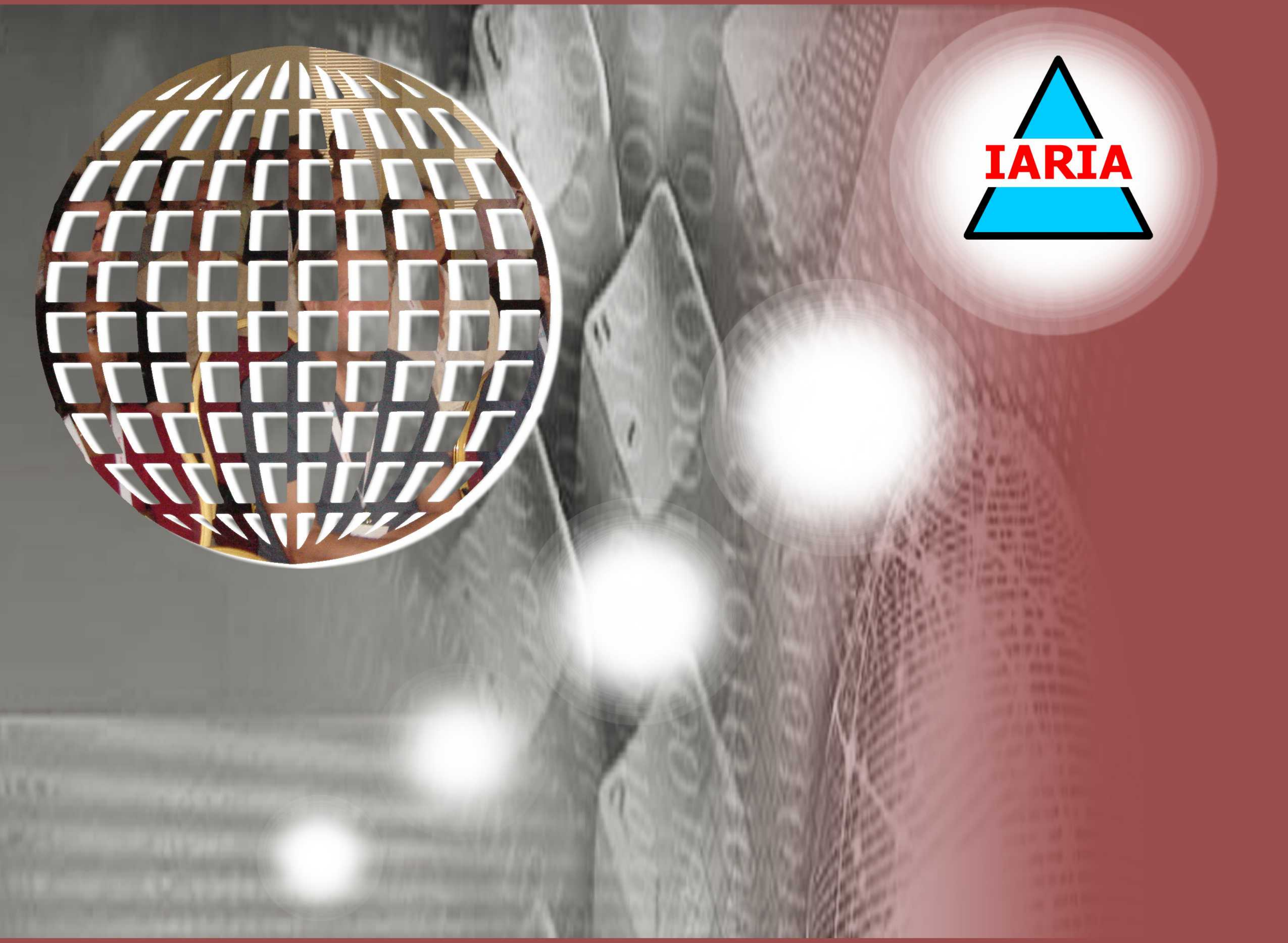
International Journal on Advances in Telecommunications



2009 vol. 2 nr. 1

The *International Journal On Advances in Telecommunications* is Published by IARIA.
ISSN: 1942-2601
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal On Advances in Telecommunications
Volume 2, Number 1, 2009

- Tulin Atmaca, IT/Telecom&Management SudParis, France
- Claus Bauer, Dolby Systems, USA
- Claude Chaudet, ENST, France
- Gerard Damm, Alcatel-Lucent, France
- Michael Grottke, Universitat Erlangen-Nurnberg, Germany
- Yuri Ivanov, Movidia Ltd. – Dublin, Ireland
- Ousmane Kone, UPPA - University of Bordeaux, France
- Wen-hsing Lai, National Kaohsiung First University of Science and Technology, Taiwan
- Pascal Lorenz, University of Haute Alsace, France
- Jan Lucenius, Helsinki University of Technology, Finland
- Dario Maggiorini, University of Milano, Italy
- Pubudu Pathirana, Deakin University, Australia
- Mei-Ling Shyu, University of Miami, USA

Communication Theory, QoS and Reliability

- Eugen Borcoci, University "Politehnica" of Bucharest (UPB), Romania
- Piotr Cholda, AGH University of Science and Technology - Krakow, Poland
- Michel Diaz, LAAS, France
- Ivan Gojmerac, Telecommunications Research Center Vienna (FTW), Austria
- Patrick Gratz, University of Luxembourg, Luxembourg
- Axel Kupper, Ludwig Maximilians University Munich, Germany
- Michael Menth, University of Wuerzburg, Germany
- Gianluca Reali, University of Perugia, Italy
- Joel Rodriques, University of Beira Interior, Portugal
- Zary Segall, University of Maryland, USA

Wireless and Mobile Communications

- Tommi Aihkisalo, VTT Technical Research Center of Finland - Oulu, Finland
- Zhiquan Bai, Shandong University - Jinan, P. R. China
- David Boyle, University of Limerick, Ireland
- Bezalel Gavish, Southern Methodist University - Dallas, USA
- Xiang Gui, Massey University-Palmerston North, New Zealand
- David Lozano, Telefonica Investigacion y Desarrollo (R&D), Spain
- D. Manivannan (Mani), University of Kentucky - Lexington, USA
- Himanshukumar Soni, G H Patel College of Engineering & Technology, India
- Radu Stoleru, Texas A&M University, USA
- Jose Villalon, University of Castilla La Mancha, Spain
- Natalija Vlajic, York University, Canada
- Xinbing Wang, Shanghai Jiaotong University, China
- Ossama Younis, Telcordia Technologies, USA

Systems and Network Communications

- Fernando Boronat, Integrated Management Coastal Research Institute, Spain
- Anne-Marie Bosneag, Ericsson Ireland Research Centre, Ireland
- Huaqun Guo, Institute for Infocomm Research, A*STAR, Singapore
- Jong-Hyouk Lee, Sungkyunkwan University, Korea
- Elizabeth I. Leonard, Naval Research Laboratory – Washington DC, USA
- Sjouke Mauw, University of Luxembourg, Luxembourg
- Reijo Savola, VTT, Finland

