDF[3], DAML+OIL[4], Topic-Maps[5]) and general or domain-specific ontologies. The set of all contents in a publication **m** will be called **c(m)**. The introduced market is rather based on a competition in **c(m)** than in **m**.

The consumer of a publication has to understand **c(m)**, since a publication comprises references to the used languages and ontologies as well as the original meaning of **c(m)**. As the result of this consumption process the consumer will formulate a new demand to continue the communication process. However, formulating its demand it can only use the offers for content supply provided by **m**. The publication provides these offers as a set of production rules called **r(m)**. These production rules can be seen as a language for the content demand description. To summarise, we will define a publication **m** as the following tuple **m=(c(m),r(m))**.

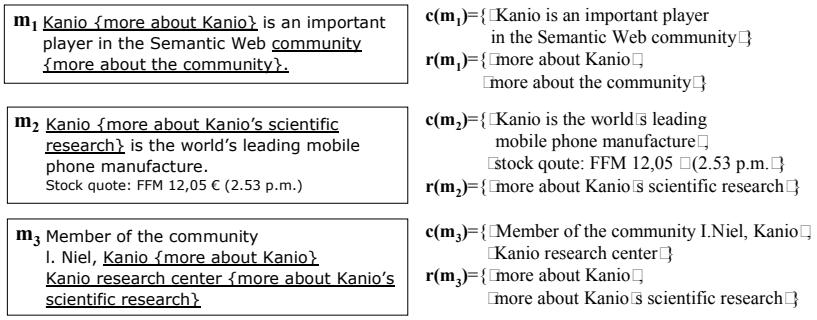


Fig. 3. Example of three publications

We will define the set of possible sentences of **r(m) as k(m)**. At present the main possibility to formulate a demand in the web is the usage of hyperlinks. For this kind of publications **m** the set of all these links is **r(m)**. As shown by the simple example introduced in figure 3, **r(m)** and **k(m)** are equal in this case. Although **r(m)** and **k(m)** are not equal at the entry site of a search engine. If **m** provides the possibility to enter a search query, **r(m)** describes only all valid letters and the valid length of this query. **k(m)** is the set of all possible search queries which can be produced dependent on the constraints formulated by **r(m)**.

In the Semantic Web the communication process will further be maintained by more sophisticated mechanisms than simple hyperlinks. Therefore, the web will alter

[2] see http://www.w3.org/MarkUp/

[3] see http://www.w3.org/RDF/

[4] see http://www.w3.org/TR/daml+oil-reference

[5] see http://www.topicmaps.org/1.0/

from a "web of links" to a "web of offers". Already some features of the XML Linking Language[6] show possible further developments.

The Model of the Communication Space

After introducing publications the communication space will be modelled. It is a pair {**S**,**D**} and exists where published **m** and their consumers meet. This is based on the chosen market perspective. A publication supplies content according to a demand formulated by a customer. Furthermore, a publication offers new content supply to continue the communication process. If a content supply offer is published, the producer is forced to accommodate the induced demand. So, the production space is obliged to produce the offered content as long as the offer is available. This modelling approach based on transactions prevents the introduction of a time model.

The communication space consists of a supply system **S** and a fictive demand system **D**. The supply system **S={m}** is the set of all publications made by the producer which are available to customers. One has to bear in mind that each sentence **k(m)** is, on the one hand, a possible content supply offered by **m**. On the other hand, customers can only formulate their demand with one sentence of **k(m)**. This ambiguity of **k(m)** is important for the model. Because **m** is a tuple **(c(m),r(m))** and **k(m)** is the extension of **r(m)** the supply system is **S={(c(m),k(m))}** as well.

The demand system **D={(m,d)}** is a set of tuple **(m,d)**. *Each* consumer of a publication **m** is described by its real demand **d**. The real demand describes *all* about the consumer's wants and possibilities. So **d** includes much more statements than expressible by **r(m)**.

Matching processes occur in the communication space. Each costumer consumes **c(m)** of a specific publication **m**. While consuming the publication its real demand **d** alters. If the customer wants to continue the communication process, it has to formulate a new content demand. The publication provides only the limited language **r(m)** and the customer describes its demand by choosing one sentence from **k(m)**. In most cases this leads to a loss of information. Furthermore, one has to pay attention to the strong interrelations between the concepts "demand" and "context" already discussed. In the case, other customers do not want to maintain the communication process while the state transformation they can alternatively choose the empty set. This is formalised in expression 1:

$$P_d : (m, d) \rightarrow k(m) \vee \{\varnothing\} \qquad (1)$$

The matching between supply and demand is realised by $\mathbf{P_d}$. The production space interprets the chosen sentence from **k(m)** and transforms the communication and production space according to its goals. This transformation function **T** is subsequently discussed in detail.

[6] see http://www.w3.org/TR/xlink/

The state transformation function **R** is formalised in expression 2:

$$R : D \xrightarrow{P_d} k(m) \xrightarrow{T} S \qquad (2)$$

Figure 4 illustrates an example. There are three publications available to the enterprise Konia (see figure 3). In publication $\mathbf{m_3}$ only the decision between „more about Konia" and „more about Konia's scientific research" is possible formulating a content demand. In the demand system **D** four customers consume publications in the state *i*. Publication $\mathbf{m_3}$ is consumed by customer *4*. While consuming the publication $\mathbf{d_4}$ of this customer alters. Perhaps it needs more information about Topic Maps and the Semantic Web. Yet, $\mathbf{m_3}$ provides only something „about Konia's scientific research". In this case, the customer formulates its demand with this sentence. The customer expects a publication which maximises its expected utility at the state *i+1*. When the production space receives the new demand, the communication (and production) space will be altered according to the goals of the producer. A considerable difference between the customer's expectation and the producer's deliveries is caused by the wide range of possible interpretations of the demand.

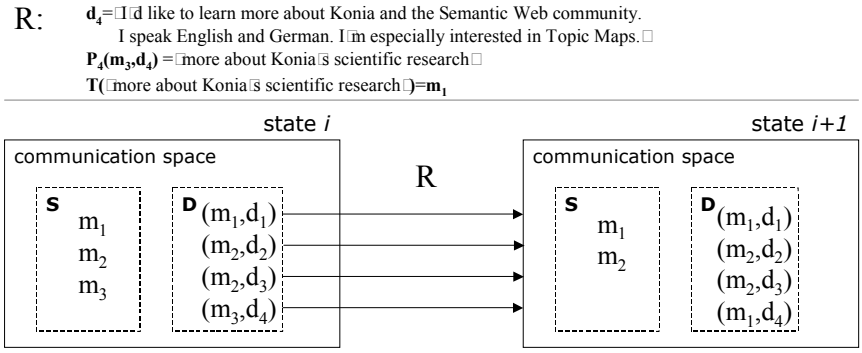


Fig. 4. Example for the communication space

In the example the other customers did not want to continue their communication processes at the given state. They chose the empty set. The result in the example is the reduction of the supply system. According to the demand, the production space has provided publication $\mathbf{m_1}$ to customer *4*. In the new state *i+1* the production space is only to provide content supplies which are offered in publication $\mathbf{m_1}$ and $\mathbf{m_2}$.

The Basic Model of the Production Space

In the production space the CEP is implemented. The CEP is the collection and production of a set of (raw) content objects **q** (texts from internal and external authors, external or internal databases, ERP-systems, RSS-feeds[7]) and their transformation in a set of customised publications **m** which will be published in **S**.

[7] see http://www.mnot.net/rss/tutorial.

Each content object **q** can be either atomic, i.e. a special text from an author, a set of content objects or a function. Domain and co-domain of these functions are content objects too. For example, a function can be a service which provides the stock quote for a given enterprise in a specified content object. These extensions allows the encapsulation of intelligence in the content objects.

Concerning these assumption the set of all *1..I* source objects **q** is the knowledge base **Q** of the CEP. The set of all *1..L* offered publications **m** is **M**. The set of all possible publications which can be produced from **Q** is $\mathbf{M_Q}$. Formally, the whole CEP is the production of **Q** and of a function **T** which chooses for all content supply offers published in **S** a publication from $\mathbf{M_Q}$:

$$\forall m \in S \quad \forall l \in k(m) \quad T(l) \in M_Q \tag{3}$$

According to the market perspective each publication **m** should be producible on demand if the offer is published. The demand is one possible sentence of **r(m)**. The production room's task is the interpretation of these sentences and the development of the according part of the function **T**. Here, one has to bear in mind that the producer tries to maximise its utility. One of the main design issues of the Semantic Web is that it has to handle inconsistent data[8]. So, we can imagine scenarios where the users' demands will intentionally not be accommodated, because the produce tries to maximise its utility and he has the technological possibilities.

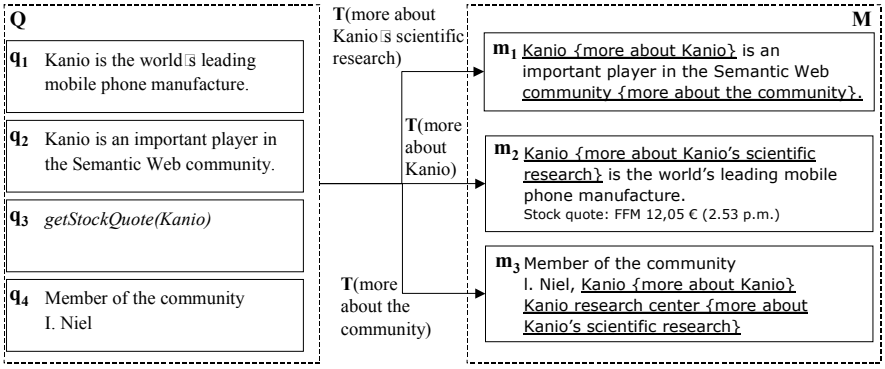


Fig. 5. Production space of the example

Figure 5 shows the production space of the introduced example. Four content objects will be collected to produce the system. The first object describes the enterprise, the second its scientific activities, the third is a service which provides its actual stock quote and the fourth characterizes the research community. If the production space receives the demand "more about Konia's scientific research", publication $\mathbf{m_1}$ from $\mathbf{M_Q}$ will be selected. There are only few opportunities in the publications to explicit a new demand, e.g. only two links are provided in $\mathbf{m_1}$ for further information.

[8] see http://www.w3.org/DesignIssues/Inconsistent.html

The Detailed Model of the Production Space

The number of possible demands and $\mathbf{M_Q}$ extremely grows in the Semantic Web. The introduced function **T** is a relevance function which chooses the right publication from the large set $\mathbf{M_Q}$. At an abstract level we adhere to this approach, but at an practical level **T** is a transformation function, which transforms **Q** into **M** in a pipeline processing model. In order to handle the remaining complexity we propose the separation of the production space in *a source system*, *a concept system* and *a publication system*. The separation shown in figure 6 structures the already introduced tasks of the CEP.

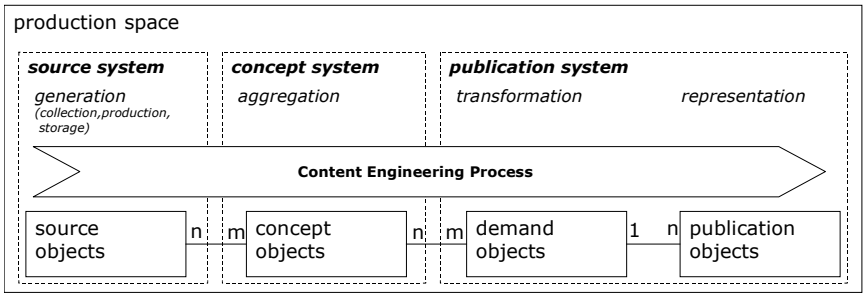


Fig. 6. Separation of the production space

The source system is **Q**, the set of all source objects **q**. All objects with relevance to the objectives of the CEP's owner have to be *produced, collected* and *stored*.

A concept[9] in knowledge representation is each real or abstract "thing of interest" whereas statements exist ([Reim91], p. 14). The concept system is the set of concept objects. A concept object is also a content object and represents a view of **Q** which pools all statements (content) according to a special concept, i.e. specific processes, roles, employees or products. The concept objects will be *aggregated* from **Q** according to the relevant concept. We put forward an object-centric knowledge representation approach (see [Reim91]) to represent the whole concepts' knowledge.

A separation of concerns between logic, content and layout (see [Roth+01]) is applied in practice. Especially in the Semantic Web the same content can be published in various ways. See as an example [LeGr+01]. According to this heuristic the publication system is separated into the set of all demand objects and the set **M** of all publications.[10] A demand object represents the demand-based view on the concept system. The production space interprets the demand and *transforms* all relevant

[9] A number of man-centered approaches to the Semantic Web are based on concepts as a central design criteria ([Avel+02], [Scha02], [Thom02]). While developing concept-based solutions the conceptual hypertext system research should be used ([Osse+01], [Bech+01]).

[10] The number of all demand objects and publications is not equal because to one demand object different representations can be produced (HTML, PDF, WML, VOXML, SVG, VRML, RTF, Plain Text).

contents from the concept system according to the goals of the producer. The task of the layout production is the development of layout transformations for each demand object. These transformations linearise the demand objects for *representation* in a medium.

How to Apply the Model?

The proposed model helps to structure the research in Semantic Web Content Engineering. We propose to use "Cocoon"[11] from the Apache project to simulate all ideas concerning the production space. Cocoon is a completely XML-based publishing framework which allows the separation of content, logic, style and management with a sophisticated pipeline processing model. Furthermore we propose the following research efforts:

Development of the model and the formalism

The following limitations of the model should be lifted in further research:

- integration of goal systems for producer and customer, which helps to evaluate the quality of the publications (difference between customers' expectation and producers' realisation),
- further enhancement of "demand" as a union of context and desire and
- further discussion of concepts in connection with the Semantic Web.

Development of IT-systems

Concerning the development of content-based applications the following ideas should be discussed in detail:

- Development of a data-type "content" as an encapsulation of content and intelligence, which can be used to model and realise content-based applications at different abstract layers.
- The content exchange in a content commerce scenario should be supported. This has strong advantages if content will be supplied as web services with defined service level agreements.
- If customers are machines, $\mathbf{P_d}$ is an automatic matching between two formal languages. This fact touches research in collaborative ontology design and usage ([Hols+02]).

Processes and Organisation

The CEP is a main business process in knowledge based enterprises. The model helps to describe requirements for the development of the following (business) processes:

[11] see http://cocoon.apache.org/2.0/

- initial definition of **T** and **Q**,
- manual or automatic development of **T** and **Q** according to the development of $\mathbf{P_d}$ and to the communication and production space,
- development of processes for goal definition and enforcement,
- identification of communities which already use Semantic Web technologies (which understand **c(m)** and use **r(m)**),
- integration of results from service engineering research.

Strategy and Trust

Strategic thoughts are necessary in the Semantic Web, especially in enterprises where each publication is the result or the preparation of an value-adding business process. In this case, the communication space represents real markets with strong impacts to the real world. Nevertheless, these markets have anomalies. Disinformation and deceit will frequently occur, which implies the use of game theory in connection to CE:

- Which parts of the demand should (not) be satisfied? Which parts should be ignored or definitely used?
- Do recipients deviate from benevolent strategies? Does the demand represent the honest transformation of **d** or does it only represent a construction to manipulate the results of the communication process?
- Do producers deviate from benevolent strategies? Does **m** represent a honest transformation according to **d** or an intentional manipulation?
- Which communities are trustworthy in which context?

Conclusion

The necessity of an integration of research in Semantic Web technologies and Content Engineering has been shown and emphasised.

The market perspective characterises the communication processes in the Semantic Web, although it is independent from the existence of real financial transactions. Especially the introduction of □offers□ and □demands□ instead of links meets the requirements of the Semantic Web. The Semantic Web will be a □web of offers□ The formalism further provides consolidated semantics of methods and notions in CE independently of realisations and marketing terms. It is advisable to apply the results from other research areas to CE. The proposed model formulates the future requirements of these research areas , which can be integrated in the CE research.

References

[AVEL+02] AVELLO, Daniel Gayo; GUTI□RREZ, Darō □lvarez: *The Cooperative Web. A Complement to the Semantic Web*. In: Proceedings of 26th Annual International Computer Software and Applications Conference (2002), S. 179-183.

[BECH+01] BECHHOFER, Sean; CARR, Leslie; GOBLE, Carole; HALL, Wendy: *Conceptual Open Hypermedia = The Semantic Web?* In: Proceedings of The Second International Workshop on the Semantic Web □SemWeb (2001).

[BERN+01] BERNERS-LEE, Tim; HENDER, James; LASSILA, Ora: *The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities*. In: Scientific American. May (2001), S. 34-43.

[FENS+03] FENSEL, Dieter; HENDLER, Jim et al. (eds.): *Spinning the Semantic Web.* Cambridge, Massachusetts, London, MIT Press, (2003).

[HOLS+02] HOLSAPPLE, Clyde W.; JOSHI, K. D.: *A Collaborative Approach to Ontology Design*. In: Communications of the ACM 45, 2 (2002), S. 42-47.

[HYV□+02] HYV□NEN, Eero (ed.): *Semantic Web Kick-Off in Finland. Vision, Technologies, Research, and Applications.* HIT Publications, (2003).

[LEGR+01] LE GRAND, B□n□dicte; SOTO, Michel: *Visualisation of the Semantic Web: Topic Map Visualisation.* Available at: http://www-rp.lip6.fr/site_rp/

[MAIC02] MAICHER, Lutz: *Evaluation von Topic-Maps zur Beschreibung von Wissensstrukturen als Grundlage für das Wissensmanagement in IT-Projekten.* Diploma-thesis at university of Leipzig, August (2002).

[OSSE+01] OSSENBRUGGEN, J.R. van; HARDMAN, H.L.; RUTLEDGE, L.: *Hypermedia and the semantic web: A research agenda.* In: Technical Report INS-R0105, CWI, (2001).

[OSSE+02] OSSENBRUGGEN, J.R. van; HARDMAN, H.L.: *Smart Style on the Semantic Web.* In: Proceedings of the International Workshop of the Semantic Web, Hawai (2002).

[REIM91] REIMER, Ulrich: *Einführung in die Wissensrepräsentation.* Stuttgart, Teubner (2001).

[ROMH98] ROMHARDT, Kai: *Die Organisation aus der Wissensperspektive – Möglichkeiten und Grenzen der Intervention.* PhD-thesis at university of Geneva, (1998).

[ROTH+01] ROTHFUSS, Gunther Elgar; RIED, Christian (Hrsg.): *Content Management mit XML: Grundlagen und Anwendungen.* Berlin, Heidelberg, New York (Springer) 2001.

[SCHA02] SCHATZ, Bruce R.: *The Interspace: Concept Navigation Across Distributed Communities.* In: IEEE Computer 35, 1 (2002), pp. 54-62.

[STAF97] STAFF, Christopher: *HyperContext: Using Context in Adaptive Hypertext.* In: Proceedings of the Sixth International Conference in User Modelling; Vienna, (Springer), 1997.

[THOM02] THOMPSON, Bryan: *The Cognitive Web.* Presentation to the Semantic Web Interest Group. Available at: http://www.cognitiveweb.org/publications

[WILD+00] WILDE, Erik; LOWE, David: *From Content-centred Publishing to a Link-based View of Information Resources.* In: Proceedings of the 33rd Hawaii International Conference on System Sciences, IEEE Computer Society Press, (2000).

Matching Algorithms in Alternative File-Sharing Systems to Find New Users Having New Content

Jürgen Nützel

Technische Universität Ilmenau, Institut für Theoretische und Technische Informatik,
D-98693 Ilmenau, Germany
Juergen.Nuetzel@TU-Ilmenau.de

Abstract. The distribution of new content in the Internet is simple and cheap. Even the consumers can act as distributors. Many peer-to-peer (P2P) systems show this effect dramatically. This happens in conflict with the traditional view of the content owners. Their business models do not accept users with equal re-distribution capabilities. We provide for the publishers of music and other virtual goods an alternative system which allows the users to play an active distribution part. Our approach which is called Potato System motivates the users to re-distribute content they have paid for and earn money with it. The Potato System pays for any re-distributed file a defined percentage on commission. This allows a fast distribution of new content. We added a matching functionality to the system. This functionality called Potato Match calculates a recommendation based on the files a user has paid for. Potato Match provides not a list of digital products; it provides a list of users. Potato Match supports users in two ways. It helps to find other users who share new content. And it assists those users who want to re-distribute content. The Potato System provides its own P2P clients which contact a central web-service to receive the needed matching information. At the end of the paper a distributed user matching functionally is discussed.

1 Motivation and Introduction

Thanks to modern compression techniques and increased bandwidth, the distribution of digital music, video and other virtual goods via Internet has become affordable and easy. In fact, it has become simple enough to allow anyone to act as a distributor. The traditional and centralized view of the publishers makes them believe that a free usage of digital content out of their control would undermine their business models. Music publishers therefore rely on so-called strong Digital Rights Management (DRM) systems which restrict and control the usage of content that has been legally downloaded and paid for [1],[2]. Many potential customers would pay for digital content without usage control. But providers restrict the usage of their digital products and treat their customers as enemies [3]. They focus on the misuse case only. This conflict blocks the development of a growing business on the Internet.

What can we do in order to put digital music publishing back on sound feet? Our idea is to focus on the "good" customers only. We propose an alternative approach which motivates the user to cooperate with the interests of the publishers and artists. We call this approach Potato System. The web-site [4] provides further information

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 180-188, 2003.

and papers around the Potato System and where the name did come from. In the Potato System we bring users und publishers back together on their common interests. But what are the common interests?

Content providers want to sell their products. And selling means distribution. Therefore, content providers have high interest in the distribution of their products. And as we know from peer-to-peer (P2P) systems [5] like KaZaA re-distribution is obviously in the interest of end-users. However, the products should be paid for. We suggest users to become official re-distribution partners. A customer, who pays for a product, gets the right to re-distribute it and earns money with it.

Why should a recipient pay for music files? The recipient likes the music and this way he supports an unknown musician. This is noble. He wants to get a reward, a percentage of the payment, which the next recipients pay for. This is not that noble but it is fair. If he does not pay, he will have no chance to get any reward later. The most important point in our system is that distribution and payment are not linked. Distribution is free of any technical restriction. Payment is optional. Commission is paid to the one who has paid for [6].

In the Potato System users have a high motivation to promote new content. They want to earn money or they simply want to find new content. At this point the community matching functionality of the Potato System is involved. This feature provides information about interesting content of other users. The Potato System recommends primarily users instead of content. This approach assists the users to promote and find new content.

2 The Potato System

In this section we describe the Potato System, which was invented at Fraunhofer AEMT [7] and the 4FriendsOnly Internet Technologies AG (4FO AG) [8]. The system is not limited to digital music. Any digital content (e.g. bitmaps, videos or software) could be managed by the Potato System. The 4FO AG will provide and run the components of described system.

The system description is divided into two subsections. In the first subsection we explain the uses-cases without a P2P system. The second subsection describes the role of the P2P functionality within Potato System.

2.1 The Provider or Artist Sells the Content Using a Payment System

First we have to describe how a content owner or artist brings his files into the Potato System community. The first use-case is called “content registration”. In this use-case three actors are involved. Let Fred, George and Potato play different roles in the content registration use-case. Fred is an artist or music producer. He produced a song which is ready to publish. The encoded song is named *mysong.mp3*. The file is located at Fred’s own web-server in a subdirectory which is unknown to the public. In the next step Fred contacts George. George runs a payment service for virtual goods. This service co-operates with the Potato System. Fred tells George’s payment service where to find the song. Fred defines a price (e.g. 1.10 Euro) and a price model. The price model defines the algorithm to calculate re-distributors commissions. Let us

suppose Fred defines a commission-rate of 50%. To complete this use-case George's server contacts the Potato System web-service [9]. George's server transfers the information to the web-service. George calculates a SHA1 hash from the file. The hash allows the Potato System web-service to check integrity in later use-cases of the file. If the file includes audio content a robust fingerprint like AudioID [10] could be added. Such a fingerprint allows Potato System to identify *mysong.mp3* even after down-sampling or other modifications. Potato System stores all these information and answers with a unique transaction number (TAN). In figure 1 *31881* is the TAN.

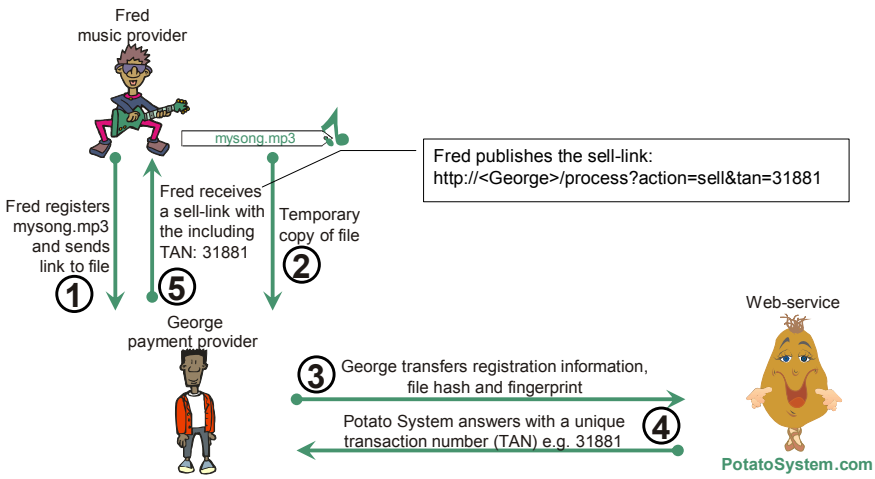


Fig. 1 Fred registers his file mysong.mp3 on the payment server of George. George cooperates with the Potato System.