Multimedia

- Dumitru Dan Burdescu, University of Craiova, Romania
- Noel Crespi, Institut TELECOM SudParis-Evry, France
- Mislav Grgic, University of Zagreb, Croatia
- Atsushi Koike, KDDI R&D Labs, Japan
- Polychronis Koutsakis, McMaster University, Canada
- Chung-Sheng Li, IBM Thomas J. Watson Research Center, USA
- Artur R. Lugmayr, Tampere University of Technology, Finland
- Parag S. Mogre, Technische Universitat Darmstadt, Germany
- Chong Wah Ngo, University of Hong Kong, Hong Kong
- Justin Zhan, Carnegie Mellon University, USA
- Yu Zheng, Microsoft Research Asia - Beijing, China

Space Communications

- Emmanuel Chaput, IRIT-CNRS, France
- Alban Duverdier, CNES (French Space Agency) Paris, France
- Istvan Frigyes, Budapest University of Technology and Economics, Hungary
- Michael Hadjitheodosiou ITT AES & University of Maryland, USA
- Mark A Johnson, The Aerospace Corporation, USA
- Massimiliano Laddomada, Texas A&M University-Texarkana, USA
- Haibin Liu, Aerospace Engineering Consultation Center-Beijing, China
- Elena-Simona Lohan, Tampere University of Technology, Finland
- Gerard Parr, University of Ulster-Coleraine, UK
- Cathryn Peoples, University of Ulster-Coleraine, UK
- Michael Sauer, Corning Incorporated/Corning R&D division, USA

International Journal On Advances in Telecommunications
Volume 2, Number 1, 2009

Foreword

The first 2009 number of the International Journal On Advances in Telecommunications compiles a set of papers with major enhancements based on previously awarded publications. It brings together a set of articles that share a common link to telecommunications. For this issue, five contributions have been selected.

The first article by Rong Zhao et al. presents an enhancement of optical broadband networks design with biconnectivity. A series of strategic decisions are shown in order to efficiently generate new networks making use of existing infrastructure.

The second article touches on the subject of overlay topology. Serban Obreja and Eugen Borcoci address the issue of finding inter-domain routes under QoS constraints.

An online mobility simulation is presented by Anders Nickelsen and Hans-Peter Schwefel in the third article. Emulating wireless multi-hop technologies is required as a real setup is subject to a multitude of variables which can contaminate end results.

As proposed by Kazumasa Takami et al., the user's state of mind can be evaluated from emoticons used in mobile phone emails. In turn, this can create a profile which benefits the selection of customized content delivery. As a self-assessment, the conclusions based on emoticons are used to select a relevant piece of music, which arguable is also tied to the emotional state of the user.

Last but not least, Jaroslav Kral and Michal Zemlicka present a system of development patterns catered to service-oriented software (SOA). As there are several approaches to SOA, the appropriate one must be selected on a case by case basis.

We hope that the contents of this journal will add to your understanding of telecommunications, and that you will be inspired to contribute to IARIA's conferences that include topics relevant to this journal.

Tulin Atmaca, Editor-in-Chief
Petre Dini, IARIA Advisory Committees Board Chair

International Journal On Advances in Telecommunications
Volume 2, Number 1, 2009

CONTENTS

Enhanced Survivable Topology Redesign of Optical Broadband Networks with Biconnectivity

Rong Zhao
Detecon International GmbH
53227 Bonn, Germany
rong.zhao@detecon.com

Christian Minge
Dimension Data Germany AG&Co.KG
61440 Oberursel, Germany
christian.minge@eu.didata.com

Mathias Schweigel
Detecon International GmbH
01187 Dresden, Germany
mathias.schweigel@detecon.com

Abstract

This paper presents an efficient strategy for the optimal network redesign with a biconnectivity-oriented topology (both edge and vertex biconnected). It helps re-designing existing networks or generating new networks considering practically relevant constraints (such as leased lines with long running contracts to remain, maximum number of ports per device) typically found in these phases. The proposed strategy is composed of Reduction, Augmentation, and Fine-Tuning Reduction. Empirical tests using several IP network topologies showed the robustness and applicability of the method. Its application to other optical backbone (or access) network redesign problems is possible.

Keywords - Optical Networks, Redesign, Broadband Networks, Survivability, Biconnectivity

1 Introduction

When designing/redesigning telecommunications networks, we have to face up to two opposing objectives: high survivability and low costs. The former leads to a fully meshed topology (expensive and highly redundant), whereas the latter results in a minimum spanning tree topology (cheap with no redundancy). The main idea of this approach is to redesign a network by retaining essential network links (reduce network migration costs), improving their utilization (but not to overload), and ensuring network survivability. If necessary, only a few links are allowed to be added into networks subject to minimum costs. Simultaneously, some constraints are considered, such as routing, maximal hop number, and node degree. A part of this work was presented at AICT2008 conference [1].

Some prior approaches addressed the topological design for backbone networks by [2][3][4][5][6] and access networks by [7][8][9][10][11]. Generally, network structures are illustrated by hierarchical star-star, tree-star, or mesh-star topology. Most of these optimization problems are NP-hard [12]. Due to the complexity of the tasks different methodologies were investigated, such as the linear programming, Simulated Annealing(SA), Tabu search(TS), Genetic Algorithms(GA). A detailed formulation for general network design problems with connectivity requirements was introduced in [13][14]. Furthermore, the network redesign is discussed in [15][16].

IP networks reportedly suffer from node failures as frequently as from link failures [17]. To avoid service degradation (and related penalties) the design of reliable communication networks is a significant problem for network providers. The general planning problem is finding the best positions of components and their links subject to minimal costs and a high reliability [18]. An essential summary of approaches to different reliability problems, such as constrained reliability measures and reliability optimization, is provided in [19]. A general definition of *reliability* of network components is the probability that the network is functioning [20]. In comparison with reliability, survivability is to describe the resilient ability of networks when one or more network components fails. More precisely, the survivability analysis is to make a conservative assumption of failures and study how to prevent them. Typical network survivability techniques are based on well designed networks and network restoration [21][22], e.g. link restoration or link/node protection.

A possible solution for the above mentioned design problem is a two-connected topology that can be described by means of graph theory. Graph connectivity properties are meaningful for transport network designs. In order to survive all single edge failures, a graph must be at least *two-edge connected*. Furthermore, it has been shown that every *two-vertex connected* graph is also two-edge connected, while the reverse is not valid [23]. In this work a redesigned network has to be at least *two-vertex connected*. In the following we characterize the *two-vertex connected* graph as *biconnected* graph.

The proposed strategy for network topology redesign to improve network reliability and to find out a cost-effective structure is composed of three parts: *Reduction*, *Augmentation* and *Fine-Tuning Reduction*. An approximation algorithm for the *Augmentation* problem was introduced in 1981 in [24]. In 1993 another heuristic for the same problem with better time complexity was proposed in [25]. Later, this approach has been improved [26] and it has been solved by applying a Genetic Algorithm [27]. In [28] Hackbarth et al. introduced a heuristic that covered the whole problem of telecommunications network design from a totally meshed network to a *two-edge connected* topology solving the *Reduction* and *Augmentation* problem. This contribution extended their approach:

- The problem is extended to a *two-vertex connected* topology;
- An additional step is proposed to remove redundant edges through (*Fine-Tuning Reduction*) for the final solution set;
- Time complexity and efficiency of the *Augmentation* are improved by applying a *modified Depth First Search* (DFS)[29] and classifying the candidate edges before *Augmentation*;
- Links are classified for efficient manipulation of the optimization considering practically relevant network optimization constraints.

The paper is organized as follows: firstly, it provides a mathematical formulation of the considered problem and a discussion of the related work. In terms of biconnectivity, several relevant concepts of graph theory are introduced and analyzed. Next, a detailed description of the algorithm is given. Furthermore, the efficiency of algorithm by means of the presentation of optimization results of *AT&T* backbone network and *G-WiN* network is shown. Finally, some parameters of the redesign algorithm are analyzed.

2 Problem Statement

2.1 Objective Function

The existing telecommunications network can be described as an undirected Graph $G(V, E, W)$ with node $v \in V$, edge $e \in E$ and weight $w \in W$. The weight *W* is a problem-specific parameter. In this approach the weight of edges is not the same for all calculation steps. For calculating network costs, it is based on bandwidth and length. But for other redesign steps, different functions are defined to represent the weight of an edge. This will be discussed in detail by the introduction of the redesign procedure.

The objective is to minimize total network costs by redesigning the network structure. Some links can be removed or added to the network considering the given constraints at all the time. All network nodes are fixed without any change and their costs are ignored in this work. The objective function is defined as:

$$C_{net} = \sum_{e \in E} x_e c_e \tag{1}$$

where e is an edge in *E*; x_e is a binary variable to check if edge *e* is accepted or not. c_e is the leasing cost of edge e, see Eq.2.

$$c_e = f(l_e, \mu_e) + C_{e,k} \tag{2}$$

where $f(l_e, \mu_e)$ is the leasing cost function of edge (optical cable) e depending on edge length l_e and bandwidth (i.e. capacity) μ_e. $C_{e,k}$ is the fixed cost for different edge (link) types *k* in terms of bandwidth μ_e. A practical example will be introduced in the section IV.

2.2 Constraints

A packet transmitted over an asynchronous, time–multiplexed, packet switched network (like IP networks) undergoes a queuing and serialization delay at the end/beginning of a link (router entrance/exit respectively) which depends on the link utilization [30][31]. Mean packet delay depends on the traffic rate λ_e and capacities of links μ_e, i.e. utilization $\rho_e = \lambda_e/\mu_e$. In this work the maximum link utilization is taken as the first constraint. Therefore, we firstly study the relationship between link utilization and delay. A number of systems models have been proposed to describe different characterized IP networks in the last years. Here each edge is modeled as an independent M/M/1 queuing system [32]. To apply this queueing model in this work, the following assumptions are used: 1) each queue has an exponentially distributed mean service time; 2) an average arrival rate of new packets, which follows a Poisson distribution; 3) the packet length is also exponentially distributed; 4) the network structure should have a fixed routing, where the channels are error-free, etc. If mean packet length is $E[l_P]$, mean packet delay is derived as Eq.3.

$$t_{M/M/1} = \frac{E[l_P]}{\mu_e} \cdot \frac{1}{1-\rho_e} \tag{3}$$

If ρ_e increases, mean packet delay will be incremented, too. The maximum link utilization allows us to keep the expected delay low.

The second constraint in this approach is the *maximum hop number* between source and destination, which influences delay, reliability and survivability of networks. The hop number is given as number of links one packet has to pass on its way from the source to the destination node. A network solution is only valid, if the maximum hop number is not exceeded by any relevant source-destination combination. Thus, by limiting the maximum number of used edges per routed path an additional indirect constraint for a low end-to-end delay is given. Generally, every link has a certain reliability. By limiting the maximum hop number also the end-to-end reliability is manipulated, because along one path the single link reliabilities affect the total end-to-end reliability. Moreover, too high hop number makes it difficult to apply some routing protocols, such as Distance-vector Routing Protocol. The actual propagation delay on the edges is not directly incorporated in the optimization algorithm, which can be studied as further work.

In addition, the proposed algorithm also takes into account the *maximum node degree* (maximum number of links which are connected to a node). For some network planning tasks the *maximum node degree* was a limiting requirement in [8][33][34].

Furthermore, network survivability should be fulfilled during the redesign. The link restoration assumes single, total failures of individual links and restores the entire (or partial) capacity of the failed link on one or several paths between two end nodes of the link [6]. In this work the edge/node-biconnectivity is applied to protect single link or node failures during the network redesign.

3 Graph Analysis

3.1 Node/Edge-Connectivity

A graph is connected, if there exists at least one path from any point to any other point in the graph; otherwise the graph is called disconnected. The edge (or node) connectivity of a graph *G* is the minimum number of edge (or node) deletions sufficient to disconnect *G*, which is characterized by Menger's theorem [35]. In view of the disjoint paths, an undirected graph $G = (V, E)$ is defined as *k*-edge-connected, if there are *k* paths between two nodes $v, v' \in V$ and these paths do not share any edge. If these paths do not have node between v and v' in common, graph G is defined as *k*-node-connected. Two connected graphs are shown in Fig.1, where the left tree-topology is a *1*-edge/node-connected graph (or edge/node-connected) and the right mesh-topology is 3-edge/node-connected. The term mesh does not imply that the network topology is a full mesh, but rather that the network is at least two (edge) connected [36][23]. Therefore, the mesh structure has a higher survivability than the tree structure in telecommunications networks, but is more expensive due to additional edges. There is a close relationship between edge- and node-connectivity. The node connectivity is never smaller than the edge connectivity, since deleting one node incident on each edge in a cut set succeeds in disconnecting the graph [37][38].

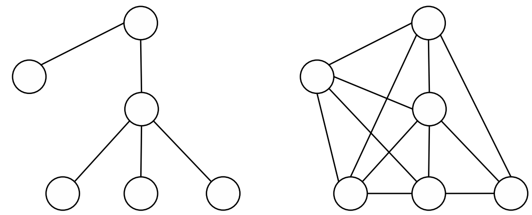


Figure 1. Connected graph: tree and mesh

3.2 Biconnectivity

A graph is called two-edge connected (edge-biconnected), if there are at least two edge-disjoint paths between every pair of nodes. Similarly, a graph is called two-node connected (node-biconnected), if there are at least two node-disjoint paths between each pair of nodes. An example is shown in Fig.2. Every node-biconnectivity graph is also edge-biconnectivity, while the reverse is not valid [23]. A biconnected topology can effectively ensure the network survivability. If an efficient method is applied to change a graph from one-connectivity to biconnectivity, the network cost will not be significantly incremented. For instance, adding only a few necessary links can make a tree topology biconnected. Hence, this work addresses the node-biconnectivity to improve the network survivability, where two node-disjoint paths can be available for the flows between source and destination.