The TAN is the receipt for successfully registration. Every TAN in the Potato System follows the same syntax. It starts with a customer number (here "3188") followed by a customer specific transaction number. The first digit (here "3") of the customer number defines how many digits follow. The customer transaction number is "1" because it is Fred's first transaction.

George uses the TAN to build a sell-link for Fred. Fred publishes this sell-link *http://<George>/process?action=sell&tan=31881* on his web-site. *<George>* stands for the address of George's payment server.

Figure 2 shows the next use-case. To simplify the description we suppose that Ginny already has login and customer number ("3712") from the Potato System. Let Ginny play the role of a fan who wants to buy the newest song of Fred. Ginny enters Fred's web-site and clicks the sell-link. The link leads Ginny to George's payment service. After successful payment George contacts the Potato System web-service to register Ginny's purchase. After this registration process Ginny is the re-distributor of Fred's song. The Potato System answers with a new TAN ("37121"). The TAN is the receipt for Ginny's purchase.

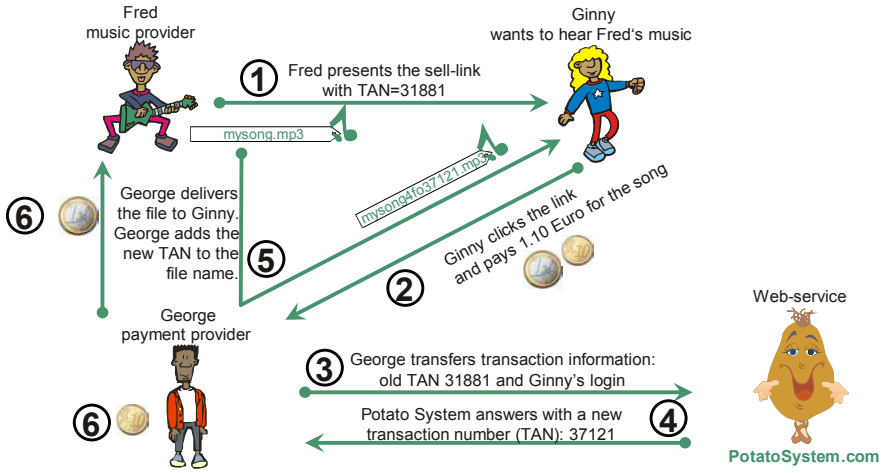


Fig. 2 Ginny pays for Fred's song using George's payment service. Ginny becomes an official re-distributor

While Ginny downloads the file via George's server George adds the new TAN to file name. The new file name is *mysong4fo37121.mp3*. File renaming is the easiest way to add a receipt to a file. Beside the file with the receipt Ginny receives also a sell-link like: *http://<George>/process?action=sell&tan=37121*. Ginny can also publish this like Fred. If a new user follows this link and pays, Ginny will receive her commission.

In [3] a Java-archive-like approach for the receipt is given. Similar approach to add buyer's identity is called "Light Weighted DRM" [2]. This approach uses a new file format which is called signed media format (SMF). SMF brings more comfort to users but it reduces interoperability and increases system complexity.

2.2 Potato Users Share and Pay Files in the Peer-to-Peer System

Ginny has several motivations to pay for Fred's song. The song was brand new and there was no other way to find the file. A second motivation was that Ginny wants to become a re-distributor. As a re-distributor Ginny sends her sell-link to her friend Harry. Or Harry finds this link on Ginny's home-page. If Harry buys the song using this link Ginny receives 50 Cents from Fred's revenue. This is a kind of affiliated marketing [11]. But if we follow only this use-case we do not really promote new content to new customers. This was the reason to provide Ginny a special P2P client. We call this client Potato Messenger. The client is a signed Java applet, which uses the open source framework JXTA [12].

Using the P2P Potato Messenger Ginny gets free access to content other Potato System users have paid for. But Ginny is only able to transfer content of limited value. The limit is (e.g.) 20-times of the amount she has paid for Fred's song. If Ginny wants to "test" more new music she needs to pay for one of the songs she already has "tested" and transferred. But Ginny has a second opportunity. She could

pay with the credits on her Potato System account. To earn more credits Ginny has to provide on her computer songs she has paid for.

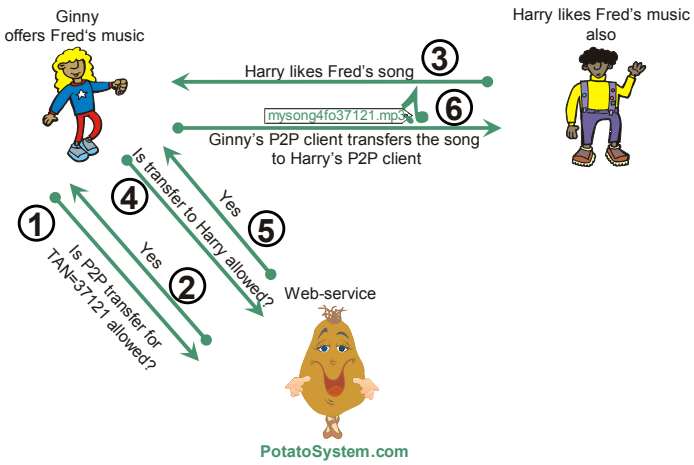


Fig. 3 Ginny uses the P2P system to transfer Fred's song to Harry. Harry receives a free copy of the song.

Figure 3 shows a use-case with Ginny, Harry and the central Potato System web-service. Ginny uses the Potato Messenger to offer Fred's song. To setup, the Potato Messenger contact the web-service to check what files Ginny is allowed to offer. The messenger sends for every file the TAN and the SHA1 hash. Optionally the messenger could send the AudioID [10]. Harry also uses the P2P client. He found Ginny's offer. But before Ginny's client is allowed to transfer the file to Harry the client has to ask the web-service. Potato System checks if Harry is allowed to make free file transfers because the free transfer is restricted by value.

At this point we have two questions. What users provide for Harry and Ginny more content they are interested in? And what songs are best to pay for?

3 The Matching Algorithm in the Potato System

In the Potato System every paying user becomes automatically an official re-distributor. This basic idea transforms the Potato System into a community based system. Every active user in the Potato System has a high motivation to contact other users with similar interests. The recommendation algorithm has to keep this special characteristic of Potato System in mind. Primarily the Potato System does not recommend products; it matches users with similar interests. In a second step Potato System might recommend a specific file to pay for. Such a recommendation would be based on economic assumptions. Actually we are not supporting such functionality. But we are developing one.

The Potato System is neither limited to specific music nor to music in general. To keep the Potato System as much as flexible no categories have been introduced. This makes it more difficult to match users with the same taste. In following text we de-

scribe step-by-step our matching algorithms which is called Potato Match. In [13] Frank Zimmermann describes in detail the shown algorithm, its implementation and many variations.

3.1 Ranking of Files

Many other music recommendation approaches try to find similarities in the content or in the meta data [14]. Potato System is focused on the purchase statistics stored in database of the central web-service. Potato System knows what files a user has paid for. Potato System knows also the time a file was registered in the Potato System. This enables the system to calculate several typical rankings:

- *Top 50* brings a list of 50 files which have been sold most.
- *Top 50 of the month* brings a list of 50 files which have been sold most last month.
- *New entries of the week* brings a list of the new files in the actual week.

These rankings are easily to calculate. The server has to do this only once a day. George and Fred can publish these rankings on their web-sites to attract new users for the Potato System. But this information is not really helpful for existing Potato System users. These users are interested in community based information. The Potato System matching functionality which is independent from content type tries to fill this gap.

3.2 The User/File Matching – Potato Match

In the first step we describe how Potato Match finds songs Ginny is interested in. To make our algorithm more clearly we use the example scenario from Table 1. The table shows what songs each user has bought. We see Ginny already bought the songs "S01", "S09" and "S10". Fred is the provider of the songs "S01"-"S05". Joe provides the songs "S06"-"S10".

Table 1. Example scenario of users and their songs.

User/*Provider*	Songs
Fred	*S01-S05*
Joe	*S06-S10*
Ginny	S01, S09, S10
Draco	S02, S03, S04
Mario	S01, S02, S03, S09
Alex	S01, S07, S08, S10
Frank	S01, S03, S04
Stephan	S01, S05, S06, S10
Carsten	S07, S08
Robert	S01, S03, S06, S09
Anja	S01, S05, S10
Julia	S01, S02, S03, S04

We suppose Ginny selects song “S01” for the matching algorithm because it is her newest song. The Potato Match calculates in the first step a list of users who bought or provide “S01” also. In our example the users Draco and Carsten and the provider Joe do not belong to this temporary list. In the second step all songs of the users from this temporary result are listed. Table 2 shows this list sorted by the frequency of occurrence. Songs which Ginny already has are shown in brackets.

Table 2. Relevant songs sorted by frequency of occurrence.

Song	Frequency
(S01)	9
S03	5
(S10)	4
(S09)	3
S02	3
S04	3
S05	3
S06	2
S07	1
S08	1

Song “S3” with 5 “points” is most relevant for Ginny. If Ginny wants to follow this recommendation she can buy the song from Fred’s web-site using George’s payment service (see fig. 2). We think this is not the best opportunity for Ginny. Ginny can contact (via P2P) other users of the Potato System to receive a free copy of “S3”. This is the point to calculate a user recommendation for Ginny. What is the best user for Ginny to contact? The Potato System calculates for Ginny a user rating. The user rating is based on the frequency (“points”) of the files in Table 2. The rating of Alex is 2. This is very low, because Alex has only two files in Table 1 which are new for Ginny. “S07” and “S08” bring only 1 point from Table 2.

Table 3. User rating for Ginny’s song S01

User/*Provider*	“Points”
Fred	*14*
Draco	11
Julia	11
Mario	8
Frank	8
Robert	7
Stephan	5
Joe	*4*
Anja	3
Carsten	2
Alex	2

As we see in table 3 Draco and Julia are most attractive for Ginny. These users have the songs “S03”, “S04” and “S02” which are the most popular in a virtual community around the song “S01”. If Ginny selects “S10”, the Potato System would rec-

ommend Alex and Carsten. Both users have the songs "S07" and "S08" which found twice (Joe and Alex) near the song "S10".

Currently we are discussing a lot of modifications for Potato Match. One modification is targeted on the function which calculates Table 2. Is it very clever to give songs with high frequency the most point? Maybe it is better to drop the files (here "S03") with the highest frequency, because Ginny would not find enough new users for "S03".

4 The P2P Version of the User Matching Algorithm

We have other problems with Potato Match. The calculation of Potato Match on a central database system is the bottleneck of the system [13]. This brings us to the idea to shift the time consuming part from the server to the client.

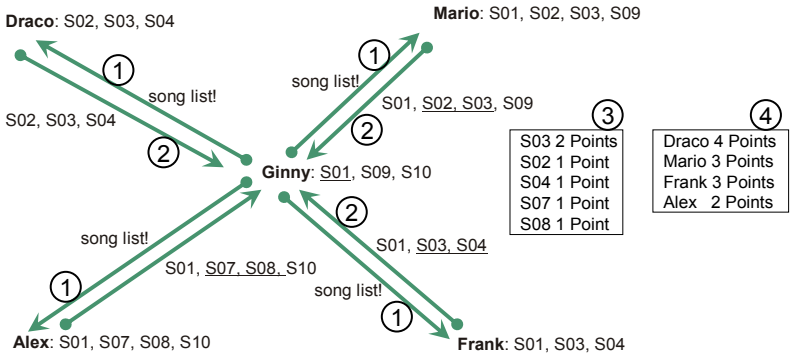


Fig. 4 Ginny asks all (four) reachable peers for their file list.

Figure 4 shows the distributed version of Potato Match. Ginny asks all currently reachable peers to send their file list. With this information Ginny's client calculates the file rating table. This table is used to calculate the final user rating.

In figure 4 we can illustrate the possible modification of the matching algorithm. If we drop the most popular song "S03", we get a different user rating for Ginny. In this case Alex and Draco receive 2 points, Frank and Mario only 1 point.

5 Conclusion and Further Work

In the Potato System, the consumption of multimedia products is linked to an incentive for payment. Both, the content provider and the consumer would gain an economic profit from the payment. Re-distribution of content without payment is still possible: it is neither forbidden, nor technically blocked. However, it is not attractive.

Those who pay and re-distribute are the users themselves. They would build up a decentralized distribution infrastructure, which may live in parallel with a centrally controlled distribution system of the content providers. The decentralized re-

distribution infrastructure of the users can be self-organized and grow stably bottom-up. Nobody loses, all win. This is possibly not an effective retail system yet. But it can follow the PGP example and provide a system of "pretty good distribution" with "pretty good profits".

More information will be publicly available (www.PotatoSystem.com) [4]. We are starting with a P2P Java applet using JXTA [12]. There are future plans to implement (and study) different matching algorithms. In field trials we will work on user acceptance. We need these field trials to collect statistical data. We provide further use-cases for the Potato System web-service. One of these use-cases works with point-of-sale systems and mobile devices.

References

1. 3GPP TS 22.242 V6.0.0 (2002-06). Digital Rights Management (DRM) Stage 1, Release 6, June 2002.
2. Neubauer, Ch.; Brandenburg, K.; Siebenhaar, F.: Technical Aspects of Digital Rights Management Systems, In 113th AES-Convention, Los Angeles, October 2002. Convention Paper 5688
3. Grimm, R., Nützel, J.: A Friendly Peer-to-Peer File Sharing System with Profit but Without Copy Protection. Intelligent Internet Computing Systems I2CS, Kühlungsborn, June 2002, Springer, LNCS 2346.
4. Homepage of the Potato System, www.PotatoSystem.com
5. Biedermann, K.: Gnutella und Co: Ein Praxisbericht über aktuelle Peer-to-Peer-Systeme, Hauptseminar (in German language), Technische Universität Ilmenau, May 2002. http://www.4friendsonly.org/ger/students.htm#biedermann
6. Grimm, R., Nützel, J.: Peer-to-Peer Music-Sharing with Profit but Without Copy Protection, Wedelmusic 2002 Conference, Darmstadt, 9th - 11th December 2002. In the Proceedings of Wedelmusic 2002, IEEE Computer Society, ISBN 0-7695-1623-8, http://www.4friendsonly.org/eng/papers.htm
7. Homepage of Fraunhofer AEMT, http://www.emt.iis.fraunhofer.de
8. Homepage of 4FriendsOnly Internet Technologies AG, http://www.4FriendsOnly.com
9. W3C: Web Services Architecture, 2002. Working Draft. http://www.w3.org/TR/2002/WD-ws-arch-20021114/
10. Allamanche, E.; Herre, J.; Hellmuth, O.; Fröba, B.; Cremer, M.: AudioID: Towards Content-Based Identification of Audio Material. In 110th AES-Convention, Amsterdam, 2001. Convention Paper 5380.
11. Krauß, H.; Konzeption und Realisierung der Server-Komponente für ein P2P-File-Sharing-System, bei dem die User am Umsatz beteiligt sind, Diploma Thesis (in German language), Technische Universität Ilmenau, August 2002. http://www.4friendsonly.org/ger/students.htm#krauss
12. Homepage of JXTA open source peer-to-peer framework, http://www.jxta.org
13. Zimmermann, F.: Entwurf und Implementierung eines User-File-Matching-Algorithmus unter Verwendung eines MySQL-Datenbanksystems im Potato-System, Studienarbeit (in German language), Technische Universität Ilmenau, February 2003. http://www.4friendsonly.org/ger/students.htm#zimmermann
14. Baumann, S.; Klüter, A.: Super Convenience for Non-Musicians: Querying MP3 and the Semantic Web, Proceedings of the ISMIR 2002, Paris, France, Oct 13-17, (2002).

A Fuzzy Kernel-Based Method for Real-Time Network Intrusion Detection

Mikhail Petrovskiy[1]

Computer Science Department of Lomonosov Moscow State University,
Building 2, MSU, Vorobjovy Gory, Moscow, 119899, Russia
michael@cs.msu.su

Abstract. Most existing intrusion detection systems use signature-based approach to detect intrusions in audit data streams. This approach has a serious drawback. It cannot protect against novel types of attacks. Thereby there is a growing interest to application of data mining and machine learning methods to intrusion detection. This paper presents a new method for mining outliers designed for application in network intrusion detection systems. This method involves kernel-based fuzzy clustering technique. Network audit records are considered as vectors with numeric and nominal attributes. These vectors are implicitly mapped by means of a special kernel function into a high dimensional feature space, where the possibilistic clustering algorithm is applied to calculate the measure of "typicalness" and to discover outliers. The performance of the suggested method is evaluated experimentally over KDD CUP 1999 data set.

1 Introduction

Security of Internet and Intranet systems has become extremely important recently, since more and more sensitive and privileged information is stored and manipulated online. Intrusion Detection is a powerful technology to help protecting the information from malicious actions and unauthorized access. Most existing Intrusion Detection Systems (IDSs) use the signature-based approach. Usually they involve expert knowledge hard-coded as a rule set. These IDSs match current activity on the network against a priori known attack scenarios. The rule set database has to be manually updated for each new type of attack. It leads to substantial latency in deployment of newly created signatures across the computer system. But the main problem of this type of IDSs is that they are not tolerant to new types of attacks, since they do not have predefined scenarios for them. Because of this, there is a growing interest to data mining and machine learning algorithms, which train on historical data. These algorithms build models of normal or abnormal behavior of system activities. The approach based on models of abnormal behavior is called *misuse detection*. It compares real-time system activities to generalized scenarios of attacks misuse detection models. The approach based on models of normal behavior is called

[1] Research is supported by RFBR grant # 03-01-00745

T. Bhm e, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 189-200, 2003.

anomaly detection. Unlike the misuse detection it compares current system activities to normal activity profiles anomaly detection models. In this approach anomalies are considered as possible intrusions [1].

The method discussed in this paper belongs to anomaly detection approach. In comparison to misuse detection the anomaly detection approach has several advantages. The first advantage is the higher ability to detect new types of attacks, since potentially any new attack differs from normal behavior. The second advantage is that for building anomaly detection models one does not need labeled patterns of attacks. It means that unsupervised techniques can be used. But anomaly detection approach has problems too. The main one is comparatively high rate of false alerts. The reason is clear: not every unusual activity is an attack.

Anomaly detection approach has been intensively studied since it was firstly suggested in [1]. Different learning techniques have been applied. First of all, traditional statistical outlier detection methods that based on probabilistic generative models [10]. Besides, a lot of other state-of-the-art data mining techniques such as decision trees, association rules, crisp clustering, k-nearest neighbor algorithms [9] and neural networks [11] are reported to show promising results. But in general, most of these methods have two important issues. The first one is training models on labeled data only. The second is the ability to handle the numeric attributes only. To avoid these problems Eskin et al. in [6] proposed kernel-based approach, which they called a *geometric framework for unsupervised anomaly detection*.

2 Feature Space for Anomaly Detection

In the geometric framework for unsupervised anomaly detection, developed in [6], data elements, representing network connection records, are mapped into a feature space, which is considered as $\Re^m$. In the feature space standard outlier detection algorithms are applied to discover outliers. Our method is within this framework too, but several significant improvements have been done. First of all, we suggested new feature map that leads to infinite dimensional Hilbert feature space. The second improvement is a novel fuzzy outlier detection algorithm, which is formulated and applied in the feature space.

2.1 Feature Spaces and Kernels

One of the key ideas of all kernel-based methods is mapping data instances from the input space X to a feature space H. The non-linear mapping is applied implicitly by means of a kernel function K. This technique is widely used in machine learning, e.g. in SVM, kernel PCA, kernel Fisher Discriminant and others [3]. The feature space is a high or infinite dimensional space. The map φ is called a *feature map*:

$$\varphi : X \to H \tag{1}$$

This map associates with every x from the input space the image $\varphi(x)$ in the feature space. The kernel function corresponds to dot product in the feature space:

$$K(x,y) = \langle \varphi(x), \varphi(y) \rangle_H \tag{2}$$

This definition of kernel function allows introducing of a distance metric as:

$$d(x,y) = \sqrt{K(x,x) - 2K(x,y) + K(y,y)} \tag{3}$$

The main advantages of applying kernel-based methods to network intrusion detection are as follows. Foremost, it allows handle network connection records geometrically. Thus it allows application geometrical algorithms formulated in terms of distance metric and dot product. Secondly, the kernel function K can be considered as a similarity measure for records in the input space. The freedom to choose the feature map φ and corresponded kernel function K enables us to design a large variety of similarity measures and analysis algorithms. It is called "kernel trick". Suppose given algorithm formulated in terms of distance metric or dot product, and one substitutes the distance metric with "kernelized" distance metric or replace the dot product by another kernel function. As a result the new algorithm with new properties and possibly better performance is formulated. Thirdly, using kernels allows us to work with feature map implicitly. It means that there is no need to calculate and store high dimensional vectors of images $\varphi(x)$.

2.2 Designing Kernel for Anomaly Detection

The choice of the feature space, i.e. the choice of the kernel function is application specific and greatly depends on the ability of the feature space to capture the information, relevant to the application domain. In [3,4] Smola, Vapnik et al. investigated this problem for several application domains. They suggested that the good performance in outlier detection could be achieved with Gaussian kernel:

$$K(x,y) = e^{-q(x-y)^2} \tag{4}$$

where q is a parameter, controlling the kernel width.

It is obvious that Gaussian kernel cannot be applied directly to the network connection records data. There are two reasons. First of all, there are nominal attributes like protocol name, connection flags, etc. For nominal attributes we use standard data mining approach: each nominal attribute that takes n different values we consider as n numeric (actually binary) attributes. The second problem is that attributes may have different ranges of values. To avoid this problem we use data-dependent normalization kernels. All attributes values are normalized to the number of standard deviation away from mean. Taking it into account we can define the kernel for network connection records as a product of kernels, defined for each attribute:

$$K(x,y) = \prod_{i \in Num} e^{-q_i \frac{(x_i - y_i)^2}{\sigma_i^2}} \prod_{j \in Discr \wedge x_j \neq y_j} e^{-\frac{q_j}{N_J^2}\left(\frac{1}{P(x_j)(1-P(x_j))} + \frac{1}{P(y_j)(1-P(y_j))}\right)} \tag{5}$$

where x_i is a value of *i*-th attribute of record *x*; *Num* is the set of indexes for numeric attributes; *Discr* is the set of indexes for nominal attributes; $P(x_i)$ is a probability of the value x_j for *j*-th nominal attribute; N_j is a size of the domain for *j*-th nominal attribute; σ_i is a dispersion of the *i*-th numeric attribute and q_i is a kernel width parameter. It is important to note that q_i can be considered as an importance weight of *i*-th attribute. An expert to tune the algorithm can control this parameter. Initially Eskin et al. in [6] suggested the simpler kernel that leads to $\Re^m$ feature space, but in our case the feature space generated by (5) is infinite dimensional Hilbert space.

3 Anomaly Detection Algorithms in the Feature Space

Algorithms for outlier detection in the feature space are based on the assumption that some probability distribution generated the data exists and the feature map is topologically correct. That is high density regions from the input space are mapped into high density regions in the feature space. The elements from the input space are labeled as outliers if their images in the feature space lie in low-density regions [3]. Thus the idea of most outlier detection algorithms in the feature space is to examine the image location whether or not it lies in sparse region.

3.1 Support Vector Clustering and Kernelized Distance-Based Outlier Detection Algorithms

Eskin et al. in [6] studied the performance of three well-know kernel-based outlier detection algorithms in application to intrusion detection. They investigated kernelized versions of two standard distance-based outlier detection algorithms, involving k-nearest neighbors (KNN) and crisp fixed-width clustering techniques. Informally speaking, k-nearest neighbors outlier detection algorithm labels as outliers the points having "small" number of "neighbors" and crisp cluster-based algorithm labels as outliers the points lying "far" from centers of "big" clusters. In kernelized versions of these algorithms the Euclidian or other distance metric is substitute with kernelized distance metric (3). The third algorithm they studied is one of the most efficient outlier detection algorithms working in the feature space. It is Support Vector Clustering (SVC) algorithm [3,4]. It computes the binary function, which is supposed to capture regions in the input space where the probability density is in some sense large; i.e. the function is nonzero in a region where most of the data located. Informally the idea of this algorithm can be described as follows. The data instances from the input space are mapped by means of the kernel function into high dimensional feature space where the algorithm searches the sphere with minimal radius enclosing "the most part" of images of data. The size of this "most part" is controlled by the special parameter ν. The data instances which images lie outside the sphere in the feature space are labeled as an outliers.

Though these tree algorithms demonstrated acceptable detection and false positive rates [6], the practical application of them in real IDSs faces serious problem. The

performance of these algorithms strongly depends on a priori set parameters. They are:

- size and width of the neighborhood for KNN;
- width of cluster for Cluster-based algorithm;
- quantile ν for SVC.