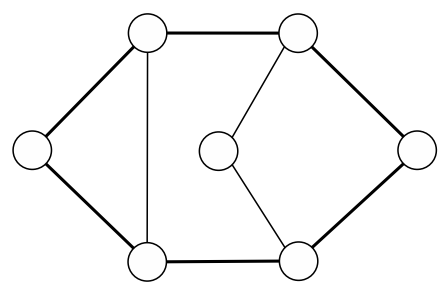


Figure 2. Two-node/edge-connected topology

3.3 Block Structure and Articulation

The *articulation* of a connected graph is a node whose removal will disconnect the graph [39]. Fig.3 shows two graphs with articulation nodes, which are depicted with gray color. The right graph represents a graph with 1-node-connectivity, but 3-edge-connectivity for more than two edge-disjoint paths for any pair of nodes.

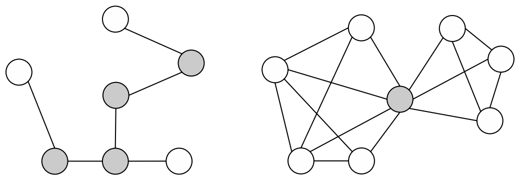


Figure 3. Examples with only 1-node-connectivity

A *block* is defined as a maximal biconnected subgraph for an undirected graph $G = (V, E)$. If this graph is biconnected, G itself is called a block. If graph G has N blocks and $i, j \in [1, N]$, $G_i = (V_i, E_i)$ is defined as block i with

(a) $|V_i \cap V_j| \leq 1$ for $i \neq j$;

(b) articulation node $a \in V$, if $|V_i \cap V_j| = \{a\}$ for $i \neq j$.

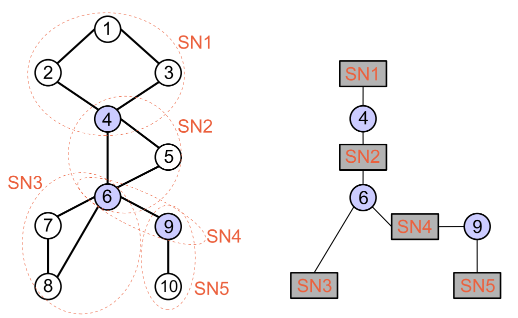


Figure 4. Graph G and its block graph *B(G)*

The block structure is defined as a block graph *B(G) = (V',E')* of graph *G*. *B(G)* is made up of blocks and articulation nodes, which can be found by a modified *Depth First Search* (DFS) [29] based on the method of Tarjan [40]. In Fig.4, the left graph presents a 10-nodes-topology with 5 subnets (SNs), the right graph presents a block graph with 5 blocks (square) and articulation nodes *4, 6, 9*.

4 Description of the Algorithm

4.1 Notation

In the following an overview of the used notations is given:

E_0 Set of edges representing the existing links (active);

E_R Set of edges representing the links (active) after the *Reduction*;

E_A Set of edges representing the links (active) after the *Augmentation*;

E_F Set of edges representing the links (active) after the *Fine-Tuning Reduction* $\longrightarrow$ Solution set;

E_{FIX} Set of edges representing the fixed links (active);

E_{POT} Set of edges representing the potential links (inactive);

E_{RED} Set of edges representing the reduced links (inactive) during the *Reduction*;

E_{AUG} Set of edges representing the augmented links (active) during the *Augmentation*;

E_{FTR} Set of edges representing the Fine-Tuning reduced links (inactive) during the *Fine-Tuning Reduction*;

w_{cost} Weight of cost for calculating cost-metric;

$w_{capacity}$ Weight of capacity for calculating cost-metric during the *Reduction*;

w_{flow} Weight of flow for calculating cost-metric during the *Augmentation*;

$w_{utilization}$ Weight of utilization for calculating cost-metric during the *Fine-Tuning Reduction*.

Note: The *active edges* are a part of the current network and thus they are used during the network calculation. The *inactive edges* are not a part of the current network.

4.2 Redesign Strategy

The complete redesign procedure consists of *Reduction*, *Augmentation*, *Fine-Tuning-Reduction*, as shown in Fig. 5.

Graph theory helps us to formulate the problem as follows (*nodes* are represented by *vertices* and *links* by *edges*): Let $E_0 \subset E$ be a fixed set of operational edges (representing the existing links), such that $G(V, E_0)$ is connected. And Let $E_{POT} \subset E$ be a fixed set of given edges (representing the potential links), such that $E_0 \cup E_{POT} = E$. The biconnectivity problem can be subdivided as follows:

1. The first step (*Reduction*) is to find a set $E_{RED} \subset E_0$ with $E_R = E_0 - E_{RED}$ which reduces the number of edges (the network costs) to a minimum without violating the constraints.

2. The second step (*Augmentation*) is to find a set $E_{AUG} \subset (E_{POT} \cup E_{RED})$ of augmenting edges with minimal costs, such as the biconnected graph $G(V, E_A, W)$ with $E_A = E_{AUG} \cup E_R$.

3. On augmenting E_R to E_A it is possible that edges from E_R become redundant for the graph biconnectivity. Hence, the third step (*Fine-Tuning-Reduction*) is to further reduce the network cost by finding a set $E_{FTR} \subset E_R$. Then the number of edges are decremented to $E_F = E_A - E_{FTR}$ subject to constraints and biconnectivity.

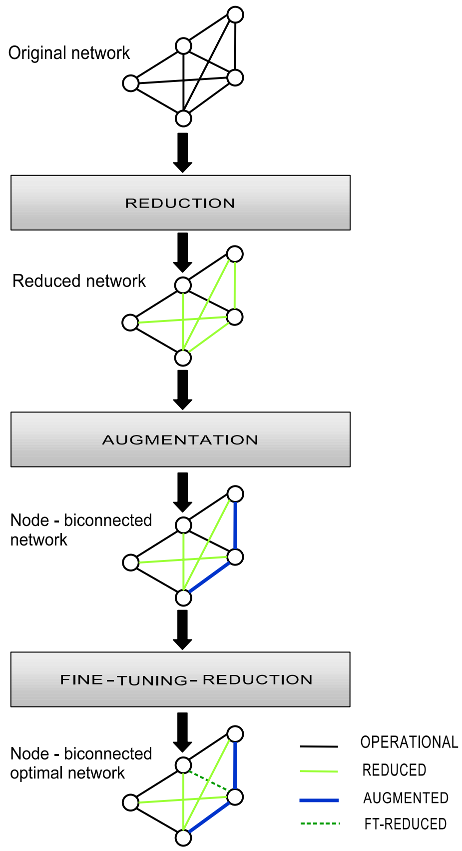


Figure 5. Flow chart of the redesign strategy

It is assumed that the network consisting of initial links (E_0) and potential links (E_{POT}) must fulfill the constraints and guarantee the biconnectivity.