Changing these parameters implies recreating and retraining of the models. Although parameters estimation methods for these algorithms do exist, they are heuristic, computationally expensive and ineffective in practice. Because they require step-by-step models recreation with new settings, but on real data of medium size the model creating and training may take hours and even days. Besides, the algorithms have binary decision functions or binary decision rules and these critical parameters actually define the outlier criterion or the outlier factor. It means that to change the outlier criterion we have to recreate and retrain the models.

3.2 Kernel-Based Fuzzy Algorithm for Anomaly Detection

To avoid these problems we suggest fuzzy approach. The presented method is hybrid method involving fuzzy and kernel-based techniques. Recently several contributions in this area have been published, for instance, the Fuzzy SVM method for multi-class classification problem [7], or EM fuzzy clustering algorithm in the feature space [5]. Our method inherits the ideas of SVC, but instead of looking for the crisp sphere in the feature space, we suggest to search fuzzy sphere including all data images. This problem can be considered as calculating single fuzzy cluster in the feature space using possibilistic fuzzy clustering approach [2]. In this case the fuzzy membership can be described as a measure of "typicalness" of audit data instances. The network connection records with low "typicalness" are considered as outliers. Changing threshold or in other words changing the outlier criterion does not lead to recalculating the models, as it was for SVC and kernelized distance-based algorithms. Mathematically the problem is formulated as follows:

$$\min_{U,a,\eta} J(U,a,\eta)$$

$$J(U,a,\eta) = \sum_{i=1}^{N} (u_i)^m (\varphi(x_i) - a)^2 - \eta \sum_{i=1}^{N} (1-u_i)^m \tag{6}$$

where a is a center of the fuzzy cluster in the feature space; N is a number of instances in X; U is a membership vector, where $u_i \in [0,1]$ is membership of the image $\varphi(x_i)$ and besides the "typicalness" of datum x_i; m is fuzzyfier and η controls the distance where the membership becomes 0.5. It is necessary to note that unlike traditional possibilistic fuzzy clustering approach in the input space [2] neither cluster center a nor $\varphi(x_i)$ can be calculated explicitly, but it is easy to shown that $J(U,a)$ can be minimized by simple iterative algorithm, formulated in terms of kernels.

Kernel-based Fuzzy Algorithm for Outliers Detection

/* l – iteration counter*/

Step 0. Initialize membership vector U and parameter η

$$u_n^{(0)} = 1$$
$$\eta^{(0)} = \max_{n\in[1,N]} d^2(x_n, a) = \max_{n\in[1,N]}\left[\sum_{j=1}^{N}\sum_{i=1}^{N} K(x_i, x_j) + K(x_n, x_n) - 2\sum_{i=1}^{N} K(x_i, x_n)\right] \quad (7)$$

REPEAT

Step 1. For all x calculate the distance to cluster center

$$d^2(x_n, a) = (\phi(x_n) - a^{(l)})^2 = (< a^{(l)}, a^{(l)} > + K(x_n, x_n) - 2 < a^{(l)}, \phi(x_n) >) \quad (8)$$

where

$$< a^{(l)}, a^{(l)} >= \left(\sum_{j=1}^{N}(u_j^{(l)})^m \sum_{i=1}^{N}(u_i^{(l)})^m K(x_i, x_j)\right) \Big/ \left(\sum_{i=1}^{N}(u_i^{(l)})^m\right)^2$$
$$< \phi(x_n), a^{(l)} >= \left(\sum_{i=1}^{N}(u_i^{(l)})^m K(x_i, x_n)\right) \Big/ \left(\sum_{i=1}^{N}(u_i^{(l)})^m\right) \quad (9)$$

Step 2. Update U

$$u_n^{(l+1)} = \left[1 + \left(d^2(x_n, a^{(l)}) \big/ \eta^{(l)}\right)^{1/(m-1)}\right]^{-1} \quad (10)$$

UNTIL $\left\|U^{(l)} - U^{(l+1)}\right\| < \varepsilon$

The decision function for new connection record x is:

$$u(x) = \left[1 + \left(\frac{\sum_{j=1}^{N} u_j{}^m \sum_{i=1}^{N} u_i{}^m K(x_i, x_j)}{\eta(\sum_{i=1}^{N} u_i{}^m)^2} - 2\frac{\sum_{i=1}^{N} u_i{}^m K(x, x_i)}{\eta\sum_{i=1}^{N} u_i{}^m} + \frac{K(x,x)}{\eta}\right)^{1/(m-1)}\right]^{-1} \quad (11)$$

where N is a number of records in training set; u_i is membership of the record x_i.

The main advantage of presented algorithm is smooth decision function. That is why changing the outlier criterion does not lead to model recreating and retraining. The outlier criterion is just a threshold and does not affect the calculating of the measure of "typicalness". Another advantage of presented algorithm is that it is simpler than SVC from computational point of view. SVC requires solving quadratic programming problem [3, 4].

The first version of the algorithm was designed to work with constant η. But it would be desirable to include in the algorithm another important parameter ν, which controls the fraction of noise in training data set. As a result the problem statement (6) can be reformulated:

$$\min_{U,a,\eta} J(U,a,\eta)$$
$$J(U,a,\eta) = \sum_{i=1}^{N} (u_i)^m (\varphi(x_i) - a)^2 - \eta \sum_{i=1}^{N} (1-u_i)^m \quad (12)$$
$$\text{subject to } (\varphi(x_i) - a)^2 \geq \eta \text{ for } \nu N \text{ elements from } X$$

J minimization algorithm will be changed slightly. New step will be added after step 1 and before step 2:

New Step. Adaptive η estimation

$$\text{Let } d^2(x_{n:1}, a^{(l)}) \leq d^2(x_{n:2}, a^{(l)}) \leq \ldots \leq d^2(x_{n:N}, a^{(l)}) \text{ then}$$
$$\eta^{(1)} = d^2(x_{n:N(1-\nu)}, a^{(l)}) \quad (13)$$

In this case ν fraction of data elements will have the membership lower than 0.5, others will have membership higher than 0.5. It is very useful parameter because usually in intrusion detection we cannot guarantee that training set contains only normal data. But we can be sure that there is less that ν % of unknown attacks in it. That is why this parameter is very important.

4 Real-Time Intrusion Detection

Computational complexity of training and evaluation stages is one of the serious issues obstructing application of sophisticated data mining and machine learning algorithms to real-time network intrusion detection [12]. The training time for models creation is not so crucial, because effective sampling and filtering methods can be applied to the training set [9, 13]. But the performance of evaluation stage is extremely important. In real environment new events accrue in audit streams with a very high speed. The evaluation stage speed in our algorithm is the speed of calculation the measure of "typicalness" (11). It is obvious that it depends on the size of training set *N*. The only solution here is to reduce it, possibly loosing the decision function accuracy. It is important to note that this problem is common for mostly all machine-learning methods. It is called Reduced Set (RS) problem. This problem is discussed for statistical methods [15], for distance-based outliers mining algorithms [14], and for Support Vector Machines in [3].

4.1 Reduced Set Problem

For mostly all kernel-based methods the solution *a* and decision function *f* are presented in the form:

$$a = \sum_{i}^{N} \beta_i \varphi(x_i); f(a,x) = f\left(\sum_{i}^{N} \beta_i K(x_i, x) \right) \quad (14)$$

In our case:

$$\beta_i = (u_i)^m \Big/ \sum_j (u_j)^m \tag{15}$$

$$f(a,x) = \left[1 + \left((< a,a > + K(x,x) - 2 < \varphi(x), a >)/\eta\right)^{1/(m-1)}\right]^{-1}$$

The reduced set problem is formulated as approximation of solution a (14) by a':

$$\min \left\| a - a' \right\|$$

$$a = \sum_i^N \beta_i \varphi(x_i),\ a' = \sum_j^L \alpha_i \varphi(z_j), L << N \tag{16}$$

So, there are to subproblems have to be solved. The first one is to select the reduced subset $\{z_j\} \subset X$ and the second is to find expansion coefficients α_i. Reduced set construction methods are divided into two general categories: "Global" (PCA-based, regression-based, l_2 penalization) and "Greedy" methods. The methods from the former group solve both subproblems simultaneously and find in some sense optimal solution, the methods from the latter group perform greedy heuristic selection of subset $\{z_j\}$ and then expansion coefficients calculation using precise formulas:

$$\alpha = (K^z)^{-1} K^{xz} \beta, where\ K_{ij}^z = \left\langle \varphi(z_i), \varphi(z_j) \right\rangle and\ K_{ij}^{zx} = \left\langle \varphi(z_i), \varphi(x_j) \right\rangle \tag{17}$$

Global methods are very complex and computationally expensive. But greedy methods are computationally effective and usually achieve satisfactory results, though they are less theoretically correct and do not find optimal solution.

4.2 Greedy Clustering Algorithm for RS Selection

In our method we choose greedy approach. In particular, we apply greedy clustering algorithm in *X*, and clusters prototypes form the reduced set. The idea of greedy clustering algorithm is based on *r*-clustering algorithm proposed by Ruspini in [16]. As a reflexive, symmetrical similarity relation *r* defined on the set *X* we take kernel function (5). In this case fuzzy *r*-cluster with prototype *c* is a fuzzy set r_c in *X* such that for every *x* from *X* $r_c(x) = r(c,x) = K(c,x)$. The idea of Ruspini's subtractive clustering algorithm is simple. It selects the best *r*-cluster prototype *c*, according to aggregation function criterion, removes it from the analyzed set and looks for next cluster candidate. Until some specified stopping criterion is satisfied. In our case we use as a stopping criteria the distance in the feature space to the precise center *a*. It means that the stopping criterion is acceptable approximation of the solution *a*: $\left\| a - a' \right\| < \varepsilon$. As an aggregation function we use Sugeno fuzzy integral [17] with respect to our measure of "typicalness" *u(x)*. It is important to note that initial aggregation criterion, suggested by Ruspini, was the maximum power of cluster, associated with selected prototype. Ruspini's criterion is a special case of our if we suppose *u(x)* to be a constant. It means that our subtractive clustering algorithm

iteratively chooses *r*-cluster that covers the maximum number of the most "typical" data instances, removes them from analyzed set and looks for next the most "typical" *r*-cluster. After the stopping criteria is satisfied the *r*-cluster prototypes, selected by our algorithm form the reduced set $\{z_j\}$ and expansion coefficients are calculated using (17).

5 Experiments

In this section the performance evaluation experiments are presented. We apply our method to network connection records from the etalon dataset, which is used by mainly all researchers over the world to analyze performance of intrusion detection algorithms. It is KDD Cup 1999 Data [8], which contains a wide variety of intrusions simulated in military network environment. Standard performance evaluation measures for intrusion detection algorithms are used. They are *detection rate* and *false positive rate.* The detection rate is defined as the number of intrusions detected by the method divided by the total number of intrusions in the dataset. The false positive rate is defined as total number of normal connection records incorrectly labeled as intrusions divided to total number of normal connection records in the dataset. The trade-off between these rates precisely characterizes the ability of the method to discover intrusions.

5.1 Experimental Setup

For the experiments "10% version" of well-known MIT Lincoln Labs KDD Cup 1999 data set [8] is used. This dataset was obtained by simulating a large number of different types of attacks, with normal activity in background. It consisted of approximately 500,000 data instance in training dataset and 300,000 in test dataset. Test data has attack types that are not present in the training data. Train set contains 22 attack types. Test data contains additional 17 new attack types that belong to one of four main categories: DOS - Denial of Service (e.g. syn flood); Probe - surveillance and other probing (e.g. port scanning); U2R - unauthorized access to root privileges (e.g. password guessing); R2L - unauthorized remote login to machine (e.g. buffer overflow). Each data instance presents the single connection record. It has 3 types of attributes: basic features of TCP connection (e.g. duration, protocol, number of transferred bytes); content features within a connection suggested by domain knowledge experts (e.g. number of failed login attempts); time-based traffic features. We run our method over KDD Cup 1999 dataset using 30% random sampling. The kernel function is (5), all kernel width parameters are set to 1 for all attributes. The training data set was filtered using technique, described in [6], i.e. 1% of attacks left in training dataset against 99% of normal connections. The upper limit of reduced set size is set to 35 records.

5.2 Experimental Results

We will compare the results achieved by our method with results of two groups of the intrusion detection methods. Group I contains misuse intrusion detection algorithms based on different machine learning techniques. They are:

- RIPPER algorithm. It involves association rules mining and decision trees classifier techniques.
- 1-NN classifier. It is state-of-the-art classification algorithm based on nearest neighbor classification method.
- SNN clustering. It implements nearest neighbor clustering algorithm.

The details on these algorithms, dataset filtering techniques and settings for them can be found in [9].

Table 1. The results of experiments with Group I algorithms are presented in the table. Detection rate results are presented separately for each attack category.

Algorithm	DOS	U2R	R2L	Probe	False Positive
RIPPER	99%	84%	96%	98%	4%
1NN classifier	88%	40%	93%	50%	3%
SNN clustering	99%	60%	91%	94%	3%
Our Fuzzy Method (RS=35)	**97%**	**96%**	**51%**	**98%**	**5%**

The Group II contains algorithms from the class, i.e. kernel-based anomaly detection algorithms studied by Eskin et al. [6]. There are three kernel-based algorithms: modified kernel-based version of k-nearest neighbor algorithm; modified kernel-based version of fixed width crisp clustering algorithm; and Support Vector Clustering algorithm. They are trained on the same dataset with same filtering settings.

Table 2. The results of experiments with Group II algorithms are presented in the table.

Algorithm	Detection Rate	False Positive Rate
Cluster	93%	10%
k-NN outliers	91%	8%
SVM	98%	10%
Our Fuzzy Method (RS=35)	**94%**	**5%**

We can see that the performance of our method is nearly the same as the performance of the algorithms included in Group I, though the false positive rate is a little bit worse. It is promising result because algorithms in Group I based on misuse detection approach and that is why they must have better false positive rate. Regarding to algorithms included in Group II, the precision of our algorithm is better than precision of standard outlier detection algorithms, running in the feature space and the second to SVC. On the other hand, the false positive rate of our method is better among all algorithms in the group.

6 Conclusions

This paper presents a novel fuzzy kernel-based method for outlier detection. It is designed for application in real-time anomaly detection IDSs. It is based on the geometric framework for unsupervised anomaly detection proposed in [6], but has several improvements. In our method new data-dependent kernel is designed. Unlike in [6] the designed kernel reproduces the infinite dimensional feature space, where a novel fuzzy algorithm is applied to find outliers. This algorithm is based on ideas of SVC, but instead of margin estimation it involves a possibilistic fuzzy clustering approach. In the feature space it calculates the measure of "typicalness" for connection records. The records are considered as outliers if their "typicalness" is smaller than a specified threshold. The benefits and drawbacks of the suggested approach are discussed in the paper. The main benefit is smooth decision function and that is why changing outlier criterion does not lead to recreating models. The greedy reduced set selection algorithm is designed to support run-time mode t. It is important to note that in geometric framework for unsupervised anomaly detection [6] the problem of run-time mode was not investigated and even was not discussed. The performance of the suggested method is evaluated experimentally over data from KDD Cup 1999 dataset. The experiments demonstrated that the results of suggested method are very close to the results of the best misuse and anomaly intrusion detection algorithms. Besides, there are still possibilities to tune parameters and improve performance of the method. The algorithm was implemented as OLEDB for Data Mining provider and can be integrated in various MS Windows based Intrusion Detection Systems.

References

1. Denning D.E.: An intrusion detection model. IEEE Transactions on Software Engineering, SE-13 (1987), 222-232
2. Krishnapuram, R., Keller, J. M.: A Possibilistic Approach to Clustering. IEEE Trans. Fuzzy Systems. Vol. 1. No. 1 (1993), 98-110
3. Scholkopf, B., Smola, A., J.: Learning with kernels: Support Vector Machines, Regularization, Optimization and Beyond. The MIT Press Cambridge, Massachusetss (2000)
4. Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V.: Support vector clustering. Journal of Machine learning Research, Vol. 2, (2001), 125-137
5. Girolami, M.: Mercer Kernel Based Clustering in Feature Space. IEEE Transactions on Neural Networks, 13 (4), (2001), 780 - 784
6. Eskin, E., Arnold, A. , Prerau, M., Portnoy, L., Stolfo, S.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. Applications of Data Mining in Computer Security, Kluwer, (2002)
7. Takuya Inoue, Shigeo Abe: Fuzzy Support Vector Machine for Pattern Classification. In Proc. of IJCNN, (2001) 1449-1455
8. The third international knowledge discovery and data mining tools competition dataset KDD99-Cup http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, (1999)
9. Kumar V.: Data Mining for Network Intrusion Detection, Presentation at NSF Workshop on Next Generation Data Mining, Nov 1-3, (2002)

10. Eskin, E,: Anomaly detection over noisy data using learned probability distributions. In Proceedings of the International Conference on Machine Learning (2000)
11. Ghosh A., Schwartzband A. : A study in using neural networks for anomaly and misuse detection. In Proceedings of 8th USENIX Security Symposium, (1999)
12. Lee, W., Stolfo, s., Chan, P., Eskin, E., Fan, W., Miller, M., Hershkop S., Zhang, J.: Real Time Data Mining-based Intrusion Detection. In Proceedings of DISCEX II. (2001)
13. Balcazar, J., Dai, Y., Watanabe O.: A random sampling technique for training support vector machines, in Proceeding of Algorithmic Learning Theory (ALT‐01), 2001.
14. Jin, W., Tung, A., Han, J.: Mining top-n local outliers in large databases. In 7th ACM SIGKDD International Conference on KDD and Data Mining, (2001) 293-298
15. Girolami, M., He, C.: Probability Density Estimation from Optimally Condensed Data Samples. Computing & Information Systems Technical Reports, ISSN-1461-6122 (2002)
16. Ruspini, E.H. : Recent developments in fuzzy clustering. In R.R. Yager (Ed.): Fuzzy Set and Possibility Theory: Recent Developments. Pergamon Press, New York, 133-147 (1982)
17. Marichal, J.-L.: On Sugeno integral as an aggregation function", Fuzzy Sets and Systems 114 (2000) 347-365

Testing Protocol Robustness

Antoine Rollet and Hacène Fouchal

Département de Mathématiques et Informatique,
Université de Reims Champagne-Ardenne,
BP 1039 F-51687 Reims Cedex, France,
{Antoine.Rollet,Hacene.Fouchal}@univ-reims.fr

Abstract. Some new protocols as multimedia or wireless protocols are constrained and sometimes critical. We need to ensure their correct functioning before their development. They handle time constraints to model important aspects (delays, timeouts). This issue should be considered in specification language used to model such protocols. This paper presents a methodology for the development of reliable timed systems in general. It might be used to develop complex protocols.
We use the RT-LOTOS language as a high level model and we use the timed automata model as a low level model. This later is the basis of our validation technique. We collect all possible errors on such systems and show how to integrate them in some automated derived test sequences in order to observe the system reactions when it executes faulty behavior. Our aim is to observe the robustness of the whole system in presence of simulated errors.

Key-words : Robustness Testing, Validation, Protocol Testing, Timed Automata, Automata Theory.

1 Introduction

In the software or hardware development, conformance testing is highly needed in order to avoid catastrophic errors and to tackle the industrial development of the product with confidence. Since few years, time is considered as a crucial feature of many sensitive systems as multimedia protocols, embedded systems, air traffic systems. Then it should be seriously considered by designers and developers. This study deals with complex systems described as Input Output Timed Automata (defined as automata where each transition can bear either an input action or output action with timing constraints in some cases). Our work is inspired by the protocol engineering area where people usually deal with two main validation techniques:

– the *verification approach*, which handles the system specification and tries to prove its correctness (in this case the system is a white box). Usually, the user properties are expressed by another formalism as temporal logics and must be verified on the specification by using a model-checker for example,

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 201–215, 2003.

– the *testing approach*, which uses the implementation of the system and tries to find any faulty behavior on it without having a priori any information about the structure of the system (in this case the system is a black box). The test generation produces sequences of inputs (actions) from the specification, and the implementation must be able to execute these sequences (called *'test sequences'*) and to answer with the expected outputs.

In this paper, we will suggest a methodology able to deal with many steps in system development: High-level modelling, Low-level modelling, Conformance testing and Robustness testing.

The high-level modelling is performed by the RT-LOTOS language [CO95], [ISO97], [Led92], [LL93]. This language is an extension of LOTOS [BB89] (Language Of Temporal Ordering Specification) defined by the ISO [ISO87]. It describes any system by a process algebra, i.e., any system is seen as a mathematical formulae. Each system component is a part of this formulae. A formal semantics of this formalism are formally defined and the semantic model is the LTS model(Labelled Transition System).

Our low-level model is timed automata defined in [AD94]. Here the whole system is seen as a graph where edges express actions or reactions of the system. On edges, we may have also some timing constraints which are related to the action execution. The translation of specifications from RT-LOTOS to low-level models uses mainly rules defined in [Kar92]

We have defined formally the possible set of different faults that any system can perform. We show how to integrate these errors in test sequences in such a way to check reactions of the system when it performs faulty actions. In fact, we intend to test, in addition to the conformance testing, the robustness of the system in a controlled way.

This paper is structured as follows:

Section 2 contains related works to the timed testing field. In section 3, we describe briefly the RT-LOTOS language as well as the timed automata model and its main features. Section 4 explains our technique on robustness testing. We first collect all possible errors and we show how to integrate them in test sequences. Section 6 gives the conclusion and some ideas about future works.

2 Related Work

There are many works dedicated to the verification of timed automata [ACH94], [DOY94], [DY95]. Some tools [DOTY95, BLL+98] have been developed for this purpose.

There are also other studies which proposed various testing techniques for timed systems. We will give an overview of them in the following section. But we will detail only one study about embedded system testing since the literature about this issue is quite rare.

[Kon95] deals with an adaptation of the canonical tester for timed testing and it has been extended in [LC97]. In [CL97], the authors derive test cases from

specifications described in the form of a constraint graph. They only consider the minimum and the maximum allowable delays between input/output events. [COG98] presents a specific testing technique which suggests a practical algorithm for test generation. They have used a timed transition system model. The test selection is performed without considering time constraints. [RNHW98] gives a particular method for the derivation of the more relevant inputs of the systems. [PF99] suggests a technique for translating a region graph into a graph where timing constraints are expressed by specific labels using clock zones. [NS01] suggests a selection technique of timed tests from a restricted class of dense timed automata specifications. It is based on the well known testing theory proposed by Hennessy in [DNH84]. [HNTC01] derives test cases from Timed Input Output Automata extended with data. Automata are transformed in a kind of Input Output Finite State Machine in order to apply classical test generation technique. [SVD01] gives a general outline and a theoretical framework for timed testing. They proved that exhaustive testing of deterministic timed automata with a dense interpretation is theoretically possible but is still difficult in practice. They suggested to perform a kind of discretization of the region graph model (which is an equivalent representation of the timed automata model). Clock regions are only equivalence classes of clock valuations. Their discretization step size takes into account the number of clocks as well as the timing constraints. Then they derive test cases from the generated model. The second study [ENDKE98] differs from the previous one by using discretization step size depending only on the number of clocks which reduces the timing precision of the action execution. The resulting model has to be translated into a kind of Input/Output Finite State Machine which could be done only under strong and unrealistic assumptions. Finally they extract test cases by using the Wp-method [FBK+91].

As we notice, there are different ways to tackle the problem of timed testing. All of these studies focus on the reduction of the specification formalism in order to be able to derive test cases feasible in practice. In contrast to these studies, we use the timed automata model without neither translation nor transformation of labels on transitions.

3 Models

3.1 High-Level Modelling

RT-LOTOS (Real Time LOTOS) [CO95] is a temporal extension of the standard description technique LOTOS. RT-LOTOS is useful to describe complex critical systems with time constraints, with a high level of concurrency, such as real time embedded systems or multimedia protocols. It is an upward compatible extension of LOTOS, and so belongs to the process algebras family. The power of these algebras is to make possible to express formal specifications at different abstraction levels and to have a lot of theoretical framework on behavior equivalences.

LOTOS LOTOS (Language of Temporal Ordering Specifications) is a formal description technique standardized at ISO (ISO 8807) based on both CCS and ACT-ONE. The concept of LOTOS is to specify a system by expressing the relations among the interactions that constitute their externally observable behavior.

A LOTOS specification describes a system with a hierarchy of process definitions. A distributed system is seen as a process which can contain several subprocesses, and each subprocess is itself a process. A process is an entity able to perform internal, unobservable actions, and to interact with other processes which form its environment.

In fact, LOTOS implements a "black box" approach: it is possible to express the interactions of a process with its environment without having to describe its internal structure or implementation. Process definitions are described with expressions with operators, using recursion and multi-way "rendez-vous" mechanism which represents the basic communication facility between processes. Among the operators, action prefixing choice, parallel composition and hiding play an important role. Furthermore, in addition to the process interactions with synchronization, LOTOS allows value exchanges.

RT-LOTOS The weak of LOTOS is that only the "qualitative" ordering of events (ie occurrences of actions) can be expressed. Thus, the "quantitative" aspect of the time at which the action occurs is not provided. As the number of systems with time constraints such as multimedia protocols or embedded real time systems is increasing nowadays, the timing aspects of LOTOS became a need.