The advantage of our edge redesign is that the history of the network and the experience of the network planner is taken into consideration because *Reduction* bases on the set of edges E_0 which represents the real existing links. Moreover, we introduce another possibility to further influence the direction of optimization by implementing the set $E_{FIX} \subset E_0$. Edges of the set E_{FIX} are not allowed to be reduced by the algorithm and hence they constitute a definitive part of the solution set E_F. E_{FIX} makes it possible to consider practical relevant situation, e.g. a long term of a leased line link. (Note that real existing links as well as potential links can be assigned to the set E_{FIX} at the beginning of the optimization.)

4.3 Reduction

Suppose E_0 is the set of edges in a graph G. The objective of the *Reduction* is to find a set E_{RED} that decreases the set of edges E_0 to a minimum E_R ($E_R, E_{RED} \subset E_0$ and $E_R \cup E_{RED} = E_0$) such that E_R is a graph with minimal number of edges (costs) still fulfilling the constraints (maxHop, maxUtil, maxNodeDegree). Ideally, this graph is a minimal spanning tree. Any edge $e \in E_0$ is evaluated by a cost metric $Metric_e^{RED}$ with normalized cost c_e and bandwidth μ_e:

$$Metric_e^{RED} = \frac{w_{cost} \cdot min\{c_{e,e\in E_0}\}}{c_e} + \frac{w_{capacity} \cdot \mu_e}{max\{\mu_{e,e\in E_0}\}} \tag{4}$$

with constraints:

$$w_{capacity} + w_{cost} = 1$$

$$0 \leq w_{capacity} \leq 1$$
$$0 \leq w_{cost} \leq 1$$

$$0 < Metric_e^{RED} \leq 1$$

The normalized values are used to efficiently represent the cost metric. The cost and capacity weights influence the evaluation of the cost metric. We assumed a cost function for the links, which is derived and estimated from basis network rate for leased lines of the *Deutsche Telekom* (2004) [41]. Depending on the link capacity and length, the costs consist of a base rate and a piecewise linear increasing cost function (CU: *Cost Unit*), as shown in Fig. 6. In a sense, the leasing cost of *10* Gbit/s is less than the leasing cost of *2.5* Gbit/s multiplied by *4*. Therefore, the edges with high cost (long optical cable) and low capacity will be preferably removed. The smaller $Metric_e^{RED}$ is, the earlier the edge *e* is reduced. During the *Reduction* the constraints have to be fulfilled, such as capacity, utilization, etc.

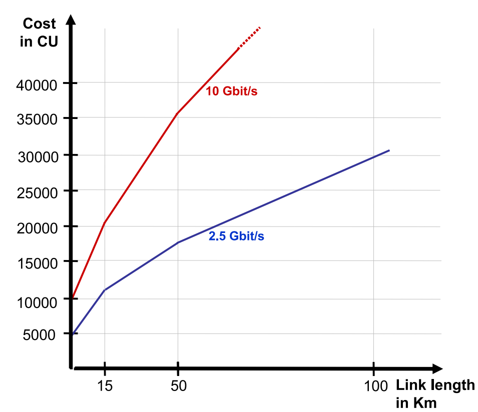


Figure 6. Cost function for calculation of leased line link cost per year in terms of link capacity and link length

We suppose that all demands are routed, and the edge loads, edge utilizations are calculated. The *Reduction* algorithm can then be described as follows:

1. The edge weights composed of a weighted standardized sum of real edge cost and load, are calculated for all edges of E_0.

2. On basis of the edge weights one edge is selected and temporarily added to the set E_{RED}: Thus, the selected edge is deactivated.

3. The demands are rerouted and then edge loads and edge utilizations are calculated considering all edges belonging to the sets E_0 and E_R.

4. If all constraints are fulfilled, the selected edge is reduced, which means it is finally added to the set E_{RED}. If a constraint is not fulfilled, the selected edge is marked as required, which means it is added to the set E_R.

5. The algorithm recalculates the edge weights and selects the next edge.

This process is repeated until all edges from E_0 are transferred to the sets E_{RED} or E_R. The program flow chart (PFC) of the *Reduction* is depicted in Fig. 7.

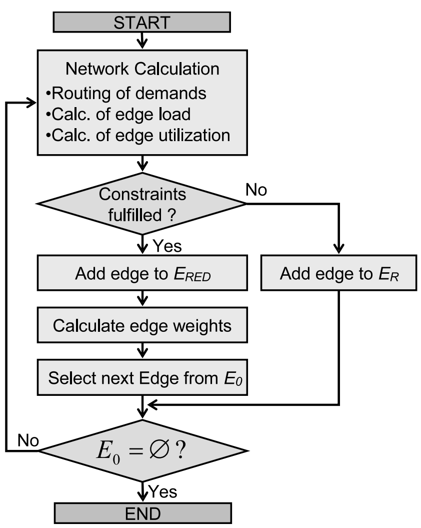


Figure 7. Program flow chart of the *Reduction*

Note that the *Reduction* algorithm solution depends on the sequence of selected edges. Hence, a unique branch of the complete combinatorial solution tree is calculated.

4.4 Augmentation

The objective of the *Augmentation* is to find out a set E_{AUG} from sets E_{RED} and E_{POT} so that the graph *G* with the edges $E_A = E_R \cup E_{AUG}$ between vertices is biconnected. In Fig. 8 the PFC of the *Augmentation* is depicted. At first the graph is analyzed by means of the modified *Depth First Search* (DFS) mentioned in the last section. The DFS finds all articulation points of *G*, marks them, and splits the graph into its biconnected components (subnets). If the graph is biconnected without any articulation point), the *Augmentation* will be stopped, otherwise the next *Augmentation* step is done.

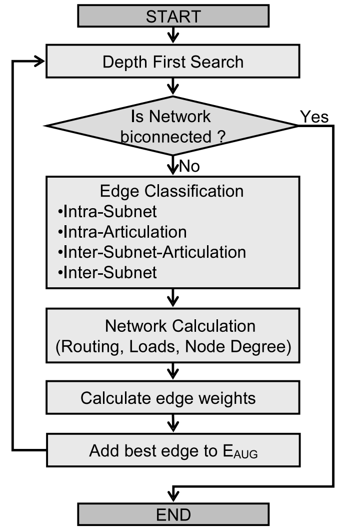


Figure 8. Program flow chart of the *Augmentation*

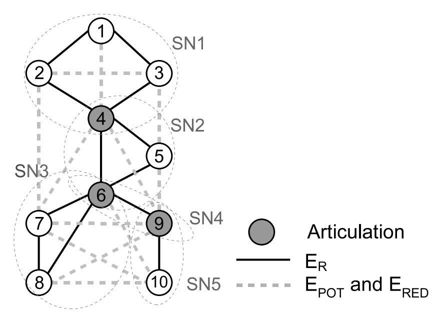


Figure 9. *Augmentation* network example: DFS marks articulation points and divides network into subnets: In black are the edges of the set E_R and in gray (dashed lines) are the edges depicted of the sets E_{RED} and E_{POT} (potential candidates for the *Augmentation*). The three articulation points are marked with a dark color and the different subnets are surrounded by dashed lines.