RT-LOTOS was inspired by Timed LOTOS [LL93] and T-LOTOS [BL92]. Many assumptions have been decided to preserve as much convergence as possible with ET-LOTOS, [BDS95], the successor of Timed LOTOS.

In summary, RT-LOTOS is useful to express several time-constrained behaviors, with some features such that:

- delay the occurrence of observable and internal actions (see delay operator Δ^t)
- express time non-determinism (see latency operator Ω^t)
- limit the time during which an observable action may be offered to its environment
- measure and store into a variable, which may be later referenced to in the specification, the time at which some action actually occurred (@ operator)
- associate a temporal violation recovery mechanism with some observable actions of the specification.

As a consequence, RT-LOTOS seems to be an interesting formalism for real time systems since it provides a model with actions, timing aspects, data and communication possibilities. Moreover, it is possible to use several tools already developed, such as RTL (RT-LOTOS Laboratory) which offers large possibilities such as simulating our system, validation tools or translation applications into

other formal representations like the Dynamic Timed Automata or even the Timed Automata.

A Simple Example Now, we will show a simple example of RT-LOTOS specification.

```
specification MEDIUM : noexit :=
behaviour
  hide iu_s, iu_d in
  let period : nat = 30000 in

  stream_sender[iu_s](0,period)
  |[iu_s]|
  medium[iu_s,iu_d](14000,20000)
  where
    process stream_sender [iu_s] (n : nat, period : nat) : noexit :=
      iu_s0!n; delay(period) stream_sender[iu_s] (n+1, period)
    endproc
    process medium [n_in,m_out] (dmin, dmax : nat) : noexit :=
     m_in?x:nat; delay(dmin,dmax)m_out!x;
     medium[m_in,m_out](dmin,dmax)
    endproc
endspec
```

The specification MEDIUM describes a situation where some periodic stream is sent (process `stream_sender`) through a one-slot medium (process `medium`) with a transmission delay belonging to interval [14ms,20ms]. With each information unit is associated an integer sequence number. The stream information units, assumed to be submitted by the environment on action `iu_s`, are delivered by the medium on action `iu_d`. Due to the one-slot assumption, the non deterministic transmission delay is chosen to be less than the period (30ms).

3.2 Low-Level Modelling

In this section, we will recall the definitions of *timed input output automaton*. Timed input output automata have been proposed to model finite-state real-time systems. Each automaton has a finite set of *states* and a finite set of *clocks* which are real-valued variables. All clocks proceed at the same rate and measure the amount of time that has elapsed since they were started or reset. Each transition of the system might reset some of the clocks, and has an associated enabling condition which is a constraint on the values of the clocks. A transition can be taken only if the current clock values satisfy its enabling condition.

The following definitions are mainly identical to those given in [AD94].

Definition 1 (Clock constraints and clock guard). *A* clock constraint *over a set C of clocks is a boolean expression of the form x* **oprel** *z where $x \in C$,* **oprel** *is a classical relational operator ($<, \leq, =, \geq, >$), and z is an integer constant. A* clock guard *over C is a conjunction of clock constraints over C.*

Definition 2 (Timed Input Output Automata). *A timed input output automaton [AD94] A is defined as a tuple* $(\Sigma_A, L_A, l_A^0, C_A, E_A)$*, where :*

- Σ_A *is a finite alphabet, split in two sets :* $\mathcal{I}$ *(input actions) beginning with a "?", and* $\mathcal{O}$ *(output actions) beginning with a "!".*
- L_A *is a finite set of states,*
- $l_A^0 \in S$ *is the initial state,*
- C_A *is a finite set of clocks,*
- $E_A \subseteq L_A \times L_A \times \Sigma_A \times 2^{C_A} \times \Phi(C_A)$ *is the set of transitions.*

An edge (l, l', a, λ, G) *represents a transition from state* l *to state* l' *on input or output symbol* a*. The subset* $\lambda \subseteq C_A$ *allows the clocks to be reset with this transition, and* G *is a clock guard over* C_A*.* $\Phi(C_A)$ *is the set clock guards over* C_A

4 Robustness Testing

4.1 Robustness Testing Issues

In this section, we will try to summarize aspects and new problems inherent to robustness testing. We will compare robustness testing to a well-known testing technique, conformance testing which was the main subject of many surveys.

Conformance testing allows us to check if an implementation I satisfies its specification S. Of course, we should define a satisfaction relation. We can cite for example the trace inclusion or the trace equivalence. The test sequence generation step suffers in general from the size explosion since in real systems, the number of test sequences are very large. For timed systems, this problem is more complex and generates sometimes infinite sequences. To solve the problem, it is sometimes necessary to select only a subset of sequences under some assumptions like uniformity or behavior reduction. But we should to think about the fault coverage which decreases if the testing sequence number does so. We can sometimes focus only on some test purposes to lighten the testing step. The testing step ends with a verdict: PASS, FAIL, or INCONCLUSIVE. The figure 1 recalls the main aspects of conformance testing.

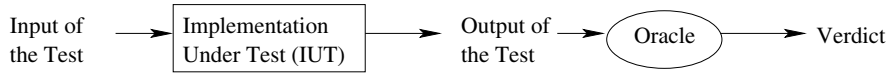


Fig. 1. Conformance testing principle

To tackle robustness testing, we need to answer the following questions: what are the (possible) differences between conformance testing and robustness testing? How to tackle the input domain of the test? How to interpret the output

domain? How to model the system? And finally what testing architecture will we use?

We define the robustness notion as: a system is considered as robust if it is able to operate correctly in the presence of invalid inputs or stressful environment (IEEE definition). We will measure the ability of the system to have a "correct" or "acceptable" behavior in the presence of hazards (random errors).

So, we can make out an idea to test the robustness of a system: we can use the sequences obtained for the conformance testing as the basis of our method. We will apply them to the system, and in case of error (internal hazard, not provoked), we measure the ability of the system to assume this error. Then, it is possible to insert some well chosen hazards in the test sequences, and apply the previous step again (we should simulate the external hazards). In this case, we should consider again the oracle and modify the different verdicts. Actually, the verdicts considered in robustness testing are much more smart, notions of success or failure are not enough. We can add for example a robustness measure, or we can model robustness properties in some logics. Formally, we say that we have to extend the test input domain of hazards. Then, the input domain extension implies necessarily an output domain extension (moreover, even if the input domain is not extended, it is possible to extend the output domain). We need to interpret this issue with a robustness perspective, it is an observability aspect. In this work, the notion of hazard is very important. That is why we will see it later in detail.

In parallel to these problems, it is necessary to find a model of the system. Many choices are possible:

1. we can decide that the system model does not consider the hazards. In this case, the difficulty will be to search for significant hazards. Actually, it is possible that a large part of guided injected faults will be not activated in the system, or their consequences are not observable. In this case, we should look for heuristic methods to select the faults to inject in order to obtain the more pertinent and critical scenari. It would be necessary to identify the fault sort(hardware, software or human), and eventually to define a notion of fault power (catastrophic, ...). We can think also about testing the system in the presence of an abnormal amount of work.
2. we can, on the contrary, choose to consider the hazards in the model, which brings us closer to the conformance testing process. We separate in this model the aspects of "nominal" functioning from the "degraded" functioning (figure 2). In this case, it is possible to develop an approach taking as inputs: a specification S, a fault model M, and a robustness property P and taking as output test sequences.
3. an intermediate model of both previous approachs.

Besides, whatever the choice of approach described bellow, we have to decide what kind of model to use. We can choose the Labelled Transitions Systems (LTS), in an extended version for example, or perhaps a kind of LOTOS representation (RT-LOTOS for example), or the suggestion of our own model.

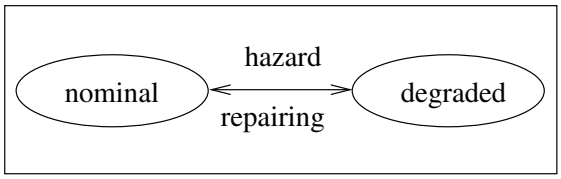


Fig. 2. Nominal and degraded modes

To summarize, it would be helpful if this model could consider actions, timing constraints, data, and finally the possibility for the different components to communicate with each other.

The purpose of robustness testing is not to detect possible faults, but rather to see how the system reacts to hazards, and consequently to some situations. We consider that the system is described by two specifications written in the timed automata formalism (TIOA): a *nominal* specification $S = (\Sigma, \mathcal{S}, s_0, \mathcal{C}, \mathcal{T})$ which describes the behavior of the system, and a degraded specification S_{degr} $=(\Sigma_{degr}, \mathcal{S}_{degr}, s_{0degr}, \mathcal{C}_{degr}, \mathcal{T}_{degr})$ which describes the system in a degraded mode,i.e., it describes the vital functionalities and the minimum required behavior. For example for a robot, we could require that it has to send its position at least every 10 seconds (in the degraded specification) whereas it sends its position every one second in the nominal specification. The idea is to generate test sequences from the nominal specification based on any classic conformance testing method. Then, we insert some hazards to these sequences (for the external hazards). The tester has to send stimuli at the right moment respecting the timing constraints, and has to check the response validity. As soon as a fault is detected by the tester, we only record the system responses, and we continue to send the expected inputs (of the nominal behavior), ie, to execute the test sequence without checking the responses. At the end of the sequence, if we have some unexpected responses, we look if the obtained execution trace (see the definitions below) is accepted by the degraded specification S_{degr}. If no fault has been detected by the tester, then the system is considered as robust enough regarding the considered hazards and the desired robustness level. Furthermore, to measure the system robustness in case of internal hazard, we execute the testing process with the method described before without any insertion of hazards in sequences. In fact, each time we find an error during the execution step, we record the event and we continue testing and event recording. In fact, we watch the ability of the system to react to one of its own error (which is different from the conformance testing).

In the following, we present some definitions needed to explain our test robustness algorithm.

We consider that we have at our disposal a set of test sequences produced by any derivation algorithm on Input Output timed automata.

Definition 3 (Set of test sequences). *Let $S = (\Sigma, \mathcal{S}, s_0, \mathcal{C}, \mathcal{T})$ a timed automaton of the nominal specification, we denote TSS (Test Sequences Set)=$\{seq_1$, ..., $seq_n\}$, where $\forall i \in [1..n], seq_i = \{t_{i_1}, ..., t_{i_m}\}$ and $t_{i_j} \in \mathcal{T}, m = card(seq_i)$.*

We define the concept of an event which is the execution of an action at a specific time valuation.

Definition 4 (Event). *An event (a, t) is the execution of an action $a \in \Sigma$ at the timing t corresponding to a valuation of all the clocks. In our case, each action is considered observable (no ϵ in our alphabet).*

A timed trace is a sequence of actions (with their execution valuations) starting at the initial state.

Definition 5 (Timed trace). *A timed trace is a sequence $\sigma = (a_1, t_1)(a_2, t_2)...(a_n, t_n)$ of observable events going from the initial state. From this initial state, σ allows us to know that the action a_1 is observed at the time valuation t_1, a_2 at t_2, etc ... $\forall j, t_j$ are time valuations.*

Here, we define a relation between a sequence of actions and its possible clock valuations.

Definition 6 (Execute). *Let σ a timed trace of the TIOA A, going from the initial state, and ending at a state considered as final. Let $L = \bigcup_{\infty}^{i=0} \sigma_i$ the set of all timed traces. As each path p in the automaton can lead to a different observation, we define the relation $execute(p, A, \sigma)$ which unifies a path p of an automaton A and its timed trace σ ($\sigma \in L$). A path contains only action labels 9without timing constraints).*

In order to generalize this notion of path, we define the notion of route.

Definition 7 (Route).
Every possible observation of a path p for an automaton A is in:
$Route(p, A) = \{\sigma \in L, execute(p, A, \sigma)\}$

$Route(p, A)$ is the union of all the timed traces obtained by going through the path p, and instead to handle an infinity of consecutive instants for a precise event, we gather them in an interval. This union allows us, in the following, to use an interval instead of a set of consecutive instants.

Definition 8 (Conformance relation). *Let $exec = (a_1, t_1)...(a_n, t_n)$ a timed trace and spec a TIOA. We say that exec is conform to spec if:*

For every path p in A, by noting $Route(p, A) = (a'_1, T'_1)...(a'_m, T'_m)$, with T_i a time interval, we have: $\forall j \in [1..(n-1)], \exists k$ so that $a'_j = a_k, t_k \in T'_j$ and $\exists k' > k$ so that $a'_{j+1} = a_{k'}, t_{k'} \in T'_{j+1}$. In other words, each action of a timed trace of spec has to be in exec, respecting the timing constraints, and of course in the same order than the actions of spec.

4.2 Robustness Testing Algorithm

We will see now the general algorithm of our technique. Then, we will experiment all the test sequences, and check their response validity. In case of error, we continue to experiment the rest of the test sequence. Before starting any new test sequence, the tester checks if the obtained trace is conform to the degraded specification S_{degr}, by using the conformance relation defined below. Then we insert hazards in the testing sequences of TSS. At the end, if one sequence gives a non conform trace, then we consider that the system is not robust enough in comparison with the desired robustness.

We consider $TSS = \{seq_0, ..., seq_n\}$ and $\forall i \in [1..n], seq_i = \{t_{i_1}, ..., t_{i_m}\}$ with $t_{i_j} \in \mathcal{T}$. Then, $n = card(TSS)$ and $m = card(seq_i)$. This algorithm is then presented in Algorithm 1.

Notice that in order to check the validity of an output transition, the tester only checks the sent action is correct, and its time interval is also correct. Moreover, when an error is detected, we check that the execution trace is conform to S_{degr}.

5 An Example

Suppose we have a robot in an hostile environment. The simplified whole specification of this robot describes that it sends its position after a position request (?positionReq), or the temperature after a temperature request (?temperatureReq). But the system must send its position and the temperature with regularity (the limit is 120s for temperature and 60s for position). The robot has a moving mode: it is able to turn or to go forward during a certain period, interrupted by a stop signal (?stopTurn or ?stopForward). The figure 3 shows this specification, which is the specification in a "normal" mode.

An example of a degrated specification of this system could be the obligation for it to send its position at least every 300s, and to send the temperature at least every 600s. Then, the degrated specification could be described in figure 4. These functionalities are necessary to decide that the system is robust.

An example of test sequences generated with our conformance testing method, using the normal specification is :

- S1: (?temperatureReq,$x < 120$),(!temperature,$x := 0$);
- S2: Non Controllable;
- S3: Non Controllable;
- S4: (?endMove),(?positionReq,$y < 60$),(!position,$y := 0$);
- S5: (?stopTurn),(?endMove),(?positionReq,$y < 60$),(!position,$y := 0$);
- S6: (?stopForward),(?endMove),(?positionReq,$y < 60$),(!position,$y := 0$);

Notice that the complete test sequences are much more longer, since we have to reach every controllable state, to test it with its particular sequence, and then to test each transition and to check for each transition if the state reached is the expected one.

```
Data    : TSS , S_degr
Result  : trace , robusteEnough
robusteEnough ⟵ true;
while there are hazards left to insert in TSS and robustEnough do
    i ⟵ 1;
    while i ≤ n and robustEnough do
        j ⟵ 1;
        trace ⟵ NULL;
        errorFound ⟵ faux;
        while j ≤ m do
            if t_ij ∈ ℑ then
                apply t_ij to the system;
                if errorFound then
                    add t_ij to trace;
                end
            else
                if t_ij ∈ ℴ then
                    if not(errorFound) then
                        verify that t_ij is correct ;
                        if incorrect(t_ij) then
                            errorFound ⟵ true;
                            add t_ij to trace;
                        end
                    else
                        add t_ij to trace;
                    end
                end
            end
            j ⟵ j + 1;
        end
        if errorFound then
            if inclusionTrace(S_degr, trace) then
                System robust enough in comparison to S_degr for seq_i;
            else
                System not robust enough in comparison to S_degr for seq_i;
                robustEnough ⟵ faux;
            end
        end
        System robust enough for seq_i;
        i ⟵ i + 1;
    end
    if there are hazards left to insert to TSS then
        insert new hazards in TSS;
    end
end
```

Algorithm 1: Robustness testing algorithm

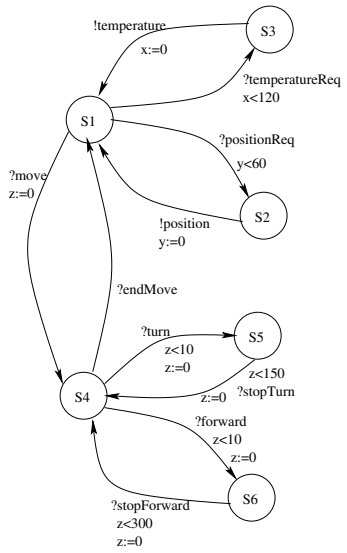


Fig. 3. Normal specification

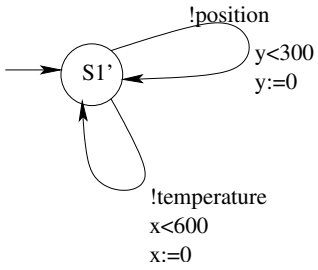


Fig. 4. Degrated specification

6 Conclusion

In this paper we introduced a simple approach for robustness testing. We have presented a complete methodology from specification to testing. We have used RT-LOTOS as a high-level formalism, and timed automata as low-level formalism. We have chosen these formalism since they deal with time constraints with accuracy.

We have suggested an approach to test robustness of critical systems. We first consider that we have two system specifications: one which contains all functionalities, denoted the nominal specification and a second one which contains the most important functionalities denoted as the degraded specification.

The robustness testing technique is based on the generation of test sequences from the whole specification. Then after the execution of these test sequences on the implementation, we check if the implementation responses are valid on the reduced specification. In fact we test if the crucial behavior is performed by the implementation.

The main limitation of this methodology is that we cannot guarantee to have an entire fault coverage since it is difficult to ensure a large fault coverage for timed systems (they are infinite systems).

We have undertaken the implementation of our methodology in an integrated tool, then we will be able to experiment it on some real cases as a robot functioning or a multimedia protocol.

References

[ACH94] R. Alur, C. Courcoubetis, and T.A. Henzinger. The observational power of clocks. In B. Jonsson and J. Parrow, editors, *Proceedings CONCUR 94,* Uppsala, Sweden, volume 836 of *Lecture Notes in Computer Science*, pages 162–177. Springer-Verlag, 1994.

[AD94] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[BB89] T. Bolognesi and E. Brinksma. Introduction to the ISO specification language LOTOS. In P.H.L. van Eijk, C.A. Vissers, and M. Diaz, editors, *The formal description technique LOTOS*, pages 23–73. Elsevier Science Publishers, 1989.

[BDS95] J. Bryans, J. Davies, and S. Schneider. Real-time CSP and ET-LOTOS. Technical report, Dept. of Computer Science, University of Reading, 1995.

[BL92] T. Bolognesi and F. Lucidi. Timed process algebras with urgent interactions and a unique powerful binary operator. In J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors, *Proceedings REX Workshop on Real-Time: Theory in Practice,* Mook, The Netherlands, June 1991, volume 600 of *Lecture Notes in Computer Science*, pages 124–148. Springer-Verlag, 1992.

[BLL+98] J. Bengtsson, K.G. Larsen, F. Larsen, P. Petterson, W. Yi, and C. Weise. New Generation of UPPAAL. In *Proceedings of the International Workshop on SOftware Tools and Technology Transfer (Aalborg, Denmark, July 12-13)*, July 1998.

[CL97] Duncan Clarke and Insup Lee. Automatic generation of tests for timing constraints from requirements. In *Proceedings of the Third International Workshop on Object-Oriented Real-Time Dependable Systems,* Newport Beach, California, February 1997.

[CO95] Jean-Pierre Courtiat and Roberto C. De Oliveira. On rt-lotos and its application to the formal design of multimedia protocols. *Annals of Telecommunications*, 50(11-12):888–906, nov 1995.

[COG98] R. Cardel-Oliver and T. Glover. A practical and complete algorithm for testing real-time systems. In *Proc. of the 5th. Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 1486 of Lecture Notes in Computer Science, pages 251–261. SpringerVerlag, 1998.

[DNH84] R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[DOTY95] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool Kronos. In R. Alur, T.A. Henzinger, and E.D. Sontag, editors, *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*, pages –. Springer-Verlag, 1995.

[DOY94] C. Daws, A. Olivero, and S. Yovine. Verifying ET-LOTOS programs with KRONOS. In D. Hogrefe and S. Leue, editors, *Proceedings of the 7^{th} International Conference on Formal Description Techniques, FORTE'94*, pages 207–222. North-Holland, 1994.

[DY95] C. Daws and S. Yovine. Two examples of verification of multirate timed automata with KRONOS. In *Proceedings of the 1995 IEEE Real-Time Systems Symposium, RTSS'95,* Pisa, Italy. IEEE Computer Society Press, 1995.

[ENDKE98] A. En-Nouaary, R. Dssouli, F. Khendek, and A. Elqortobi. Timed test cases generation based on state characterization technique. In *19th IEEE Real Time Systems Symposium (RTSS'98)* Madrid, Spain, 1998.

[FBK+91] S. Fujiwara, G. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi. Test selection based on finite-state models. *IEEE Transactions on Software Engineering*, 17(6):591–603, June 1991.

[HNTC01] Teruo Hogashino, Akio Nakata, Kenichi Taniguchi, and Ana R. Cavalli. Generating Test Cases for a Timed I/O Automaton Model. October 2001.

[ISO87] ISO. *Information processing systems – open systems interconnection – LOTOS – a formal description technique based on the temporal ordering of observational behaviour* ISO/TC97/SC21/N DIS8807, 1987.

[ISO97] ISO. Working draft on enhancements to lotos. Technical Report ISO/IEC JTC1/SC21, International Organization for Standardization — WG7, Genève, January 1997.

[Kar92] G. Karjoth. Generating Transition Graphs from LOTOS Specifications. In Michel Diaz and Rolland Groz, editors, *Proceedings of the 5th International Conference on Formal Description Techniques FORTE'92 (Perros-Guirec, France)*, Amsterdam, November 1992. North-Holland.

[Kon95] O. Koné. Designing test for time dependant systems. In *Proceedings of the 13^{th} IFIP International Conference on Computer Communication* Séoul, South Korea, 1995.

[LC97] Patrice Laurencot and Richard Castanet. Integration of Time in Canonical Testers for Real-Time Systems. In *International Workshop on Object-Oriented Real-Time Dependable Systems, California*. IEEE Computer Society Press, 1997.

[Led92] G. Leduc. An upward compatible timed extension to LOTOS. In K. Parker and G. Rose, editors, *Proceedings of the 4^{th} International Conference on Formal Description Techniques, FORTE'91*. North-Holland, 1992.

[LL93] G. Leduc and L. Léonard. A timed LOTOS supporting a dense time domain and including new timed operators. In *Proceedings of the 5^{th} International Conference on Formal Description Techniques, FORTE'92*. North-Holland, 1993.

[NS01] Brian Nielsen and Arne Skou. Automated Test Generation from Timed Automata. In T. Margaria and W. Yi, editors, *Proceedings of the Workshop on Tools and Algorithms for the Construction and Analysis of Systems,* Genova, Italy, volume 2031 of *Lecture Notes in Computer Science*, pages 343–357. Springer-Verlag, April 2001.

[PF99] E. Petitjean and H. Fouchal. From Timed Automata to Testable Untimeed Automata. In *24th IFAC/IFIP International Workshop on Real-Time Programming, Schloss Dagstuhl, Germany*, 1999.

[RNHW98] P. Raymond, X. Nicollin, N. Halbwatchs, and D. Waber. Automatic testing of reactive systems, madrid, spain. In *Proceedings of the 1998 IEEE Real-Time Systems Symposium, RTSS'98*, pages 200–209. IEEE Computer Society Press, December 1998.

[SVD01] J. Springintveld, F.W. Vaandrager, and P. R. D'Argenio. Timed Testing Automata. *Theoretical Computer Science*, 254(254):225–257, 2001.

Semantic Geocast Using a Self-organizing Infrastructure

Jrg R oth

University of Hagen
Department for Computer Science
58084 Hagen, Germany
Joerg.Roth@Fernuni-hagen.de

Abstract. Geocast mechanisms allow a sender to transmit network packets to receivers residing at a certain geographical region. Geocast forms the basis for a number of location-based services, such as announcement services, advertisement services or friend-finders. In this paper, we introduce the notion of *semantic geocast*, where a target area is specified by its *meaning*. A sender can broadcast messages to, e.g., a city centre or a specific building, without precisely knowing the physical co-ordinates. We implemented semantic geocast on top of our self-organizing *Location Server Infrastructure* (*LSI*), which reflects a location domain model especially designed to cover the needs of mobile users. As our infrastructure is self-organizing, it is flexible and easy to extend. We consider scalability and stability issues. LSI and its geocast mechanism is fully implemented and tested. Evaluations show the effectiveness of our approach.

1 Introduction

Location-based services will become increasingly popular in the future. Applications that take into account a mobile user S current location play a major role in the area of ubiquitous, pervasive and handheld computing. Many people expect a high potential of location-based services such as city guides or navigation systems for m-commerce scenarios. To support developers of location-based services we created the platform *LSI* (*Location Server Infrastructure*). LSI hides the specific mechanism to determine the mobile user S current location and provides both physical co-ordinates as well as semantic information about the current location. With LSI, mobile users can switch between satellite navigation systems such as GPS, positioning systems based on cell-phone infrastructures, or indoor positioning systems without affecting the location based-service. A developer can concentrate on the actual service function and has not to deal with positioning sensors or capturing protocols.

One powerful tool to develop location-based services is *geocast*. Like multicast, geocast transmits a network packet to a number of receivers, but in contrast to multicast, the target is a certain geographical region. Geocast is an ideal basic function for a number of location-based applications. With geocast, we can send warning announcements to a region with bad weather conditions, supermarkets can send advertisement messages to all clients inside a building, and friend-finder applications can look for friends in the nearer area of a mobile user.

T. Bh me, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 216-228, 2003.

In this paper, we introduce the notion of *semantic geocast*: the target address is not defined by a *physical* area (specified by, e.g., a polygon), but by a *semantic* location such as "University of Hagen". Using semantic geocast, users and applications do not have to deal with raw physical co-ordinates, but can use simple location names to describe the target.

2 Related Work

The notion of geocast was introduced by Imielinski and Navas ([5], [6], and [11]). As a basic idea, geocast extends traditional networks by services to use geographical target addresses. In [11], Navas and Imielinski suggested a hierarchical network of *GeoRouters* that reflects a structure of a wireless cellular (e.g., cell-phone) network. As there is no notion of semantic locations, we can only use physically defined targets. In [5], semantic locations are partly supported, as some semantic locations (e.g., countries and cities) are represented by individual multicast addresses using multicast IP group addresses. This approach, however, was not scalable, because the number of potential multicast IP groups is far to small to cover a reasonable area such as an entire country. In addition, the multicast IP infrastructure is not prepared for a huge number of multicast members, moreover not generally available for mobile users.

Many location-based applications have been developed in the last years, which use semantic locations. Cyberguide [1], Guide [2] and the PinPoint Tourist Guide [16] offer information to tourists, taking into account their current (semantic) location. Context-aware messaging tools trigger actions according to a specific semantic location [18]. ComMotion [10] and CybreMinder [3] link locations to events, e.g. give an alarm if time is "9:00" and location is "my office". These tourist guides and messaging services use their own, hard-coded mechanisms to express semantic locations. They would heavily benefit from a general infrastructure to use semantic geocast.

Several research platforms provide a basis to develop location-based services. Cooltown [8] is a collection of location-aware applications, tools and development environments. Nexus [4] introduces so-called augmented areas to formalize location information. Augmented areas represent spatially limited areas, which may contain real as well as virtual objects. OpenLS [12] is an upcoming project and provides a high-level framework to build location-based services. All these systems could heavily benefit from a framework supporting semantic locations as well as semantic geocast services.

3 Semantic Locations and the Location Server Infrastructure

The notion of semantic locations is not new (e.g., [9], [18]), but descriptions often tend to be very abstract. Pradhan distinguishes three types of locations [14]: *physical* locations such as GPS coordinates, *geographical* locations such as "City of Hagen" and *semantic* locations such as "Jrg s office at th e university". In this paper, we do not distinguish geographical and semantic locations, but view any location other than physical as a semantic location. In the following, we first introduce a formal model

for semantic locations. We then present our infrastructure LSI, which reflects this formal model.

3.1 Semantic Locations

Semantic locations are appropriate for a number of applications, sometimes in combination with physical locations. Semantic locations have some important advantages:

- Semantic locations have a meaning to the user; in contrast, physical locations usually have no meaning at all to most peoples.
- Semantic locations can easily be used as a search key for traditional databases, tables or lists. Looking up physical presentations, we need spatial databases with the ability to deal with geometric objects such as polygons.

In this section, we want to describe the concept of semantic locations more precisely. We especially want to relate semantic locations to physical locations. Let P denote the set of all physical locations. We call each coherent area $S \subseteq P$ a *semantic location* of P. We further call each set $C \subseteq 2^P$ of semantic locations, a *semantic coordinate system* of P. (2^P denotes the power set of P.) Note that we do not assume two semantic locations to be generally disjoint. A reasonable semantic coordinate system C contains semantic locations S with certain meanings, e.g.,

- locations with a political meaning: countries, states, districts, cities;
- geographical locations: continents, mountains, rivers, lakes, forests;
- mobile locations: trains, planes, cars;
- temporary locations: construction zones, fairs;
- other locations: campus, malls, city centres.

We further introduce a *name* for a semantic location. Let N be the set of all possible names. We define a function $NAME$: $C \rightarrow N$, which maps a semantic location to a string. We require names to be unique, i.e. $NAME(c_1) \neq NAME(c_2)$ for $c_1 \neq c_2$. We call a semantic location with its corresponding name a *domain*. For a domain d, $d.name$ denotes the domain name, $d.c$ the semantic location.

In principle, a semantic coordinate system C could be an arbitrary subset of 2^P that contains coherent areas. Looking at real-world scenarios, however, we usually find hierarchical structures (fig. 1), e.g., a room is inside a building, a building is in a city, a city is in a country etc.

We divide C into so-called *hierarchies*. A hierarchy contains domains with a similar meaning, e.g., domains of german cities or domains of geographical items. Each hierarchy has a *root domain* and a number of *subdomains*; each of it can in turn be divided into subdomains. We call a top node of a subhierarchy a *master* of the corresponding subdomains. We denote $m \rhd s$ for master m of subdomain s. Further $\succ$ denotes the reflexive and transitive closure of $\rhd$, i.e. $d_1 \succ d_2$ if either $d_1 = d_2$ or d_1 is a top node of a subtree which contains d_2.

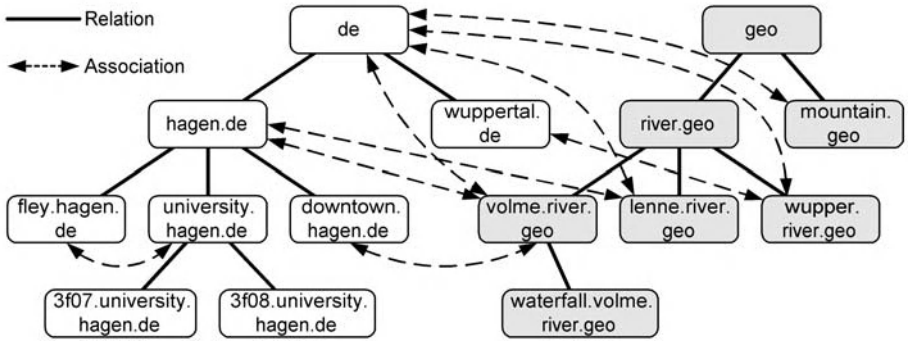


Fig. 1. A sample semantic coordination system

Fig. 1 shows two hierarchies, a `de` hierarchy (the area of Germany, white boxes) and a `geo` hierarchy (geographical entities such as rivers and mountains, grey boxes).

We call a link between a subdomain and its master *relation*. Relations carry information about containment of one domain according to another. Hierarchies are built according to three rules:

- The area of a subdomain has to be completely inside the area of its master, i.e. if $d_1 \triangleright d_2$ then $d_2.c \subset d_1.c$.
- The name of a subdomain d_2 extends the name of its master d_1 according to the rule $d_2.name$=<extension> + □□+ $d_1.name$, where <extension> can be an arbitrary string containing only letters and digits. With the help of this rule, we can effectively check if $d_1 \succ d_2$ or $d_1 \triangleright d_2$ with the help of the domain names.
- Root domain names of two hierarchies must be different.

In addition to relations, a domain can be *associated* to other domains. Two domains d_1, d_2 are associated, if they share a physical area (i.e. d_1.c $\cap$ d_2.c $\neq \{\}$) and neither $d_1 \succ d_2$ nor $d_2 \succ d_1$. Associated domains can be in different hierarchies or in the same hierarchy. The domain `downtown.hagen.de` is associated to `volme.river.geo`, because Volme is a river which flows through the downtown of Hagen. Associations carry important information for location-based services. E.g. with the help of associations, we can discover *all* semantic locations of a specific physical location.

The number of associations can be very high for high-level domains. We reduce the amount of associations with a compression mechanism [17], which deletes associations without loosing the corresponding information. In fig. 1, e.g. `de` is not associated to `geo`, since lower-level associations carry all necessary information.

3.2 The Technical Infrastructure

Our technical infrastructure LSI reflects the domain model described above. A distributed system of so-called *location servers* (*LS*) stores location information and provides services for mobile clients and the corresponding location based services (fig 2). The infrastructure consists of three *segments*:

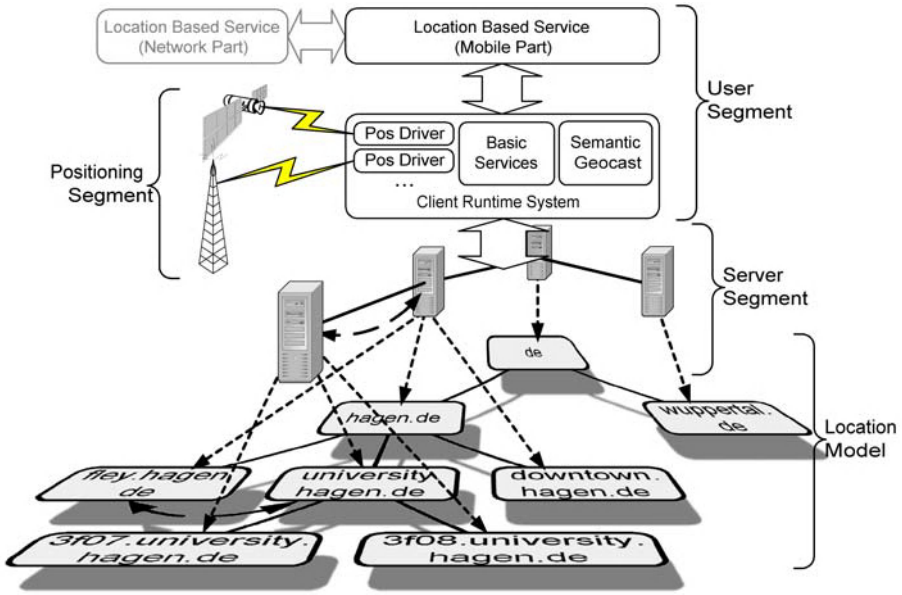


Fig. 2. The system architecture

The *positioning segment* contains the positioning systems, e.g., indoor positioning systems, satellite navigation systems or systems based on cell phone infrastructures. The *user segment* contains the mobile node with the LSI runtime system and the mobile part of the location based service. Note that our infrastructure does not cover the network part of a location-based service. It depends on the mobile part to establish a connection to a specific server and to use the service. The runtime system accesses the positioning systems through *position drivers*. With the help of drivers, we can change the positioning systems, even at runtime, without affecting the rest of the system. The client runtime system also contains the following components:

- *Basic services* provide a homogenous view on locations for location-based services. These services map raw location information from the positing drivers to both physical as well as global unique semantic locations. These services are described in [17].
- *Semantic geocast*: With this component, a location-based service can send and receive geocast messages.

The *server segment* contains the location servers that store the domain data. In principle, we could use one huge database and store hierarchies with the corresponding domains on a single server. One database for a huge number of potential clients, however, would be a bottleneck. In addition, information about local domains is usually available only locally and difficult to administrate in a central database. As a solution, we use a distributed system of location server each storing a number of domains. Each location server is responsible for a specific domain and all subdomains, for which no other location server exists. In our example, the location server for `hagen.de` covers `fley.hagen.de` and `downtown.hagen.de`, but not `university.hagen.de`, as this domain has its own location server.

The entire system is self-organizing. The location servers establish the relation and association links among each other automatically. Building these links is done by a set of lookup and discovery protocols not described in this paper (see [17] for details). In order to decouple the infrastructure from communication aspects, we use a communication middleware, especially designed for mobile scenarios [15]. When a mobile node moves to another location, it automatically looks up an appropriate location server (called the *local location server*, *LLS*). The LLS is the representative of the entire infrastructure for a mobile node. Any service usage is directed to the LLS. As mobile users are distributed among different location servers, this infrastructure is highly scalable. Especially, our system does not overload top-level servers.

4 Semantic Geocast Using LSI

The logical structure of relations and associations forms an ideal platform for a geocast mechanism as domain information can be distributed among this logical network. A geocast request r from a mobile node contains a target domain $r.domain$ and a message $r.message$. The goal is to transfer $r.message$ to all mobile nodes residing at positions $p \in r.domain.c$.

In the following, we make a simplification: we represent every domain by its own location server. We assume that a communication between two domains always needs a network transaction. In reality, the performance of our system is far better, as communication often can be done inside a location server. Thus, our performance evaluations in a later section describe a worst-case scenario.

4.1 The Semantic Geocast Mechanism

The basic idea of our semantic geocast mechanism is as follows:

- *Registration*: Each mobile node registers itself at all location servers, which cover the occupied semantic locations. The location servers accept geocast requests and in turn deliver other geocast messages to the mobile nodes.
- *Address Propagation*: Each location server builds a list of network addresses of other location servers. The lists are periodically updated, thus, they notice, when servers start up or are shut down.
- *Message Passing*: When a location server receives a geocast request, it looks up an appropriate location server in its address list for delivery and redirects the request. Often, this server is not the final destination, thus additional transfers may be required.
- *Delivery*: Finally, a target location server receives the message and distributes it to the registered mobile nodes. As more than one location server may cover the target domain, additional transfers to other servers may be required.

Fig. 3 illustrates the basic mechanism. Note that in this figure, we equate domains to their location servers.

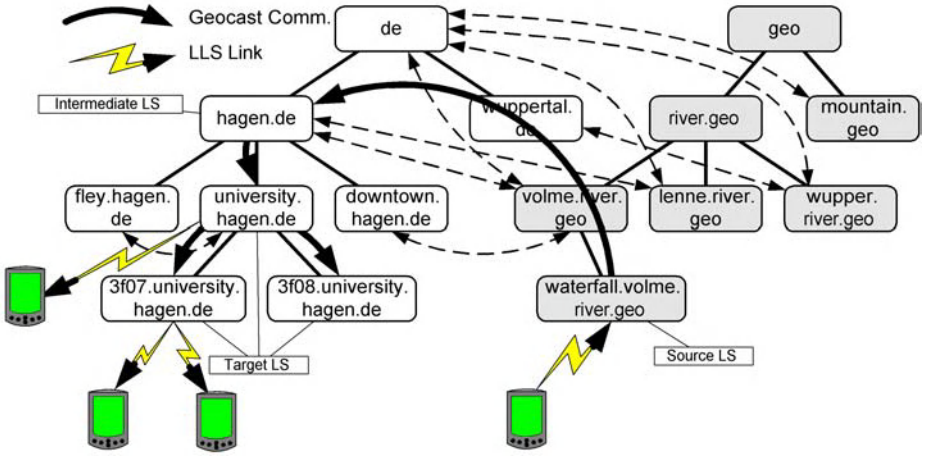


Fig. 3. The geocast mechanism

In this scenario, a mobile node residing at `waterfall.volme.river.geo` sends a geocast message to all mobile nodes at `university.hagen.de`. We distinguish the following servers, which process a geocast request:

- The *source LS* accepts the geocast request from a mobile node (`waterfall.volme.river.geo` in fig. 3).
- *Intermediate LSs* relay geocast requests to the target domains (`hagen.de` in fig. 3).
- *Target LSs* are responsible for the target domain and send geocast messages to the mobile nodes (`university.hagen.de`, `3.f07.university.hagen.de` and `3f08.university.hagen.de` in fig. 3).

Note that the communication between LSs can use "short cuts" and are not restricted to associations and relations. We assume that all servers are connected via a global network (usually the worldwide Internet). Once a server has another server in its address list, they can communicate directly.

The geocast mechanism mainly contains two parts: a proactive part to collect addresses and a reactive part to route geocast requests.

4.2 Address Propagation

Each server proactively collects addresses of other locations servers. This is permanently done in the background, thus new servers are registered after a delay. As the list of all location servers in the system can be very large (e.g., many thousands entries), we allow each server to build a list with a specific length limitation. This reduces the amount of memory required by an LS, but also reduces the traffic to maintain the address lists. Our mechanism ensures stability, even if a target LS is not listed by the source LS. Locations servers collect addresses according to two mechanisms: *slow propagation* and *fast propagation*.

The *slow propagation* is built according to the propagation mechanism integrated in the DSDV ad-hoc routing algorithm [13]. When a server starts up, it sends an update message with its own address to its "neighbours", i.e. its master, all subdomains and all associated servers. The message contains a sequence number, which a server has to increase at every new start-up.

Whenever a server receives an address update, it first looks in its own table whether it already has received an update with this sequence number. If yes, the message is simply ignored; if not, it stores this new information and forwards the update to all neighbours apart from the originator. The sequence number avoids eternally circulating updates. Each server periodically (e.g. every day) increases its own address sequence number and distributes the address. Each address entry has a certain lifetime, specified by the originating server. Thus, disconnected servers are removed from the lists after the lifetime expired.

To reduce the overall traffic, each server collects update messages for a specific time (e.g. 10 minutes) and then exchanges them in a bundle. As a starting server does immediately flood updates through the network, denial of service attacks are more difficult. We call this mechanism the *slow propagation*, since it takes a considerable long time (e.g., some hours) for every location server to list an address of a new server.

To propagate new addresses much faster, we use an additional mechanism, the *fast propagation*. A new server first starts with the slow propagation. Its own address list initially is empty and thus it receives new addresses from its neighbours. Whenever it receives a root domain server, it uses the fast propagation mechanism: it *once* sends an address update to this root server. As a result, this root server distributes the new address in its own hierarchy, passing this information down the hierarchy tree. If neighbours of a new server already know the root domains of all hierarchies, addresses are distributed very fast among the entire infrastructure.

To investigate the effectiveness of our mechanisms, we ran a number of simulations. Since LSI is fully implemented and operable, we can use the real infrastructure for evaluation purposes. We developed an additional simulation tool to randomly generate a huge number of domains. The tool first creates a root domain for every hierarchy and then additional levels of domains by adding up to 10 subdomains for each domain. The process runs until we reach the required number of domains. Finally, the tool randomly adds associations between the hierarchies. We run a number of simulations with the same parameters to compensate outliers. We first use the random hierarchies to compare the slow and fast propagation (fig. 4). In the following, h denotes the number of hierarchies and n the total number of domains in all hierarchies.

As real network delays heavily depend on the actual network structure and load, we only measure the hops in our simulations. Fig. 4 shows the maximum number of hops to inform a server about a newly started server. We simulate scenarios with 2 and 16 hierarchies. If all nodes are distributed among a higher number of hierarchies, the propagation works more effectively, because associations connect domains more tightly. For any number of hierarchies, the fast propagation needs a significant lower number of hops to inform all domains, thus we always use fast propagation, whenever a new node collects a root server address.

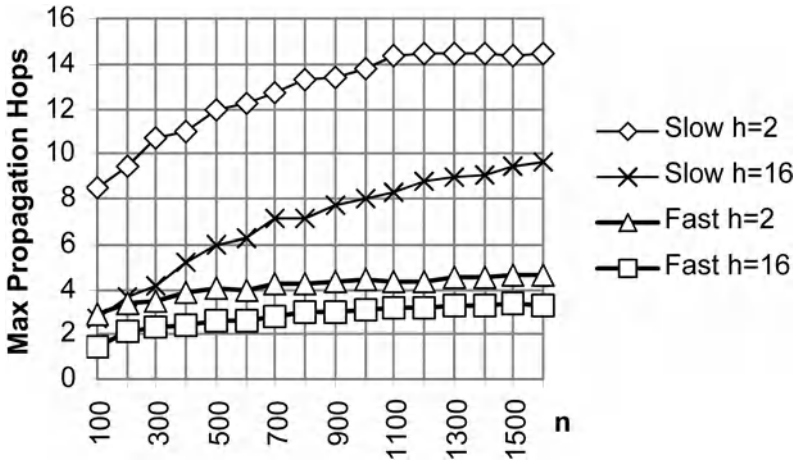


Fig. 4. Comparison of Slow and Fast Propagation

4.3 Message Passing

Once a source LS receives a geocast request from a mobile node, it looks up an appropriate location server. This location server can either be a target LS or an intermediate LS. The latter case occurs, if a target LS is not listed, either because of a limited list space or the propagation has not yet completed.

The following pseudo code outlines the thread integrated in every LS to handle geocast requests. Here, x denotes the local LS.

```
while (true) // The handle thread loops endlessly {
  wait for geocast request r;
  if r.domain≻x.domain { // I'm a target LLS ❶
    send r.message to all registered mobile nodes;
    send r to all subdomains;
  }
  else { // I'm an intermediate or a source LS ❷
    look up servers s in the local address list where
       s.domain≻r.domain – if more than one server is found,
       choose the lowest in the hierarchy;
    if such a server s is found
      send r to s;
    else { // Try routing via relations and associations ❸
      look up subdomain server y of x with y.domain≻r.domain;
      if such a server y is found  // there can only be one
        send r to y;
      else if x has associations into the target hierarchy {
        choose an appropriate associated server z
            (see selection above);
        send r to z;
      }
      else if x has a master m  // Try the master
        send r to m;
```

```
        else
          return an error to the originating node;
      }
  }
```

Usually a source LS handles a request (at ❷), which is directly passed to a target LS (at ❶). If a target is not listed, a request is send to an intermediate LS (at ❷), which may relay it in turn to another intermediate LS. The message passing mechanisms ensures that requests finally arrive at a target LS. A target LS delivers the messages to mobile nodes. In addition, it relays the request to all subdomains. Note that as subdomains are completely inside a master domain, every subdomain has to process the geocast request as well. Section ❸ contains a backup strategy, if address lists do not contain the required entries. In this case, a server asks its subdomains, its associated domains and its master to pass a request nearer to a target. If address lists contain a minimum of entries (see below), this block usually is not processed.

4.4 Dealing with Restrictions, Scalability

On one hand, our infrastructure should be scalable for a higher number (e.g., many thousands) of domains. On the other hand, each location server should be very lightweight, i.e. should not make high hardware demands. As a result, a location server could have a limited memory space for storing addresses in its list. Let l denote the maximum number of entries in the address list. To simplify the evaluation, we assume that each location server has the same space limitation. In reality, however, top-level servers may be prepared for larger lists. Our mechanism collects addresses using priorities: *1* (highest): root domains; *2*: all domains of the own hierarchy; *3*: all domains of other hierarchies. Domains with priorities 2 and 3 are further ordered by the domain level (higher levels first). The address list is filled according to these priorities: if the list is full and a new address is added, the entry with the lowest priority is dropped. For successfully passing geocast messages, at least a list of all root domains (except for the own) is necessary, thus $l \geq h\text{-}1$. Note that we do not store related or associated servers in the address list, as these links use a separate storage.

The second step ensures that in case of sufficient address space, at least all servers of a hierarchy know all servers of their own hierarchy. Thus, whenever a geocast request was directed to the target hierarchy, only one more hop is required to reach a target LS. The third step finally fills the list with domains of other hierarchies.

Fig. 5 shows the result of evaluating the message passing algorithm. Here, we count the maximum and average hops to reach the *first* target LS. The x-axis shows the limited list space in relation to the total number of domains. Not surprisingly, the average number of hops converges to 1 for higher l. Each curve has a certain break point (e.g. at 50% for h=2). At this point, address lists are capable to store all domains of the own hierarchy.

As a result, the maximum hop count to reach a target is 2: one to reach the root server of the target hierarchy and one more to reach the target LS inside the hierarchy. Assuming n_h domains for each hierarchy, where $n_h \approx n/h$, we get a maximum of two hops for $l \geq h\text{-}1+n/h$. Having less entries in the address list, but not less than h-1, message passing still is successful, but the number of hops is considerable high.

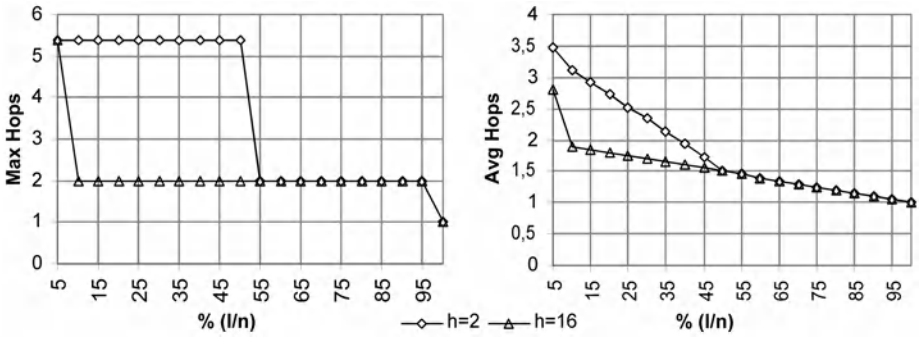


Fig. 5. Message passing with reduced address tables (n=1000)

4.5 Security Issues

Distributed systems, especially in mobile computing environments, are subject to security issues. Our security solutions are very complex, thus we can only present the ideas at this place. To protect a server or mobile node against malicious servers, a node can request an authentication certificate of the correspondence node. Authentication is proofed according to the challenge-response mechanism. A server can reject any request, if the authentication fails. This includes geocast requests as well as requests to register as a master, subdomain or associated server.