Figure 9 shows an example of a not biconnected graph after the DFS. Next, we classify the potential candidate edges in a way that only edges between vertices of different

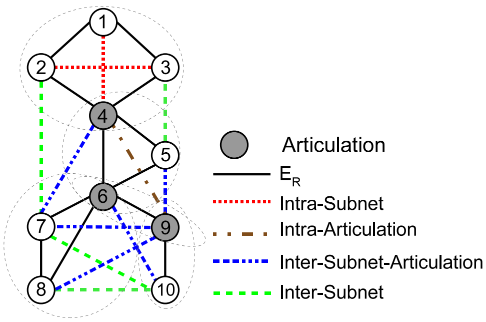


Figure 10. *Augmentation* network example: Classification of the potential candidate edges (E_{POT} and E_{RED}) depending on source and destination vertex

subnets are accepted as candidates for the *Augmentation*, while no vertex is an articulation point (Inter-Subnet). In Figure 10, the edge classification is shown for our network example. Due to the classification Intra-Subnet edges like {2,3} or {1,4} are not considered. Furthermore, we disregard Intra-Articulation edges like {4,9}, since their introduction would not improve the network situation regarding to the *Augmentation* problem. As a rule, we do not consider edges between different subnets, if one vertex is an articulation point (Inter-Subnet-Articulation), e.g. {7,9} or {10,6}, because in comparison with the Inter-Subnet edges they are only the second best option. Only in case that the graph is not biconnected, and no Inter-Subnet edge is left, these edges are accepted.

After that, a special cost metric $Metric_e^{AUG}$ is used to evaluate edge e subject to edge cost, subnet flow and node degree:

$$Metric_e^{AUG} = \frac{w_{cost} \cdot min\{c_{e,e\in(E_{POT}\cup E_{RED})}\}}{(d_v + d_{v'}) \cdot c_e} + \frac{w_{flow} \cdot f_e^{sn-sn}}{(d_v + d_{v'}) \cdot max\{f_{e,e\in(E_{POT}\cup E_{RED})}^{sn-sn}\}} \quad (5)$$

with constraints:

$$w_{flow} + w_{cost} = 1$$

$$0 \le w_{flow} \le 1$$
$$0 \le w_{cost} \le 1$$

$$0 < Metric_e^{AUG} \le 0.5$$

where v and v' are end nodes of edge *e* between two blocks (unbiconnected subnets). d_v and $d_{v'}$ are node degrees of v and v', respectively. f_e^{sn-sn} is the flow over edge *e* between two subnets. $min\{c_{e,e\in(E_{POT}\cup E_{RED})}\}$ and $max\{f_{e,e\in(E_{POT}\cup E_{RED})}^{sn-sn}\}$ present the minimal cost metric and maximal flow for $e \in (E_{POT} \cup E_{RED})\}$. The existing network topology is connected so that d_v and $d_{v'}$ are no less than *1*. Hence, $Metric_e^{AUG}$ is between *0* and *0.5*. Edges are preferably introduced when they are not costly, highly loaded (due to a high subnet-subnet-flow) and where the average degree of source and destination vertex is small. More precisely, the edges with a high cost metric preferably added.

Regularly for each edge a single routing has to be done where all other candidates are deactivated. Due to the time complexity of calculating the edge loads, an estimator is used. Therefore, all candidates are temporarily added to E_{AUG}. Then a routing of the demands and a load calculation is done considering the edges of E_{AUG} and E_R. On basis of the weights, one edge is selected and then definitively added to the set E_{AUG}, the remaining temporarily added edges are retransferred to their former sets. The described procedure is repeated until the DFS algorithm confirms biconnectivity of the graph.

4.5 Fine-Tuning Reduction

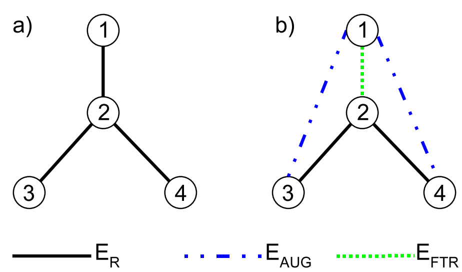


Figure 11. *Fine-Tuning Reduction* example: a) shows graph after *Reduction* b) shows that due to the *Augmentation* one edge from E_R becomes mathematically redundant for the biconnectivity of the graph

On augmenting E_R to E_A it is possible that edges from E_R become redundant for the graph biconnectivity of E_A. The example in Fig. 11 shows a graph, where E_R has the shape of a tree structure (Fig. 11a). The *Augmentation* added the minimal number of two edges {1,3}, {1,4}. Hence, E_A is biconnected. From the viewpoint of graph theory, the graph would still be biconnected, if we discard the edge {1,2} of the former tree structure (Fig. 11b). Thus, objective of the *Fine-Tuning Reduction* is to find a set $E_{FTR} \subset E_R$ that reduces the number of edges to $E_F = E_A - E_{FTR}$ with a minimum total cost, considering constraints and graph biconnectivity at any time. However, we only consider edges of the set E_R as candidates for the *Fine-Tuning Reduction*, because if the *Augmentation* performs well hardly edges from E_{AUG} will become redundant.

The PFC of the *Fine-Tuning Reduction* is in general the same as the PFC of the *Reduction* (Fig.7). At the beginning we have a set of edges E_A. At each loop one edge out of the former set E_R is selected and temporarily added to the set E_{FTR}. Then this edge is deactivated. Then the network is calculated, such as routing of the demands, calculation of edge loads and edge utilizations. After that, the selected edge is either definitely assigned to the set E_F, if a constraint is not fulfilled or the remaining graph is not biconnected. Or it is assigned to the set E_{FTR}, if the graph is still valid. This procedure is repeated until all edges of the former set E_R have been tested. The sequence of tested edges is defined by the edge weights, which are recalculated in every cycle. The cost metric $Metric_e^{FTR}$ is composed of a weighted, standardized sum of edge leasing cost c_e and edge utilization ρ_e in Eq.6. The edges with a low cost metric will be firstly considered to be reduced.