If mobile users do not want to receive any geocast message, they can register themselves in *stealth mode*. In this case, LSI only provides basic services. As the mobile user is not listed by an LLS, the system does not collect any position data. Note that protecting mobile users against malicious location servers that collect motion profiles generally is very difficult. The same unsolved problem occurs in cellphone networks. As our system is decentralized, such servers would have to cover a large area to capture motion profiles of mobile users.

A mobile user, who wants to receive geocast messages, is not willing to receive unwanted (i.e. spam) messages. As in traditional networks, an application listens for a set of pre-defined ports and ignores all messages arriving at other ports. In addition, our system offers mechanisms to discover the identity of a mobile user. As the mobile user cannot send geocast messages directly, but has to use an LLS, LSI can request a certificate of the sender for each geocast message. This information can be passed through to all receivers. We are aware that this mechanism cannot avoid spam completely, but even in traditional networks this problems is not solved.

4.6 Further Details

As mentioned above, LSI and its semantic geocast mechanism is fully implemented and tested. The following code shows how to send a geocast message with only a few lines of code:

```
LSI.startService();              // Start the runtime system
LSI.setStealthMode(false);       // I want to receive messages
byte[] msg="hello".getBytes();   // Create a message
LSI.sendGeocast(NATIVEGEOCAST,   // And send it to
     MSG_PORT,"hagen.de",msg);   // all nodes in Hagen
```

We distinguish two kinds of geocast requests: *native* requests use UDP for the last hop, i.e. from a target LS to mobile nodes. Using native requests, a receiver has to listen to a traditional UDP port to receive geocast messages. In contrast, *event-based* requests use internal protocols between the client and the LLS. Using the event-based mechanism, an application can either call a `receiveGeocast` method to wait for geocast messages or register a listener object that is called when a message arrived.

5 Conclusion and Future Work

In this paper we presented a decentralized, self-organizing approach to provide geocast services. We especially introduced the notion of *semantic geocast*, where target regions are defined by their meaning rather than by their physical area. We presented mechanisms that ensure scalability and stability, even if the servers have certain limitations concerning memory space.

LSI mainly addresses technical issues and provides a basic communication platform for location-based services. To use it in real environments, we additionally have to address organisational issues, e.g., we have to define useful hierarchies with meaningful domains. If LSI is a service inside a commercial infrastructure, e.g. a cellphone network, we need a system to charge users. Such organisational issues will be addressed in the future.

References

1. Abowd, G. D.; Atkeson, C. G.; Hong, J.; Long, S.; Kooper, R.; Pinkerton, M, 1997: Cyberguide: A mobile context-aware tour guide. ACM Wireless Networks, 3: 421-433
2. Cheverst, K.; Davies, N.; Mitchell, K.; Friday, A.; Efstratiou, C., 2000: Developing a Context-aware Electronic Tourist Guide, in Proc. of CHI'00, ACM Press
3. Dey, A., K.; Abowd, G., D., 2000: CybreMinder: A Context-aware System for Supporting Reminders, Second International Symposion on Handheld and Ubiquitous Computing 2000 (HUC2K), Bristol (UK), Sept. 25-27, 2000, LNCS 1927, Springer-Verlag, 187-199
4. Hohl, F; Kubach, U.; Leonhardi, A.; Schwehm, M.; Rothermel, K.: Nexus - an open global infrastructure for spatial-aware applications. In Proc. of the 5th Intern. Conference on Mobile Computing and Networking (MobiCom '99), Seattle, WA, USA, 1999. ACM Press
5. Imielinski, T.; Navas, J.: GPS-based Addressing and Routing, Request for Comments 2009, Nov. 1996, http://www.ietf.org/
6. Imielinski, T.; Navas, J.: GPS-based geographic addressing, routing, and resource discovery, Communications of the ACM, Vol. 42, No. 4, Apr. 1999, 86-92
7. Jos, R.; Davies, N.: Scalable and Flexible Location-Based Services for Ubiquitous Information Access, Proc. of the first Intern. Symposium on Handheld and Ubiquitous Computing HUC '99, Karlsruhe Germany, Springer-Verlag, 1999, 52-66

8. Kindberg, T.; Barton, J.; Morgan, J.; Becker G.; Caswell, D.; Debaty, P.; Gopal, G.; Frid, M.; Krishnan, V.; Morris, H.; Schettino, J.; Serra, B.; Spasojevic, M., 2000: People, Places, Things: Web Presence for the Real World, Proc. 3rd Annual Wireless and Mobile Computer Systems and Applications, Monterey CA, USA, Dec. 2000
9. Leonhardt, U.: Supporting Location-Awareness in Open Distributed Systems, PhD Thesis, University of London, 1998
10. Marmasse, N.; Schmandt, C., 2000: Location-aware Information Delivery with ComMotion, Second International Symposion on Handheld and Ubiquitous Computing 2000 (HUC2K), Bristol (UK), Sept. 25-27, 2000, LNCS 1927, Springer, 157-171
11. Navas, J.; Imielinski, T.: GeoCast – Geographic addressing and routing, Proc. of the 3rd ACM/IEEE inter. conf. on Mobile Computing and networking, Sept. 26-30, 1997, 66-76
12. Open GIS Consortium, OpenLS Home Page, www.openls.org
13. Perkins, C. E.; Bhagwat P.: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, Proc. of the SIGCOMM '94 Conference on Communications, Architectures Protocols and Applications, Aug. 1994, 234-244
14. Pradhan, S.: Semantic Locations, Personal Technologies, Vol. 4, No. 4, 2000, 213-216
15. Roth, J.: A Communication Middleware for Mobile and Ad-hoc Scenarios, Int. Conf. on Internet Computing (IC'02), June 24-27, 2002, Las Vegas, Vol. I, CSREA press, 77-84
16. Roth, J.: Context-aware Web Applications Using the PinPoint Infrastructure, IADIS Intern. Conference WWW/Internet 2002, Lisbon, Portugal, Nov. 13-15 2002, IADIS press, 3-10
17. Roth, J.: Flexible Positioning for Location-Based Services, IADIS International Conference e-Society 2003, Lisbon, Portugal, 3-6 June 2003, IADIS Press, 296-304
18. Schilit, B.; Adams, N.; Want, R., 1994: Context-Aware Computing Applications, Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, USA, 1994

based applications possible by better resource description. The second development, which is described in more detail in this paper, is the generalization of web-based functionality as web services. While web information systems employ a broad variety of not necessarily compatible technologies, web services make functionality and information available via standardized descriptions and protocols.

All three stages of web development have been picked up by the language technology and terminology management community:

1. Terminology lists and dictionaries have been published as static hypertext on the web for many years (phase 1)
2. More recently, language technology applications like stemming or information extraction have become available as web information systems (phase 2).
3. Finally, the language technology community is beginning to transform web information systems into more flexible web services (phase 3).

During the last years a great number of language resources have been made available electronically, esp. on the web. An overview of currently available language technology tools and resources may be found at the *Language Technology World* (*http://www.lt-world.org/*), a comprehensive repository for all kinds of language technology-related resources and applications. While many areas of language technology and terminology management have been covered so far, some shortcomings have become obvious as well:

- Similar steps have to be repeated to look up information on the same words or concepts in several databases.
- Databases from different vendors or organizations have different user or application programming interfaces.
- Different databases may have different data structures or query capabilities.

In this paper we are going to explore the prospects of using web service technology for offering language technology-based information like large language corpora, dictionary lookup or text mining functions. We illustrate using web services in language technologies in the context of the project *Deutscher Wortschatz* developed at Leipzig University, CS Institute (see [Quasthoff & Wolff 00], [Heyer, Quasthoff, Wolff 02], and *http://wortschatz.uni-leipzig.de* for more information on this project).

2 Web Service Standards and Modeling Criteria

A key issue in web service development is standardization, as only by providing common grounds for web service definition essential features of the third phase of web development can be achieved:

- Modular composition of different web services by loosely coupling distributed components
- Integration of web services into complex applications

- Universal access to web services from different types of client software and client applications.

2.1 Current Web Service Standards

Currently, three standards have been proposed which address key aspects of web services and which are also employed in the examples given below:

- SOAP, a messaging protocol for web service deployment (cf. [Gudgin et al. 02]),
- the Web Service Description Language (WSDL, cf. Chinnici et al. 02]) which provides a common framework for the description of web service functionality, and
- UDDI (Universal Description, Discovery and Integration), a standard for web service directories and lookup (web service discovery, cf. Bellwood et al. 02]).

In a typical web service-based architecture, these standards realize the core functions of service description, discovery, and delivery / service execution. Figure 1 illustrates their interrelationship (see [Ferris & Farrell 03:31], [Vinoski 02a:90]).

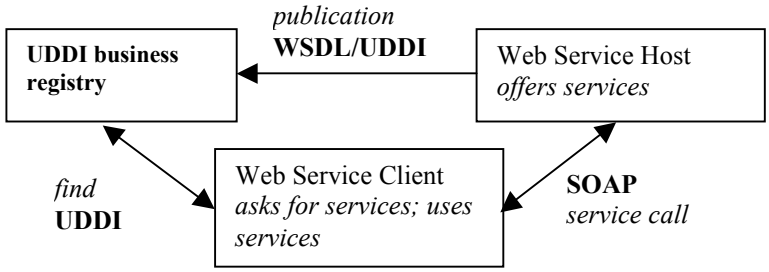


Fig. 1. Components, standards, and functions in the standard web service architecture

In a technical perspective, web services pick up the well-known idea of remote procedure calls, as arbitrary functionality can be accessed via functions calls wrapped up as SOAP messages being transported on the web. From a (na□ve) user□s perspective, a web service simply returns information to a given question.[1]

2.2 Criteria for Web Service Definition

While the basic technologies for web service definition have already been defined, a standard methodology for web service modeling has yet to be developed (see [Schranz 98]. The following table is a first attempt to identify relevant criteria for web service selection and modeling.

[1] Remote procedure calls are certainly the most prominent model of communicating via web services (see [Vinoski 02b]. More complex usage scenarios have been identified by the World Wide Web Consortium□Web Service Ar chitecture Group (see [Haas & Orchard 03]).

Criterion	Short Description	Example
General		
Domain	Application area	Language Services, Financial Services
Web Service Standards	Usage of standards for web service application integration	UDDI, WSDL, SOAP, BPEL4WS[2]
Technical Basis	Software infrastructure used for developing / hosting	Java, Java 2 Enterprise Edition, Apache Axis Module
Modeling Aspects		
Data Structures	Structure and organization of data delivered via web services	Atomic (single data item), aggregate
Mode of Operation	Type of information flow triggered by a web service call	push, pull, request for pull
Dialogue model	Type of web service usage scenario	query/response or RPC-model / more complex scenarios
Session Model	Introduction of a session concept	stateless, stateful
Access, Load, and Control		
Availability	Access restrictions for different user types	unrestricted / restricted, limitation of concurrent web service calls
Load	(estimated) server-side processing time and / or memory load	data volume, processing demand
Priority	Possible ranking of request types	ranking by estimated load / user type / query type
Description and Discovery		
Naming	Usage of a common naming convention	Java Naming Convention (*operationReferredElement*, cf. http://java.sun.com/docs/codeconv/)
Description	description vocabulary used for a web service	Ontology, vocabulary used for UDDI
Metadata	additional information for web service discovery / categorization	Ontology and / or Metadata Standard, to be defined

Table 1. Web Service Modeling Criteria

3 Language Technology Web Services

Language technology applications may benefit from using web service standards in several ways:

- Information becomes available via standardized interfaces and access mechanisms.
- All relevant details concerning data structures and implementation of components can be hidden from the user.

[2] The *Business Process Execution Language for Web Service* (BPEL4WS) is a recently introduced standard which addresses, among other issues, the aspects of web service composition and coordination and the transactional aspects of web services, see [Thatte et al. 03].

- The same type of service (e. g. stemming, key word extraction or text translation) may be offered by different developers. Using web service registries, a client application in need of a specific language technology web service may ultimately select (or change) web services even during execution.

A number of practical examples for language-related web services may be found at *http://www.remotemethods.com/home/valueman/converte/humanlan*, among them AltaVistas *Babelfish* technology for multilingual text translation.

3.1 Web Services «avant la lettre» in Language Technology

For several years we have been developing web-based applications in the language technology domain which are presented using a web browser and for which application integration mechanisms exist which make use of web-based protocols like HTML. Among these applications are

- A web site as general presentation platform for text mining results and dictionary look-up (*http://wortschatz.uni-leipzig.de*).
- Services for key word extraction from arbitrary texts (cf. [Faulstich et al. 02]).
- Tools for daily media analysis, key concept extraction and visualization (cf. [Quasthoff et al. 03]).
- Web-based services for knowledge management, especially extraction of concept networks from large text corpora (cf. [Bhm et al. 02]).

In all cases the web is used for presenting (text) analysis results. In the cases of the key word extraction tools an integration into existing Content Management Systems (CMS) is possible via the HTTP protocol, while for knowledge management also XML-based export routines for the XML Topic Map standard (XTM) exist.

The examples mentioned above represent both, complex analysis tasks as well as aggregate multimedia presentation items. They can be classified as web-based information services which do not yet make use of web service standards. A first step in achieving this goal is the migration from non-standard APIs to web services for the basic language information building blocks. The following basic tools and resources are available in the context of the *Leipzig Wortschatz* project mentioned above (see [Quasthoff & Wolff 02]):

- Large text corpora from different domains and temporal ranges as wells as in various languages like German, English, French etc.
- A comprehensive dictionary of inflected forms with a rich data structure for each entry (frequency information, semantic attributes, morphological and syntactical information).
- Additional features extracted from text via text mining tools like collocations for each entry.
- A set of tools for corpus and dictionary setup, analysis, and maintenance.

3.2 Types and Examples of Linguistic Web Services

Two of the most important criteria for web service modeling are data model and user needs: Different web service methods may be categorized either *structurally* with respect to the underlying database model developed, or concerning the *information need* modeled by a web service method. As the structural aspect is a genuinely technical one, we will concentrate on different typical information needs in the following. In general, we follow a bottom-up approach for modeling web services, starting with simple information needs and data structures. The resulting web services may then be re-used for more complex web service-based applications: A key word extraction task, for example, makes heavy use of simpler query types like stemming, compound decomposition or look-up of basic frequency information.

At the same time we could draw from our experience in which types of information needs tend to be more relevant for a broad range of language technology applications. The following lists give examples of web services modeled for the different information needs within our application domain:

Querying Word-Related Information
The most frequently used type of web services encompasses all service requests which deliver information on single linguistic entities (dictionary lookup queries). Examples are:

getWordBaseforms	base form(s) of a word
decomposeWord	get head and modifiers of a compound
getWordSpelling	get known words with small Levenshtein distance from a given string
getWordLanguage	language identification of a word or sentence
getExample	retrieves an example sentence for a given word, (sub-selection with additional criteria like sentence length, text type, date)

Statistical Information and Text Mining Results
Text mining tools store frequency data as well as significant relation between words and concepts in each text corpus database and can be queried by services such as:

getWordFrequency	frequency of a word
getHitlist	generation of a list of words which are most significant for a given text corpus
getCollocates	list of collocations (significant co-occurrences in the same sentence) for a given word

Tagging and Information Extraction
The following web service functions are examples for more complex tasks which go beyond simple database lookup and involve the execution of server-side processing modules like key word or name extraction tools:

getSentencePOSTags	(POS-tagging)
getSentenceNames	extraction of proper names from a given sentence
getTextKeywords	extract keywords from a given text

Linguistic Knowledge Services – Developing Web Services in Language Technology

Fabian Schmidt, Christian Wolff

Chemnitz University of Technology
Faculty of Computer Science, Media Computing Dept.
Straße der Nationen 62
09107 Chemnitz, Germany
{christian.wolff, fabian.schmidt}@informatik.tu-chemnitz.de

Abstract. In this paper we describe the application of web service standards in the language technology domain. Starting from a short review of web application development the notion of web services is introduced and relevant standards in this area are briefly described. Following a motivation for converting language technology applications into web services we describe relevant criteria for web service modeling and give examples for such services based on a large language information infrastructure developed over the last years. Finally, some technical details illustrate our web service prototype.

1 Introduction

Web services have come to be regarded as a key technology for developing web-based applications. Stable standards like the Simple Object Access Protocol (SOAP) or the Web Service Descriptions Language (WSDL) along with numerous web service-based applications allow the realization of web services in various domains (cf. Kreger 2003:29). In its short, 10+ year history, the web has seen dramatic technological change. It may be categorized into three phases of development (see also [Preece & Decker 02:15] who propose a simpler, two-phase development model):

1. Initially the web was designed as a means for electronically publishing *distributed hypertexts* based on simple standards and protocols (HyperText Markup Language (HTML), HyperText Transfer Protocol (HTTP), Uniform Resource Identifier (URI) and introducing the Web Browser as client software.
2. The second phase starting in the later part of the nineties brought *information systems* to the web: Web information systems as a new software development paradigm allow for the presentation of arbitrary information system functionality via common standards and common client software (the ubiquitous web browser).
3. The third phase which has begun only very recently introduces two additional innovations: On the one hand, standards for creating richer descriptive information sets on the web by providing meta information have been introduced, among them the Resource Description Framework (RDF) or the Topic Map ISO standard. The general aim is the creation of the „semantic web" which makes more complex web

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 229-238, 2003.

Meta Data on Language and Corpora
At the corpus (or database) level, metainformation on text collections is also available via web service calls:

listCorpora	get a list of available corpora
listLanguages	list of available languages with linguistic and statistical data
getCorpusInfo	information on a specific text corpus (size. Date, text type(s), language)

Complex Processing Tasks, Corpus Management, Application Tasks
While the above lists are fairly straightforward as they define web services which correspond to structurally simple database queries, more complex tasks can be made accessible by web service standards as well. Among them are text mining processes which can be triggered by web services specifying raw text location, language and further parameters relevant for corpus processing. We are currently working on the implementation of web services for these asynchronous processing tasks - a text mining task triggered by a web service may take several hours and needs another invocation model than a database lookup service.

3.3 Architecture and Implementation

While most web services for querying singular information items can be modeled by RPC-style service calls, more complex processing tasks require a session management which allows for the administration of web service users and their access rights. For this purpose a *UserHandler* module opens and manages web service sessions and checks if incoming service requests may be handled given the users access rights. An overview of this server-side architecture is given in figure 2 below.

For the implementation of Web Services we employ a number of freely available software components:

- Apache Web Server (see http://httpd.apache.org/)
- Apache Tomcat Application Server Engine (see http://jakarta.apache.org/tomcat/)
- Apache AXIS SOAP Implementation (see http://ws.apache.org/axis/)
- Cape Clear WSDLEditor (http://www.capescience.com/downloads/wsdleditor/)
- Java Software Development Kit V. 1.4.2 (see http://java.sun.com/j2se/)

WSDL descriptions and a test client written in Java are available for the web services described above,. In the appendix, code examples for a WSDL service description, a SOAP request and an actual service call in the client program (excerpt) are given.

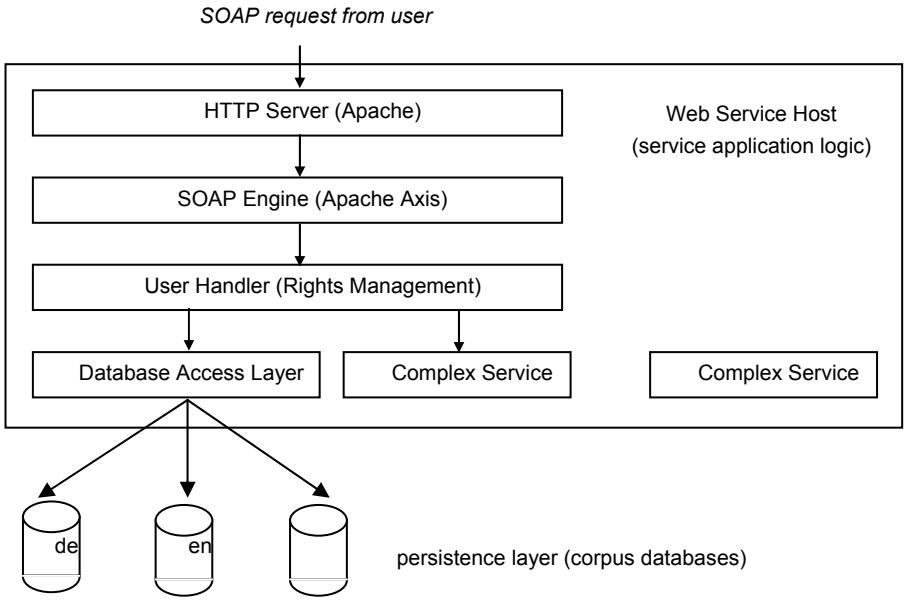


Fig. 2. Architecture Overview

4 Conclusion

In the past months we have been developing web services for terminological information which may be used for information presentation as well as integration into language technology applications. We are working on a complete set of web service functions, atomic as well as composite, for the most pressing needs of our users in the language technology and terminology management area.

While at the level of implementation technology standards for offering such services have become widely used, further standardization is needed for service naming and easier service discovery. Due to the lack of reliable registries as well as deficits in standardized service naming, we are not making use of UDDI yet.

5 Appendix: Web Service Examples

The following examples show excerpts from web service definition and request files. First, an example for a WSDL service description is given (ch. 5.1) in which a web service function for retrieving the base form of a given word is retrieved. Second, a SOAP request message for selecting the frequency class of a word is shown (ch. 5.2). Finally, we show a small excerpt from the a client side test application (Java) in which web services are called (retrieving base forms for an array of words).

5.1 WSDL Example (Excerpt)

```
<definitions name="urn:LdbApi" targetNamespace="urn:LdbApi">
<types>
  <xsd:complexType name="BaseFormResult">
    <xsd:all>
      <xsd:element name="baseforms" type="tns:StringArray"/>
    </xsd:all>
  </xsd:complexType>
</types>
<message name="GetBaseFormResponse">
  <part name="result" type="tns:BaseFormResult"/>
</message>
<portType name="LdbApiPort">
  <operation name="getWordBaseforms">
    <input message="tns:GetBaseFormRequest"/>
    <output message="tns:GetBaseFormResponse"/>
  </operation>
</portType>
<binding name="LdbApiBinding" type="tns:LdbApiPort">
  <soap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="getWordBaseforms"/>
</binding>
</definitions>
```

5.2 SOAP Request: Stemming (Excerpt)

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope  soapenv:encodingStyle=
         http://schemas.xmlsoap.org/soap/encoding/ xmlns: (...)>
  <soapenv:Body>
    <ns1:getWordFrequencyClass xmlns:ns1="urn:LdbApi">
      <request href="#id0"/>
    </ns1:getWordFrequencyClass>
    <multiRef id="id0" soapenc:root="0" xmlns:ns2="urn:LdbApi"
              xsi:type="ns2:WordWithOptionalLanguage">
      <word xsi:type="xsd:string">Sachsen</word>
      <language xsi:type="xsd:language" xsi:nil="true"/>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

5.3 Example Java Client Application Code (Excerpts)

```
// ...
LdbApiServiceLocator service = new LdbApiServiceLocator();
String serviceUrl = service.getLdbApiPortAddress();
// ...
LdbApiPort port = service.getLdbApiPort(new URL(serviceUrl));
for(int j=0; j<args.length; j++)
{
  System.out.println("Baseforms for "+args[j]+":");
  String []result=port.getWordBaseforms(args[j]).getBaseforms();
}
// ...
```

Acknowledgements

The authors would like to thank the Natural Language Processing Department at Leipzig University's Computer Science Institute (Gerhard Heyer) and the project *Deutscher Wortschatz* (Uwe Quasthoff) located there for their support in the preparation of this paper and the ideas expressed therein.

References

Bellwood, T. et al. (2002). UDDI Version 3.0, Universal Description, Discovery and Integration (UDDI) Project, Published Specification, July 2002, http://uddi.org/pubs/uddi_v3.htm.

Böhm, K.; Heyer, G.; Quasthoff, U.; Wolff, Ch. (2002). Topic Map Generation Using Text Mining. In: J.UCS Journal of Universal Computer Science 8(6) (2002), 623-633.

Chinnici, R. et al. (2003). Web Services Description Language (WSDL) Version 1.2. World Wide Web Consortium Working Draft, June 2003, http://www.w3.org/TR/wsdl12.

Faulstich, L. C.; Quasthoff, U.; Schmidt, F.; Wolff, Ch. (2002). Concept Extractor - Ein flexibler und domänenspezifischer Web Service zur Beschlagwortung von Texten. In: Hammwöhner, R.; Wolff, Ch.; Womser-Hacker, Ch. (edd.) (2002). Information und Mobilität, Proc. 8. International Symposium on Information Science, Regensburg, October 2002, 165-180.

Ferris, Ch.; Farrell, J. (2003). What are Web Services? In: Communications of the ACM 46(6) (2003), 31.

Gudgin, M. et al. (2002). SOAP Version 1.2 Part 1: Messaging Framework. World Wide Web Consortium Working Draft, June 2002, http://www.w3.org/TR/soap12-part1.

Haas, H.; Orchard, D. (2003). Web Services Architecture Usage Scenarios. World Wide Web Consortium Working Draft, May 2003, http://www.w3.org/TR/ws-arch-scenarios/.

Heyer, G.; Quasthoff, U.; Wolff, Ch. (2002). Knowledge Extraction from Text: Using Filters on Collocation Sets. In: Proc. LREC-2002. Third International Conference on Language Resources and Evaluation. Las Palmas, May 2002, Vol. III, 241-246.

Kreger, H. (2003). Fulfilling the Web Service Promise. In: Communications of the ACM 46(6) (2003), 29-34.

Quasthoff, U.; Wolff, Ch. (2000). An Infrastructure for Corpus-Based Monolingual Dictionaries. In: Proc. LREC-2000. Second International Conference on Language Resources and Evaluation. Athens, May/June 2000, Vol. I, 241-246.

Quasthoff, U.; Richter, M.; Wolff, Ch. (2003). Medienanalyse und Visualisierung: Auswertung von Online-Pressetexten durch Text Mining. In: LDV-Forum 18(1,2) (2003), 452-459.

Preece, A.; Decker, M. (2002). Intelligent Web Services. In: IEEE Intelligent Systems 17(1) (2002), 15-17.

Schranz, M. W. (1998). World Wide Web Service Engineering. Object-Oriented Hypermedia Publishing. Ph.D. Thesis, Vienna University of Technology.

Thatte, S. et al. (2003). Business Process Execution Language for Web Services Version 1.1. Specification, second public draft release, May 2003, http://www-106.ibm.com/developerworks/library/ws-bpel/.

Vinoski, St. (2002a). Web Services Interaction Models, Part 1: Current Practice. In: IEEE Internet Computing 6(3) (2002), 89-91.

Vinoski, St. (2002b). Putting the Web into Web Services. Web Services Interaction Models, Part 2. In: IEEE Internet Computing 6(4) (2002), 90-92.

The Application of XML Templates for Task Preparation in an Electronic Memory Aid System

Andrei Voinikonis and Klaus Irmscher

Leipzig University, Institute of Computer Science
Chair of Computer Networks and Distributed Systems
Augustusplatz 10-11, D-04109 Leipzig, Germany
{voinikon,irmscher}@informatik.uni-leipzig.de

Abstract. One of major requirements of the electronic memory aid system is to provide a simple way to set valid tasks for patients with memory disturbances. This article is discussing application of templates for task description to solve this problem, the presentation of templates as JDOM at run time and as XML documents in their persistent form, and application of XSLT transformations to obtain the final description of tasks for mobile devices.

1. Introduction

The appearance of mobile devices such as smart phones and palmtop computers with a permanent Internet access allows introducing new kinds of services. The treatment of patients with memory disturbances is one of possible fields for application of communication technology. The patients now can receive operative instructions from their therapists, while the therapists can effectively control the execution of these instructions by patients and correct them [3].

The aspects of this application are being researched at Leipzig University since 1998. The concept is to furnish the patient with a mobile device containing the descriptions of the tasks. Such device reminds the patient when the next task or the part of the task should be performed. In addition, the device traces the execution of the task [2].

The task can be set both by the therapist and by a patient himself or by members of his family. This requires the remote access to the device. The process of the task setting should be also simplified. Designing every new task from a scratch is a very time consuming and fault-prone process. To avoid this the tasks with a similar structure[1] can be consolidated in the groups and each group can be described with one action plan. Such action plan must be prepared by a qualified psychologist in advance. Usage of the action plans allows to avoid mistakes confusing the patient.

If the action plan for a group of tasks is developed and prepared for task processing, it's enough to set values of some parameters in order to set a new task. The action plans are implemented and stored as XML templates.

[1] Sequence of actions

T. Böhme, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 239-250, 2003.

This article discusses the implementation of task processing part of the memory aid system. The focus is put on application of XML and JDOM for the template implementation and XSLT transformations for the generation of task description.

2. Task Processing

The memory aid system (MEMOS[2]) is aimed to help patients with memory disturbances. Patients use special palmtop computers called PMA[3], which hold the tasks to perform. The tasks are downloaded from the server providing an opportunity for caregivers to set the tasks via the Internet [2].

2.1. System Requirements

The requirements for task processing part of electronic memory aid system are based on the recommendations and demands of psychologists or on desired system architecture [2].

Usage of various mobile devices. As mentioned above the technological development permanently results in appearance of new more reliable mobile devices. These new devices can be used by patients. The memory aid system has to be able to adopt such devices.

Simplicity of task setting. Since task setting process should be carried out mostly by therapists, there should be means for simple and fast way of setting valid task. This is the one of the major requirements to the system.

Correctness of produced tasks. The system should avoid mistakes confusing the patients and caregivers that are mostly the persons without or with very little computer experiences. In particular, parts of tasks and their parameters should hold consistent.

Extensibility. Since system uses plans for task preparation, it should be able to accept new plans at runtime from remote clients. The system should persistently store the accepted plans.

2.2. System Implementation

To satisfy the above-mentioned requirements the following decisions are made:

To avoid system updating due to introducing of new kinds of devices a common interface has to be developed. The XML technology is most suitable for it [9]. The

[2] **M**obile **E**xtensible **M**em**o**ry Aid **S**ystem
[3] **P**ersonal **M**emory **A**ssistant

special XML based language named M2[4] was developed to describe the tasks for PMA's. The focus of the M2 specification is put on a simple conversion between the modeled data structures and the XML structures. Because some PMA's have very limited processing capabilities, the language structure was held very simple. An example of task description in M2 language for medicine taking is shown in listing 1.

```
<deck label="Medicine Taking" id="D2:1:0">
 <control>
  <time start="100" end="200"/>
  <timer time="50"/>
 </control>
 <card id="D2:1:0_0">
  <loop time="20" reference="D2:1:0_0"/>
  <text>Please take Aspirin now!</text>
  <button label="OK"/>
 </card>
</deck>
```

Listing 1. A simplified task description for medicine taking on PMA.

To simplify the process of task preparation, it is divided into two steps. The first step is to develop one plan for a group of similar tasks. The plan is a graph [1], whose nodes are instructions to perform an atomic action and whose edges are the handlers of events generated either by the patient himself or by the time flow control elements. The application of graphs for the plan representation allows both fixing the structure of nodes and edges and at the same time keeping an opportunity to implement diverse plans. The second step is simply to set the task with its own parameter values, which should be set during of task creation.

We made the choice to use a template-based system. An XML document, which we call a template, defines a common structure of the task, specifies parameters being used and locates positions in the text, where the placeholders must be substituted with parameter values. Each template implements one plan for the specific group of task. The tasks of the same group differ only in their parameter values. Application of the templates helps the caregivers to produce the correct tasks.

Although the M2 language meets the requirements of the description of tasks well, it is not suitable for template description due to its simplicity. In particular, the M2 language does not support the definition of parameters and substitutions. Therefore, the XML based language named MTL[5] was developed to describe the templates on the server side. The MTL is built on basis of M2. An example of the template for the medicine taking task is shown in listing 2. MTL document for the template contrary to M2 document for the corresponding task has an additional part containing the parameters. The description of the task structure is quite similar in both documents with exceptions for the attributes description, which should be substituted, and for the elements, which contain the placeholders. The structure of MTL is discussed in part 3.

[4] MOBTEL Markup Language v.2
[5] **M**EMOS **T**emplate **L**anguage

```
<template title = "Medicine Taking">
 <prefix/>
 <data_section>
  <parameter id="m1" prompt="medicine to take"
             type="string" value="medicine" max="50"/>
  <parameter id="start0" prompt="start time"
                    section="0" type="date" value=""/>
  <parameter id="end0" prompt="end time" section="0"
                                type="date" value=""/>
 </data_section>
 <structure_section>
  <deck label="Medicine Taking" id=":0">
   <control>
    <time idrefstart="start0" idrefend="end0"/>
    <timer time="50"/>
   </control>
   <card id=":0_0">
    <loop time="20" reference=":0_0"/>
    <text>Please take <var pref="m1">medicine</var>
                                          now!</text>
    <button label="OK"/>
   </card>
  </deck>
 </structure_section>
</template>
```

Listing 2. A simplified template of task for medicine taking.

Application of the templates determines architecture of the system. A set of templates has to be managed dynamically. A psychologist further called a template designer must have a possibility to add or remove templates at run time without recompilation of the source code or system restart. The template designer can be a person with very limited computer knowledge. Therefore, the process of posting of new templates to the server and their preparation for task processing has to be automated.

To meet the requirement of extensibility, the templates are not implemented in program code but are handled as system data in XML format that can be added to or removed from the server.

2.3. Task Processing

At first, the template designer develops the templates with the help of a special graphical tool. The designer transmits the created template to the server where it is automatically prepared for task processing (Fig.1, action 1). At the same time, the MTL document is stored in the database on the server side, which guarantees the durability of the system. The template designer can always download the existing plan from the server for modification. In addition, the graphical tool allows him to save the plan in a local file as the intermediate result for further evaluation.

After transmitting to the server the MTL document is parsed to build the object graph as JDOM tree consisting of *standard* components. Such trees and their

components can be outputted as XML documents or DOM objects on demand [4]. A simple object adapter written in Java contains the JDOM and controls the access to it [8]. The validation of the template takes place both during the transformation of the MTL document to JDOM by parsing at load time and after creation of the template by a special function of the graphical tool, which performs a semantic check. When the JDOM is built, it can be immediately used by caregivers to set the tasks.

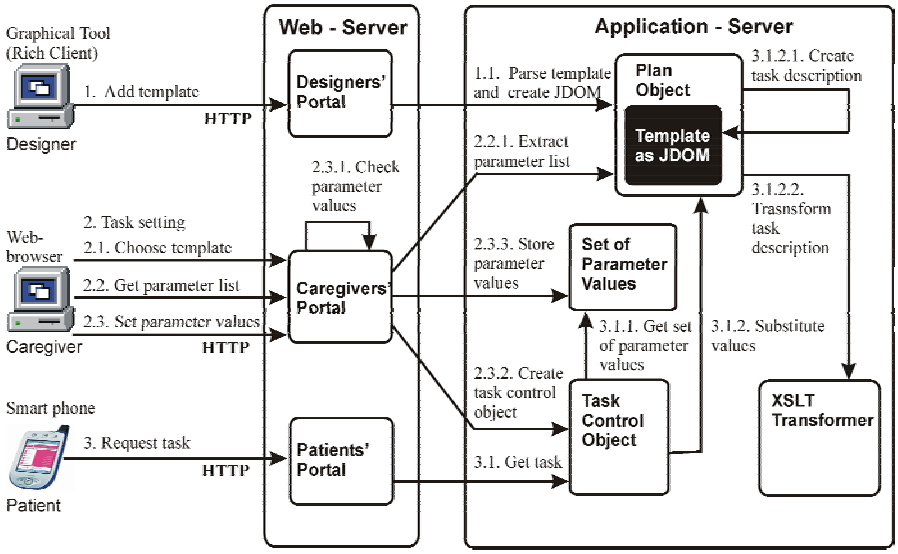


Fig. 1. A simplified process of task processing in memory aid system. Usually the actions marked as 1, 2 and 3 are broken up in time.

For this purpose, the caregiver chooses a relevant plan and gets the list of parameters as a HTML form to fill (Fig.1, actions 2.1, 2.2). A special JSP generates this form dynamically based on the parameter list [5]. Extraction of the parameters is a simple procedure because all templates hold them in a separate section with an identical structure[6]. It is executed by a function of the object adapter (Fig.1, action 2.2.1). Each parameter entry contains, inter alia:

- the prompt that explains the meaning of parameter. It is shown to the caregiver in HTML form;
- the type for input check;
- the default value that can be adopted by the caregiver;
- the type relevant limitation for input or input check.

After setting of parameter values the caregiver submits them to the server (Fig.1, action 2.3). The server validates the input: the type compliance, the limitation and others (Fig.1, action 2.3.1). If the values stand the test, the task control object is created with an unique identifier and the submitted values of task parameters (Fig.1, actions 2.3.2, 2.3.3). These values are used both for controlling the task life circle by

[6] as children of one node in JDOM tree at run time

the system and for producing the valid task description in M2 language on demand. The separate usage of parameters and templates provides a significant gain in storage and transmission of documents.

If a task with a valid identifier is requested, the system finds the set of the parameter values for this task and transmits them to the corresponding template (Fig.1, actions 3.1.1, 3.1.2). The adapter substitutes the values of parameters in the JDOM with the transmitted values. After that adapter generates the task description in MTL format and restores the default values of parameters in JDOM (Fig.1, action 3.1.2.1). However, the created document must still be converted into M2 format for PMA. It is executed by a simple XSLT transformation (Fig.1, action 3.1.2.2), which is described in part 4.

3. Template Structure

MTL is developed to describe the templates for documents in M2 language. All elements in MTL necessary for description of the task structure are derived or copied from M2 elements. The syntax of MTL is defined with a DTD[7], whose structure provides the high effect of validation by parsing of the documents. In particular, this is achieved by usage of attributes of the type ID and IDREF. All elements that describe the parameters are marked with attributes of ID type. The elements that serve as placeholders possess the attribute of IDREF type to refer the element that contains the corresponding parameter. Usage of invalid references and duplicates of identifiers are detected by parsing.

Listing 2 shows the template of a task for medicine taking. The elements and attributes that have to be changed by creation of the final task description are emphasized by the italic font.

Root-element of DTD `<template>` has three children.

```
<!ELEMENT template (prefix, data_section,
                                    structure_section)>
```

The MTL template contains numerous identifiers that must possess unique values and references on them. However, the several task descriptions created by usage of the same template can be transmitted to the same mobile device, what can result in conflicts, because in this case the decks possess the identical identifiers. Thus, it is needed to resolve identifiers and references. The `<prefix>` element serves for setting the unique identifier of the task for the newly generated document. Its value must be set before the generation process started.

```
<!ELEMENT prefix (#PCDATA)>
```

Further DTD consists of two parts: `<data_section>` and `<structure_section>`. The partition into two parts and declaration of all parameters in the first part allows us to achieve the following goals:

- The repeated redundant definition of the used parameters throughout the document is avoided.

[7] **D**ocument **T**ype **D**efinition

- The document structure becomes more robust: changes in the data part do not affect the structure part and vise versa.
- The parameter extraction is simplified. Instead of examination of the whole document, it is enough to examine only the small part of it. This part has a special uniform structure optimized for it.
- The parameter substitution is simplified. Instead of replacing the values of parameters in the whole document, it is enough to replace the information only in the small part of it.
- Mistakes, which can be made in searching or replacing of the parameters through the whole document, are avoided.

3.1. Data Part of MTL DTD

The data part of DTD contains the parameter definitions. This part of template specifies all parameters used in the template. A compact presentation simplifies the extraction of parameters required for the caregiver to set a new task.

```
<!ELEMENT data_section (parameter*)>
```

<parameter> element specifies one of the parameters used in template.

```
<!ELEMENT parameter EMPTY>
<!ATTLIST parameter
   id        ID       #REQUIRED
   section   CDATA    #IMPLIED
   prompt    CDATA    #REQUIRED
   type      CDATA    #REQUIRED
   max       CDATA    #IMPLIED
   value     CDATA    #REQUIRED>
```

Attributes of the <parameter> element are:

id	- The value of the id attribute is used to refer back to the element later.
section	- This attribute is used to order parameters at the creation of HTML form.
prompt	- This attribute explains the meaning of the parameter.
type	- This attribute specifies the type of parameter for the test of input.
value	- The value attribute is used to save the default value of the parameter. The user can change or accept the value by filling of HTML form.
max	- The attribute is used to specify the type relevant limitation of parameter, for example maximal length for string.

3.2. Structure Part of MTL DTD

The structure part of DTD specifies the description of a task and, in particular, interdependency of document elements. All elements are almost similar to the elements of the M2 language.

```
<!ELEMENT structure_section (...)>
```

The elements, which should use parameters from the data part, get placeholders for these parameters. Each placeholder has a reference as an attribute of `IDREF` type to the corresponding parameter. The value of `reference` attribute is equal to the value of `id` attribute of the referenced parameter.

A special element `<var>` serves as a placeholder for a parameter value within an item body, for example in `<text>` element:

M2 DTD	**MTL DTD**

`<!ELEMENT text #PCDATA>`	`<!ELEMENT text (#PCDATA|var)*>`
	`<!ELEMENT var #PCDATA>`
	`<!ATTLIST var`
	`pref IDREF #REQUIRED>`

Otherwise, if an attribute value should be parameterized, such attribute is converted in MTL to the reference on the used parameter. Its name obtains a prefix `idref` to distinguish it from another attributes that must not be substituted. For example in the `<time>` element:

M2 DTD	**MTL DTD**

`<!ELEMENT time EMPTY>`
`<!ATTLIST time`
`start CDATA #REQUIRED`
`end CDATA #REQUIRED`
`...>`
	`<!ELEMENT time EMPTY >`
	`<!ATTLIST time`
	`idrefstart IDREF #REQUIRED`
	`idrefend IDREF #REQUIRED`
	`...>`

4. XSLT Transformation

To produce a new document in M2 format first the document in MTL format is created, where the default values of parameters are substituted by the parameter values of the requested task. Then this document is transformed to the final document in M2 format. Listing 3 shows the prefix and the data part of the template for medicine taking task after the substitution of parameter values. The substituted values are emphasized by the italic font.

```
...<prefix>D2:1<prefix/>
   <data_section>
     <parameter id="m1" prompt="medicine to take"
              type="string" value="Aspirin" max="50"/>
     <parameter id="start0" prompt="start time"
               section="0" type="date" value= "100"/>
     <parameter id="end0" prompt="end time" section="0"
                          type="date" value= "200"/>
  </data_section>...
```

Listing 3. Prefix and data part of template after substitution of parameter values.

Since the description of the task structure is quite similar in both formats, the transformer simply rewrites the content of the structure part into the final document, and by that resolves the identifiers and the references inside of the structure part. In addition, the references on parameters are substituted by their values while rewriting. The following templates are used to copy <deck> contents into the final document:

```
<!-- Copy decks -->
<xsl:template match = "structure_section/deck">
  <xsl:copy>
    <xsl:apply-templates select = "@*|node()"/>
  </xsl:copy>
</xsl:template>

<!-- Copy element contents of any node-->
<xsl:template match = "node()">
  <xsl:copy>
    <xsl:apply-templates select = "@*|node()"/>
  </xsl:copy>
</xsl:template>
```

All <var> elements are substituted by the values of <value> attribute of the referred parameters with assistance of the following template. The priority of this template is higher than the priority of the previous template:

```
<!--Substitute a parameter value instead of var
element.-->
<xsl:template match="var">
  <xsl:value-of select = "id(@pref)/@value"/>
</xsl:template>
```

To resolve the identifiers and references, the content of the <prefix> element is concatenated by values of the attributes. The value of the prefix is contained in the variable called prfx to avoid more processing. The following template is used to resolve relations:

```
<!-- Resolve id and reference attribute. -->
<xsl:variable name="prfx" select="/template/prefix"/>
<xsl:template match = "@reference|@id">
  <xsl:attribute name = "{name()}">
    <xsl:value-of select = "concat($prfx,.)"/>
  </xsl:attribute>
</xsl:template>
```