$$Metric_e^{FTR} = \frac{w_{cost} \cdot min\{c_{e,e \in E_R}\}}{c_e} + \frac{w_{utilization} \cdot \rho_e}{max\{\rho_{e,e \in E_R}\}} \tag{6}$$

with constraints:

$$w_{cost} + w_{utilization} = 1$$

$$0 \leq w_{cost} \leq 1$$

$$0 \leq w_{utilization} \leq 1$$

$$0 < Metric_e^{FTR} \leq 1$$

4.6 Random Selection

To avoid trapping into local optima, a random process is applied to determine which edges are removed or added. In addition, we assume $pThres_{RED}$, $pThres_{AUG}$, $pThres_{FTR}$ as potential thresholds for *Reduction*, *Augmentation* and *Fine-Tuning-Reduction* to efficiently limit the search space. The random selection is described as following:

1. Evaluate cost metrics for all candidate edges and sort metrics from the minimal metric (*Min_Metric*) to the maximal metric (*Max_Metric*);

2. Find all edges with metric $\geq Min\_Metric(1 + pThres_{RED})$ for *Reduction*, or all edges with metric $\leq Min\_Metric(1 - pThres_{AUG})$ for *Augmentation*, or all edges with metric $\geq Min\_Metric(1 + pThres_{FTR})$ for *Fine-Tuning-Reduction*;

3. Select one edge by means of a uniformly distributed random number.

5 Results and Analysis

5.1 Redesign Environment

The algorithm is implemented in C++ and applied using a commercial network planning tool - NetWorks [42], designed for optimizing large scale telecommunications and IP networks. We studied the practical behavior of the algorithm on a set of real network examples. Two test networks will be introduced in this section. The optimization environment is based on a standard PC (Pentium III with *800* MHz and *384* Mbyte RAM). Each scenario with different parameters has been tested for at least 10 times.

We estimated demands between the nodes by applying the gravity coefficient method [43]. Therefore, for each pair of nodes the demands were calculated in direct proportion to the population of the corresponding area, and in indirect proportion to the distance between the individual nodes. Furthermore, the edge cost is derived from the leasing lines, see Fig. 6 and Eq. 2.

The information on the following networks has been collected from public sources. Since the traffic matrices haven't been available, the previously mentioned gravity model was used to generate one. This means that the following results are not absolutely applicable to the studied networks. But this is not the intention of the study. The idea is to show the application of the algorithm using real life topologies.

5.2 Test Network I: G-WiN

G–WiN was a part of Germany Research Education Networks by DFN-Association [44], which connects *42* core nodes by *75* links at two levels. The bandwidth of links ranges from 622 Mbit/s to 10 Gbit/s. In this approach we focus on the 10 core nodes (level 1) and their connecting 21 links (see Fig. 12). The potential edges are assumed to have a capacity of 2.5 Gbit/s. Furthermore, the maximal edge utilization is 99.9999% without limit of maximal hop number and node degree.

Fig. 12 shows the original links used for *G–WiN*. As result of the redesign redundant and high capacity (expensive) edges are successfully removed in Fig. 13. During the redesign all constraints have to be fulfilled.

More details are presented in the Tab. 1. The cost and number of links are significantly reduced. Mean edge utilization is increased, which should be taken into account. However, the original average link utilization was low. After the redesign the maximum utilization value is decreased. Hence, this increment of the edge utilization has no remarkable influence on the redesigned *G–WiN*. The reduced node degree can save the ports of core nodes.

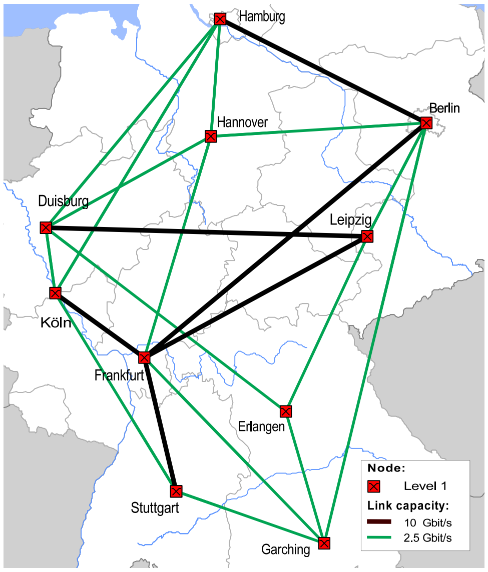


Figure 12. Original *G–WiN* (level 1)

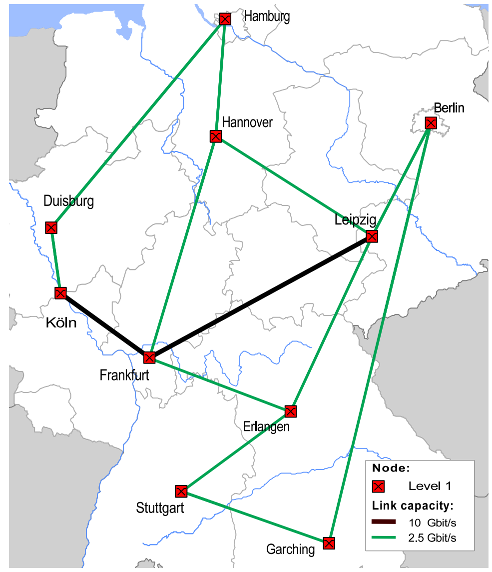


Figure 13. *G–WiN* after the redesign

Table 1. Optimization results of *G–WiN*

G–WiN	Original	Average
Cost in cost unit (CU)	1 402 247.49	671 915.49
Number of edges	21	13
Mean edge utilization	17.5%	41%
Max. edge utilization	97%	93%
Mean hop number	1.64	2.29
Max. hop number	3	5
Mean node degree	4.2	2.6
Max. node degree	6	4

5.3 Test Network II: AT&T

The level one and two of the *AT&T* backbone network (shown in Fig. 14a) is applied to redesign, which is comprised of *37* nodes in major US cities and *58* links (*10* Gbit/s or *2.5* Gbit/s link capacities) interconnecting them [42].

Figure 14 shows the procedure of the algorithm from the initial network with the existing links (set E_0) and the potential links (set E_{POT}) and to the final biconnected solution with the following settings and constraints:

- all links of the set E_{POT} have a capacity of *2.5* Gbit/s
- maximum link utilization = *99%*
- maximum hop number = *16*
- maximum node degree = *4*

Figure 14b) depicts the network after the *Reduction* (E_R = *37* links). The algorithm almost achieves a pure tree structure, only the *10* Gbit/s link {*Chicago*, *New York*} can not be reduced due to the effect of a too high utilization resulting from its removal. During the Augmentation *18* links (with dashed lines marked links in Fig. 14c) are added to the network in order to make it biconnected, whereupon the *Fine-Tuning Reduction* discards *15* links again (set E_{FTR} in Fig. 14d) taking into account the constraints and biconnectivity. Figure 15 compares the network topology before and after the optimization. The corresponding optimization results are summarized in Table 2.