The presence of a corresponding element in the data part and the presence of idref prefix are checked at the substitution of attributes. If both conditions are fulfilled, the prefix is cut from the name and the substitution is performed with assistance of the following template. The priority of this template is lower than the priority of the previous template. All other attributes are copied without changes.

```
<!-- Copy or substitute attributes if they refer a
                        parameter in data_section. -->
<xsl:template match = "@*">
  <xsl:variable name="pvalue" select="id(.)/@value"/>
  <xsl:choose>
    <xsl:when test = "$pvalue and
```

```
                    starts-with(name(),'idref') ">
      <xsl:attribute name = "{substring-
                             after(name(),'idref')}">
        <xsl:value-of select = "$pvalue"/>
      </xsl:attribute>
    </xsl:when>
    <xsl:otherwise>
      <xsl:copy>
        <xsl:apply-templates select = "@*"/>
      </xsl:copy>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

Listing 1 shows the final description of the task for medicine taking. The changed elements and attributes are emphasized by the italic font.

As shown above the given XSL transformation is simple and universal. The application of XSLT transformation allows using the *standard* JDOM components for the plan graphs, which essentially simplifies implementation of the system [10].

5. Related Works

The idea of using templates is not new. There exist a number of related works dedicated to usage and description of templates. TML[8] introduced in TRiX[9] framework [7] is a nearest work on this subject.

The TRiX is intended to separate the presentation logic from the application logic. TML combined with the notion of variables as URLs provides a language for the construction of documents from templates on-the-fly.

Although the TML contains analogous solutions, it is not suitable for our purposes. The main purpose of MTL is to describe task templates. Therefore, MTL should fix a structure of a document and should hold the document with the parameter list consistent. TML in contrast can refer another documents without checking their existence. MTL is a stand-alone language; TML must be used together with a target language. Parameters in TML can be defined also in another documents; MTL defines all parameters used in a document in a separate section within the same document. This leads to further differences. TML uses the *href*-syntax to refer the parameters, MTL defines the references on parameters as attributes of IDREF type that increases the consistence of documents and provides a possibility of using a simple XSLT transformation to obtain the final document. The complete parameter list can be easily extracted from any MTL document, TML uses the external definitions of the parameters and that makes the usage of XSLT transformation impossible.

Usage of templates for the creation of HTML forms is discussed in [6], where templates are used only to fix a structure of documents series and to produce the corresponding HTML forms. The generated XML document is used further as a storage unit. The similar approach is used here to define a source for a HTML form

[8] Template Markup Language
[9] Template Resolution in XML/HTML

for the parameter list. In contrary to [6], the filled values are stored separately from template for further usage and the requested document is generated on-the-fly from the corresponding template and the parameter set.

6. Discussions

The proposed approach for the construction of templates significantly simplifies the implementation of template-based systems. As shown above another language should be used for the description of templates, than that for the description of tasks. Therefore, transformation of the template after substitution of the parameter values is needed to obtain the task description. That can be achieved by diverse methods. The choice of the method influences the system implementation. XSLT transformation is chosen in order to get a possibility to use the *standard* JDOM components, which reduces the amount of code significantly. Only a simple object adapter has to be implemented for handling of the requests peculiar to the system.

The proposed system architecture is extensible, because a variety of task plans can be added to and removed from the system at run time. It is also robust, because the new kinds of mobile devices can be used without modification of the system.

To achieve some goals, the XSLT transformation of the template can be used for building a HTML form for the parameter input. Using the special XSLT transformation for each template achieves a clearer presentation of the parameter list for caregivers and increases a flexibility of the system. On the other side, the test of the transmitted parameters needs a parameter list anyway. Therefore, the list of parameters should be also extracted. That makes this XSLT transformation redundant. Furthermore, such transformation cannot be developed by the ordinary template designer; therefore, it has not been implemented.

7. Summary

The described system provides the simple management with remote access to the template set. The proposed template structure provides correctness of the produced tasks and consistence of task plans and parameter collections. Application of the XSLT transformation for creation of documents in a target language from the documents in template language provides a possibility to present the templates as the JDOM of *standard* components. This essentially simplifies implementation of the system. A separate storage of the parameters and the templates gives an essential gain in the disk space and the network loading.

Acknowledgements. The authors would like to thank Oleg Voinikonis and Venera Khoromskaia for support of this work.

References

1. Schulze H., Irmscher K., Mobtel - A Mobile Distributed Telemedical System, Proceedings of USM 2000 - Trends in Distributed Systems: Towards a Universal Service Market Third International IFIP/GI Conference, Munich, Springer, 2000, pp. 176
2. Schulze, H., Voinikonis, A., Hoffmann, P., Irmscher, K., Modeling a Mobile Memory Aid System, Proceedings zur 13. ITG/GI-Fachtagung “Kommunikation in Verteilten Systemen” (KiVS2003). 25.-28.02.2003, Universität Leipzig. Springer-Verlag, Reihe Informatik aktuell. Berlin, Heidelberg, New York, 2003. S. 143-153
3. Thöne-Otto, A., v.Cramon, Y., Irmscher, K., Schulze, H., MEMOS ? Mobile Extensible Memory System: Elektronische Gedächtnishilfe für hirngeschädigte Patienten, Deutsches Ärzteblatt 98, Heft 11 vom 16.03.01, Seite 14
4. Ahmed, K. et. al., Professional Java XML, Wrox Press, 2001
5. Avedal, K. et. al., JSP Professional, Wrox Press, 2001
6. Schweiger, R. et. al., Using XML for flexible data entry in healthcare, example use for pathology, XML Europe Conference Proceedings 2000, 357-362.
7. Kristensen, A., Template resolution in XML/HTML, Computer Networks and ISDN Systems, v.30 n.1-7, p.239-249, April 1, 1998
8. Gamma, E. et al., Design Patterns - Elements of Reusable Object-Oriented Software, Addison Wesley, 1995
9. Website W3C, http://www.w3c.org
10. Website "The Apache XML Project", http://xml.apache.org/xalan-j
11. Website "The JDOM Project", http://www.jdom.org

Aggregation Transformation of XML Schemas to Object-Relational Databases

Nathalia Devina Widjaya[1], David Taniar[1], and J. Wenny Rahayu[2]

[1] Monash University, School of Business Systems, Vic 3800, Australia
{Nathalia.Widjaya, David.Taniar}@infotech.monash.edu.au
[2] La Trobe University, Department of Computer Science and Engineering,
wenny@cs.latrobe.edu.au

Abstract. As XML has become an emerging standard for information exchange on the World Wide Web, it has gained attention in database communities to extract information from XML sees as a database model. Recently, many researchers have addressed mappings from XML structures onto database structures. In this paper, we present the way to transform the XML encoded format, which can be treated as a logical model, to the ORDB format. Firstly, the paper discusses the modelling of XML and why we need the transformation. Then, a number of transformation steps from the XML schema to the ORDB, with the emphasis on the transformations of aggregation relationships are presented. Two perspectives regarding this conceptual relationship (existence dependent aggregation which consists of homogeneous and ordered composition and independent aggregation) and their transformations are mainly discussed.

1. Introduction

The popularity of XML (eXtensible Markup Language) is growing and XML schema is being widely used to describe data. XML has emerged and is gradually accepted as the standard for describing data and interchanging data between various systems and databases on the Internet [1]. At the moment, XML offers the Document Type Definition (DTD) as formalism for defining the syntax and structure of XML documents. Then XML Schema definition language as a substitution of DTD provides more rich facilities for defining and constraining the content of XML documents [10].

With the wide acceptance of the Object Oriented conceptual models, more and more systems are initially modeled and being expressed with OO notation. This situation suggests the necessity to integrate the OO conceptual models and XML. The used of XML and XML Schemas to restore the data is no longer effective and efficient to store a lot of data. Because of that, there is a need to put the data from XML into the database without eliminating the object-oriented features that exist in XML Schemas.

The goal of this work is to present a coherent way to transform the XML schema into ORDB (Object-Relational Databases) using Oracle 9i features models. The

T. Bhm e, G. Heyer, H. Unger (Eds.): IICS 2003, LNCS 2877, pp. 251-262, 2003.

emphasis of this paper is only on the transformation of aggregation relationship from XML schema in order to help people conveniently and automatically generate Oracle database. This transformation is important so that all the tables that are created using XML schema can be transformed to the object relational databases using Oracle format and features.

The work presented in this paper is actually part of a larger research project on Transformation from XML Schema to Object-Relational Databases. This project consists of three stages: (i) transformation association relationship from XML Schema to Object-Relational Database, (ii) transformation inheritance relationship from XML Schema to Object Relational Database and (iii) transformation aggregation relationship from XML Schema to Object Relational Database. The research results from the first and second stage have been reported in [8] and [9]. In this paper, we focus on the final stage of the project.

The content of the article will consist of the introduction about XML schema and ORDB. We will explain the relationship that can exist in the XML schema and OO (Object-Oriented) concepts. The remainder of the paper is organised as follows. Section 2 discusses about the overview over OO concepts and OO in XML schemas. Then, we review some closely related work. Section 3 presents several generic-transforming rules from XML schema to ORDB with the emphasis on the transformation of aggregation relationship. We discuss the transformation steps and give example for each of them. Section 4 concludes the paper and further work that can be done.

2. Background and Related Work

2.1 Object-Oriented: A Brief Overview

In 1970 there was only Relational Database Management System (RDBMS) [2]. Traditional RDBMSs perform well only when working on numeric data and characters stored in tables, what are often called "simple data types." [8] Then, ORDBMS (Object-Relational Database Management System) comes later to improve RDBMSs performance. Basically, the ORDBMS is the RDBMS with the object-oriented features. ORDBMS becomes popular because of the failure of ODBMSs, which has limitations that can prevent it from taking on enterprise-wide tasks. Therefore, by storing objects in the object side of the ORDBMS but keeping the simpler data in the relational side, users may approach the best of both worlds. For the foreseeable future, however, most businesses data will continue to be stored in object relational database system.

Since ORDBMS has object-oriented features, we will discuss briefly about Object-Oriented Conceptual Model (OOCM). OOCM encapsulates the structural/static as well as behavioural/dynamic aspects of objects. The static aspects consist of the classes/objects and the relationship between them, namely inheritance, association and aggregation. The dynamic aspect of the OOCM consists of generic methods and user-defined methods. We only discuss about the static aspects since this is the topic that is relevant with this paper. Static aspects in OOCM create objects and classes that also include decisions regarding their attributes. Furthermore, they

also concern on the relationship between objects. The basic segment of the object-oriented system is an object. An object is a data abstraction that is defined by an object name as a unique identifier, valued attributes (instance variables), which give a state to the object, and methods, or routines that access the state of the object.

In XML Schemas, there are static aspects from object-oriented conceptual model that we can find. The aggregation is one of OOCM features that we will discuss in this paper. Aggregation is a part-of relationship (refer to figure 1), in which a composite object (whole) consists of other component objects (parts).

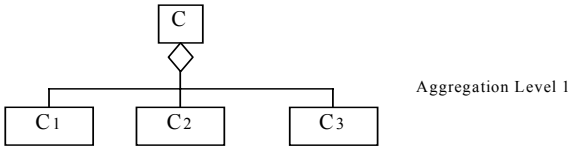


Figure 1. A one-levelled aggregation relationship rooted at *C*

There are four types of aggregation relationship according to Dillon and Tan (1993) such as: sharable dependent, sharable independent, non-sharable dependent and non-sharable independent composition. In this paper, we only focus on existence dependent and existence independent composition. It is vital to remember that in UML the term composition refers to exclusive and dependent aggregation. However, we use composition interchangeably with aggregation and use qualifications to distinguish between the different categories. Furthermore, we also look at two more types of aggregation relationship, i.e. ordered composition and homogenous/heterogeneous composition.

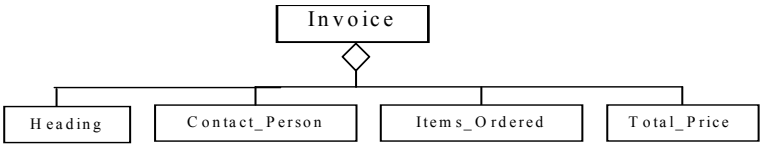


Figure 2. Class diagram showing existence dependent composition

Existence-dependent aggregation means there is a dependency between the whole object and its part object. In the existence-dependent, the deletion of the whole object will cause the deletion of that object and its elements (Refer to figure 2). While in existence-independent aggregation, there is no dependency between the whole object and its part object, therefore the deletion of the whole object will not cause the deletion of its element (Refer to figure 3).

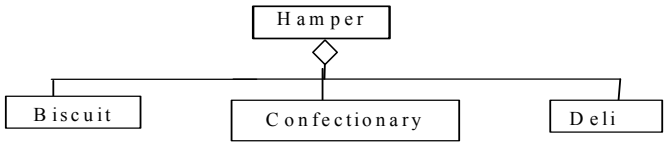


Figure 3. An existence independent composition example

We called the aggregation as ordered composition if a whole object composed of different part objects in particular order. In other words, the order of occurrence of the part objects in the composition is vital to the model.

The other types of composition are heterogeneous and homogeneous. Basically all categories of composition are heterogeneous since one whole object may consist of several different types of part objects. On the other hand, Homogenous composition means that one whole object consists of part objects that are of the same type (Refer to figure 4). The notation that is used to show the aggregation relationship is the diamond arrow. The class with the diamond next to it refers to the super class.

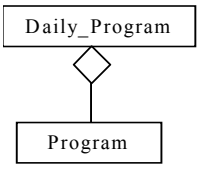


Figure 4. Class diagram showing homogeneous composition example

2.2 Related Work

Most existing work has focused on a methodology that has been designed to map a relational database to an XML database for database interoperability. The schema translation procedure is provided with an EER (Extended Entity Relationship) model mapped into XML schema [3].

There are many works that explain about the mapping from relational databases to XML. Some of them still use DTD [10,11] and some of them use XML schema [5]. Since XML is rapidly emerging as the dominant standard for exchanging data on the WWW, the previous work already discussed about mapping referential integrity constraints from Relational Database to XML, semantic data modeling using XML schemas and enhancing structural mapping for XML and ORDB.

In addition, the study about the use of new scalar and aggregate functions in SQL for constructing complex XML documents directly in the relational engine has been done [10].

Relational and object-relational database systems are a well-understood technique for managing and querying such large sets of structured data. In [5], the writers wrote about how a relevant subset of XML documents and their implied structure can be mapped onto database structures. They suggest mapping DTDs onto object-relational databases schemas and to overcome the typical problems (large schemas), they suggested an algorithm for determining an optimal hybrid database schema.

The way to model XML and to transform the OO conceptual models into XML Schema have been discussed in [10]. The writers choose the OO conceptual model because of its expressive power for developing a combined data model. They come out with several generic-transforming rules from the OO conceptual model to XML schema, with the emphasis on the transformations of generalization and aggregation relationships. The XML Schema code that is presented below, in this paper, is

adopted from the existing work that is done previously. In addition, our paper is done to improve what has been done in [10].

The work reported in this paper is distinguishes from this work in the following aspects. First, we focus the transformation from XML schema to ORDB. Second, our transformation target using OO features in Oracle 9i not just the general OO features. The similarity is we take aggregation relationships into consideration (existence dependent and independent aggregation).

3. Transformation from XML Schema to ORDB: The Proposed Methodology

In the following, we use XML Schema and Oracles 9i to interpret the aggregation relationship in OO conceptual models. We discuss the transformation or mapping from XML Schema to ORDB. In this section, we also validate the following documents against the schema. In addition, we also give the example how to insert the data to the table after creating the table in Oracle 9i. Table 1 shows the expressions that are used for data types from XML schema and ORDB in this article. Both of them have the same meaning, but using different phrase.

XML Schema Data types	ORDB Data Types
String	Varchar2
Decimal	Decimal(l,d); l = length, d = decimal

Table 1. Data Types Mapping

3.1 Existence Dependent (Ordered Composition)

The following steps generate a transformation from XML Schema to Object Oriented Relational Database in Oracles 9i for ordered existence dependent aggregation relationship.

(i) For an aggregation relationship rooted at a composite class C, an element named C with a complex Type Ctype in XML schema (<xsd:element name = "C" type= "Ctype">) can be transformed by creating a cluster named C_cluster in ORDB. Then, write the type of class C attributes (such as C_id). Usually it is in the *varchar2* format and the user will enter the length for it. Refer to table 1.0 to transform the data types from XML schema to ORDB.

XML Schema:

```
<xsd:element name="Invoice" type = "InvoiceType"/>
    <xsd:complexType name ="InvoiceType">
```

ORDB:

```
Create Cluster Invoice_Cluster
    (invoice_Id varchar2 (10))
```

(ii) Create a table for composite class *C* and the type of its attribute, which is exactly the same as `C_cluster` above and has *Not Null* besides the `C_id` which means table invoice must have an id. Then, create a primary key for this table, which is usually `C_id`. Next, create a cluster as *C* table attributes and the type will be `C_cluster (C_id)`.

ORDB:

```
Create Table Invoice
    (invoice_id        varchar2(10) Not Null,
     Primary Key       (invoice_id))
Cluster Invoice_Cluster (invoice_id);
```

(iii) Based on each sub-element named C1 within the *complexType Ctype* in the XML Schema (`<xsd:element name = "C1" type= "...">`), we need to create another table for each of sub-element. Its attributes will consist of `C_id`, `C1_id` and other attributes that are relevant with `C1`. `C_id` and `C1_id` will be the primary key and the foreign key will be *C_id references C (C_id)*. Next, create a cluster and its type that should be the same with the cluster that is created before, `C_cluster (C_id)`.

XML Schema:

```
<xsd:sequence>
    <xsd: element name = "Heading" type = "xsd:string"/>
```

ORDB:

```
Create Table Heading
    (invoice_id        varchar2 (10)  Not Null,
     heading_id        varchar2 (10)  Not Null,
     Headingvarchar2 (30),
     Primary Key (invoice_id, heading_id),
     Foreign Key (invoice_id) References Invoice (invoice_id))
Cluster Invoice_Cluster (invoice_id);
```

(iv) Create index for `C_cluster_index` on cluster `C_cluster`

ORDB:

```
Create Index Invoice_Cluster_Index
    On Cluster Invoice_Cluster;
```

Below is the full example of the transformation from ordered existence dependent aggregation relationship from XML Schema to ORDB.

XML Schema for ordered existence dependent aggregation relationship:

```
<xsd:element name= "Invoice" type = "InvoiceType"/>
<xsd:complexType name = "InvoiceType">
    <xsd:sequence>
        <xsd:element name = "Heading" type = "xsd:string"/>
        <xsd:element name = "Contact_Person" type =
        "ContactPersonType"/>
```

```
            <xsd:element name = ″Items_Ordered″ type = ″xsd:string″/>
            <xsd:element name = ″Total_Price″ type = ″xsd:decimal″>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name = ″ContactPersonType″>
        <xsd:sequence>
            <xsd:element name = ″Name″ type = ″xsd:string″/>
            <xsd:element name = ″Address″ type = ″xsd:string″/>
            <xsd:element name = ″PhoneNo″ type = ″xsd:decimal″/>
        </xsd:sequence>
    </xsd:complexType>
```

ORDB for ordered existence dependent aggregation relationship:

```
Create Cluster Invoice_Cluster
    (invoice_id         varchar2 (10));
Create Table Invoice
    (invoice_id         varchar2 (10) Not Null,
     Primary Key        (invoice_id))
     Cluster            invoice_cluster (invoice_id);
Create Table Heading
    (invoice_id         varchar2 (10) Not Null,
     heading_id         varchar2 (10) Not Null,
     Primary Key        (invoice_id, heading_id),
     Foreign Key        (invoice_id) References Invoice (invoice_id))
Cluster                 invoice_cluster (invoice_id);
Create Table Contact_Person
    (contact_person_id          varchar2 (10) Not Null,
     invoice_id         varchar2 (10) Not Null,
     name               varchar2 (40),
     address            varchar2 (40),
     phone_no           number
     Primary Key        (invoice_id, contact_person_id),
     Foreign Key        (invoice_id) References Invoice (invoice_id))
Cluster                 invoice_cluster (invoice_id);
Create Table Item_Ordered
    (invoice_id         varchar2 (10) Not Null,
     item_ordered_Id    varchar2 (10) Not Null,
     Primary Key        (invoice_id, item_ordered_id),
     Foreign Key        (invoice_id) References Invoice (invoice_id))
Cluster                 invoice_cluster (invoice_id);
Create Table Total_Price
    (invoice_id         varchar2 (10) Not Null,
     total_price_ID     varchar2 (10) Not Null,
     Primary Key        (invoice_id, total_price_id),
     Foreign Key        (invoice_id) References Invoice (invoice_id)
Cluster                 invoice_cluster (invoice_id)
Create Index Invoice_Cluster_Index On Cluster Invoice_Cluster
```

3.2 Existence Dependent (Homogeneous Composition)

Figure 3 shows the homogenous existence dependent aggregation relationship. We can generate a transformation for existence dependent homogeneous aggregation relationship from XML Schema to ORDB in Oracle 9i as follows.

(i) Each sub-element named `C1` with a complex Type `Ctype` in XML schema (`<xsd:element name = "C1" type= "…." MinOccurs= "…" maxOccurs= "…">`) need to be created as an object named `C1`. Then, write the type of `C1` attributes (such as `C1_id`), usually it is in the `varchar2` format, and the user will enter the length for it.

XML Schema:

```
<xsd:element name = "Program" type = "xsd:string"
    minOccurs="1" maxOccurs="unbounded"/>
```

The *maxOccurs* explains the maximum number of *Program* in *Daily_Program.* This may be a positive integer value or the word unbounded to specify there is no maximum number of occurrences. The *minOccurs* shows the minimum number of times an element may appear. It is always less than or equal to the default value of *maxOccurs*, i.e. it is 0 or 1. Similarly, if we only specify a value for the *maxOccurs* attribute, it must be greater than or equal to the default value of *minOccurs*, i.e. 1 or more.

ORDB:

```
Create Or Replace Type Program As Object
    (Program_id       varchar2 (10));
```

(ii) Create a table for composite class `C1` (as a table of the object above)

ORDB:

```
Create Or Replace Type Program_Table As Table Of Program
```

(iii) For a homogeneous existence dependent aggregation relationship rooted at a composite class `C`, an element named C within the `complexType Ctype` in the XML Schema (`<xsd:element name = "C" type= "Ctype">`) need to created as an object named C. Its attributes will consist of `C_id`, and other attributes that are relevant to it. `C_id` will be the primary key. Next, nested this table and stored it as the table that is created before.

XML Schema:

```
<xsd:element name = "Daily_Program" type = "DailyProgramType"/>
    <xsd:complexType name = "DailyProgramType">
```

ORDB:

```
Create Table Daily_Program
    (daily_program_idvarchar2(10) Not Null,
```

```
    program_name            Program_Table,
    Primary Key (daily_program_id))
    Nested Table program_name Store As Program_Table;
```

Below is the full example of transformation from XML schema homogeneous existence dependent aggregation to ORDB.

XML Schema for homogeneous existence dependent aggregation

```
<xsd:element name; ″Daily_Program″ type = ″DailyProgramType″/>
<xsd:complexType name = ″DailyProgramType″>
    <xsd:element name = ″Program″ type = ″xsd:string″ minOccurs = ″1″
        maxOccurs= ″unbounded″/>
```

ORDB for homogeneous existence dependent aggregation

```
Create Or Replace Type Program As Object
    (program_id             varchar2 (10));
Create Or Replace Type Program_Table As Table Of Program
Create Table Daily_Program
    (daily_program_idvarchar2(10) Not Null,
     program_name           Program_Table,
     Primary Key (daily_program_id))
     Nested Table program_name Store As Program_Table;
```

3.3 Existence Independent

The following steps generate a transformation from XML Schema to Object Oriented Relational Database in Oracle 9i for existence independent aggregation relationship.

(i) For an aggregation relationship rooted at a composite class C, an element named C with a complex Type Ctype in XML Schema (<xsd:element name = C type = Ctype>) can be transformed by creating a table named C in ORDB. Then, write the type of class C attributes (such as C_id) based on the attribute for that Ctype in the XML schemas.

XML Schemas:

```
<xs:element name="Hamper" type ="HamperType">
……
<xs:element name = "HamperType">
<xs:complexType>
   <sequence>
   <xs:element name = "hamper_id" type = "xs:string"/>
     <xs:element name = "hamper_price" type ="xs:decimal"/>
     </sequence>
</xs:complexType>
</xs:element>
```

ORDB:

```
Create Table Hamper
(hamper_id          varchar2(3)     Not Null,
```

```
hamper_price        Number,
Primary Key (hamper_id));
```

ii) Create tables for each element under choice. The element reference under choice means that it refers to the details below where the element name equals to the element reference.

XML Schema:

```
<xs:complexType>
<xs:choice>
   <xs:element ref = "Biscuit"/>
   <xs:element ref = "Confectionary"/>
   <xs:element ref = "Deli"/>
</xs:choice>
</xs:complexType>

<xs:element name = "Biscuit">
<xs:complexType>
<sequence>
   <xs:element name = "biscuit_id" type =          "xs:string"/>
   <xs:element name = "biscuit_name" type ="xs:string"/>
   <xs:element name = "biscuit_price" type = "xs:decimal"/>

</sequence>
</xs:complexType>
</xs:element>
```

ORDB:

```
Create Table Biscuit
(biscuit_id             varchar2(3)      Not Null,
 biscuit_name           varchar2(20),
 biscuit_price Number,
 Primary Key (biscuit_id));
```

iii) Create the last table that we called as an aggregate table which will link the composite class with the sub-classes. Then, create the attributes for this class which includes the id for the composite class, part_id and part_type. Lastly, create a primary key and a foreign key.

ORDB:

```
Create Table Aggregate
(hamper_id              varchar2(3)      Not Null,
 part_id                varchar2(3)      Not Null,
 part_type              varchar2(20)     Check
(part_type In ('biscuit', 'confectionery', 'deli')),
 Primary Key (hamper_id, part_id),
 Foreign Key (hamper_id) References hamper (hamper_id));
```

Below is the mapping of ORDB for existence independent from the XML Schema existence independent aggregation.

ORDB for existence independent aggregation

```
Create Table Hamper
    (hamper_id      varchar2(3)     Not Null,
     hamper_price   Number,
     Primary Key (h_id));
Create Table Biscuit
    (biscuit_id     varchar2(3)     Not Null,
     biscuit_name   varchar2(20),
     biscuit_price  Number,
     Primary Key (biscuit_id));
Create Table Confectionery
    (confectionery_id       varchar2(3)
     confectionery_name     varchar2(20),
     confectionary_price    Number,
     Primary Key (confectionary_id));
Create Table Deli
    (deli_id        varchar2(3)     Not Null,
     deli_name      varchar2(20),
     deli_price     Number,
     Primary Key (deli_id));
Create Table Aggregate
    (hamper_id      varchar2(3)     Not Null,
     part_id        varchar2(3)     Not Null,
     part_type      varchar2(20)    Check
    (part_type In('biscuit', 'confectionery', 'deli')),
     Primary Key (hamper_id, part_id),
     Foreign Key (hamper_id) References hamper (hamper_id));
```

4. Conclusion and Future Work

In this paper, we have investigated the transformation from XML schema to the ORDB by using Oracle 9*i*. We emphasis the transformation of *aggregation relationship* to help people easily understand the basic object conceptual mapping that we proposed. This transformation is important because people always eliminate the object-oriented conceptual features when they transform XML schema to the database.

Our research gives better solution in transformation XML Schema into ORDB rather than the XML features that Oracle 9i have. Oracle 9i can only convert all the data or query result in XML format but it does not deal with the type of database that is used, such as relational database or object oriented database, like we do. This transformation can be applied on any XML documents that use XML Schema.

Our future work is being planned to investigate more transformation from XML schema to ORDB for other XML Schema features that has not been discussed in this paper. In addition, further research should be done to create a query from XML schema to get the data from the Oracle 9i databases.

References

1. Bray, T., Paoli, J., Sperberg-McQueen, C.M (eds)., Extensible Markup Language (XML) 1.0. W3C (1998) http://www.w3c.org/TR/REC-xml
2. Dillon T and Tan PL, *Object Oriented Conceptual Models*. Prentice Hall, 1993.
3. Fong J., Pang F., and Bloor, C., "Converting Relational Database into XML Document", *Proc. 12th Intl. Workshop on Database and Expert Systems Appl*, 2001, pp. 61-65, 2001.
4. Klettke M. and Meyer H., "XML and Ob ject Relational Database System", *Lecture Notes in Computer Science*, vol 1997, Springer-Verlag, pp.151-170, 2001.
5. Mani M., Lee D. and Muntz R., "Seman tic Data Modelling Using XML Schemas", *Lecture Notes in Computer Science*, vol 2224, Springer-Verlag pp.149-163, 2001.
6. Shanmugasundaram, J. et al., Efficiently pub lishing relational data as XML documents", *The VLDB Journal*, vol 10, pp. 133-154
7. Stonebraker, M., and Moore, D., "*Object-relational DBMSs:the next great wave*", Mogran Kaufmann Publishers. San Francisco, 1996.
8. Widjaya, N.D., Taniar, D., Rahayu, J.W., and Pardede, E., "Association Relationship Transformation of XML Schemas to Object-Relational Databases", *Proceedings of the 4th International Conference on Information Integration and Web-based Applications and Services (IIWAS'2002*), pp. 135-142, 2002
9. Widjaya, N.D., Taniar, D., and Rahayu, J.W, "Inheritance Relationship Transformation of XML Schemas to Object-Relational Databases", *Proc. of the 4th Intl. Conference on Intelligent Data Engineering & Automated Learning (IDEAL'2003),* 2003.
10. Xiao R., Dillon T., Chang E., and Feng L., Modelling and Transformation of Object-Oriented Conceptual Models into XML Schema", *Lecture Notes in Computer Science*, vol 2113, Springer-Verlag, pp.795-804, 2001.
11. Yang X. and Wang G., Efficiently Mapping Referential Integrity Constraints from Relational Databases to XML. *LNCS*, vol 2151, Springer-Verlag, pp.338-351, 2001.
12. Yang X. and Wang G., Mapping Referentia l Integrity Constraints from Relational Databases to XML. *LNCS*, vol 2118, Springer-Verlag, pp.329-340, 2001.

Author Index

Bengsch, A., 1
Book, M., 114
Bordag, S., 10
Brinkmeier, M., 20
Bui, A., 36
Bui, M., 36

Eichler, G., 48

Fogaras, D., 65
Fouchal, H., 201
Fuhrmann, T., 80

Goel, S., 93, 156
Grossman, J.P., 104
Gruhn, V., 114, 134

Heutelbeck, D., 144
Heyer, G., 10
Hülder, M., 134

Ijioui, R., 134
Irmscher, K., 239

Kakaiya, M., 156
Kopp, H., 1

Maicher, L., 168

Nützel, J., 180

Petrovskiy, M., 189
Petry, A., 1

Quasthoff, U., 10

Rahayu, J.W., 251
Räth, R., 144
Rollet, A., 201
Roth, J., 216

Schmidt, F., 229
Schöpe, L., 134
Sharda, H., 93, 156
Sohier, D., 36

Taniar, D., 93, 156, 251
Tavangarian, D., 1

Unger, C., 144

Voinikonis, A., 239

Widjaya, N.D., 251
Wolff, C., 229