Table 2. Optimization results of *AT&T* backbone network

AT&T	Original	Best	Average
Cost in CU	8 901 599	3 712 977	4 259 874
Number of edges	58	40	42.1
Mean edge util.	16%	51%	41.8%
Max. edge util.	54%	96%	92.4%
Mean hop number	3.47	6.59	6.1
Max. hop number	9	16	15.5
Mean node degree	3.13	2.14	2.25
Max. node degree	9	3	3.5

As mentioned, the solution depends on the sequence of edges (selected during *Reduction*, *Augmentation* and *Fine-Tuning Reduction*). If the edge with the best weight is always selected, the same unique branch of the complete combinatorial solution tree will be generated again and

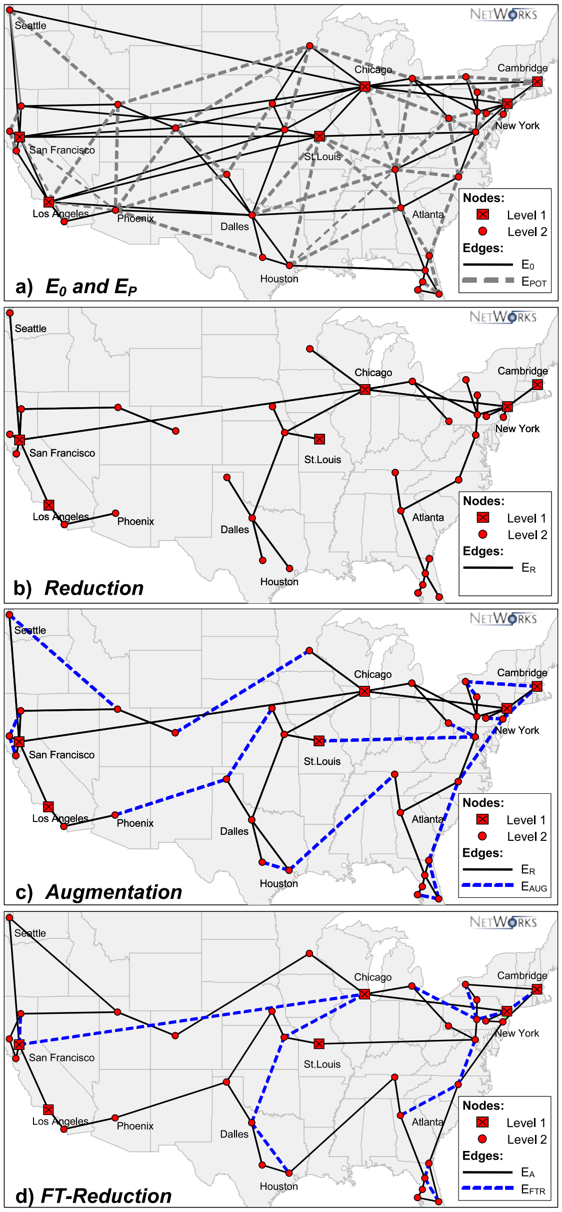


Figure 14. *AT&T* backbone network redesign: Successive depiction of the network after every optimization step. a) initial network, b) after *Reduction*, c) after *Augmentation*, d) after *Fine-Tuning Reduction*

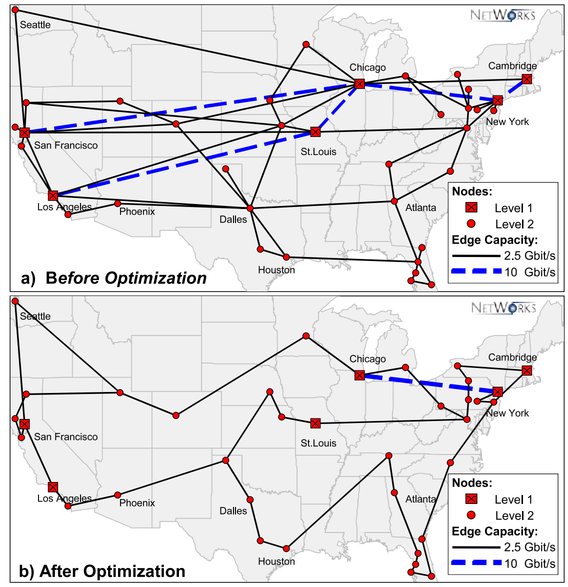


Figure 15. *AT&T* backbone network: Link capacities a) before and b) after optimization (maxHop = 16). The optimization process is illustrated in Fig. 14

again. But the deterministic solution is not always the best. Therefore, we implemented a random function that controls edge selection such that not always the best one is selected, but also the second or third best. For the *AT&T* network optimization we generated 50 solution sets (E_F) with an individual set of parameters. Afterward a statistical analysis has been performed and evaluated the efficiency of the chosen parameter. The algorithm took about *30* seconds calculating one *AT&T* network parameter set (one E_F). The result presented in Table 2 (column *Best*) and Fig. 15 is the best one out of fifty solution sets. The average values of the fifty solution sets are shown in column *Average* of Tab. 2.

Finally network cost are reduced to less than 50%. But the price is high: the mean link utilization, mean hop number and maximum hop number are increased. This results from a weakly restricted optimization. Hence, what we see is the potential saving of cost, if the network is operated at the limit and in consequence with a higher probability of having bad quality of service (QoS). This can be improved by tightening up the constraints. The degree of meshing can be decreased by adjusting the maximum hop number or the maximum link utilization. Our tests showed, that the hop number constraint is a better control because routing algorithm mostly finds the shortest path (with the least hop number), but does not consider link utilization. So a maximum utilization constraint can be easily exceeded and thus disturb the success of the optimization result (network could be not valid), although traffic engineering methods like load sharing or routing metric optimization [45] could balance network utilization situation.

In Fig. 16 an optimization result is shown with the same parameters like in the previous example, but with a maximum node degree of *5* and a maximum hop number of *9*. Thus, the maximum hop number is as big as in the original *AT&T* network. This optimized network has *51* active links and costs 4 284 965 CU. The average link utilization is *31%*.

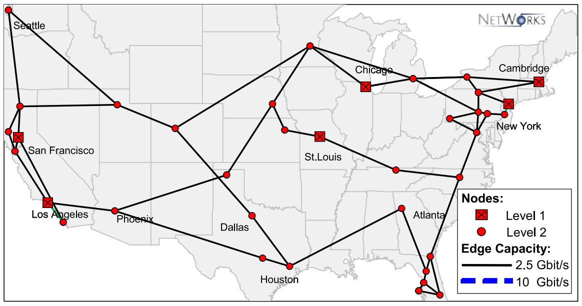


Figure 16. *AT&T* backbone network: Link capacities after optimization with constraints: max. Hop = *9*; max. link utilization = *99%*; max. node degree = *5*

Although maximum utilization is still high (*96%*), the network performs much better, since only one link is utilized with more than eighty percent and the average hop number is *4.59*. However, too high utilization has negative influence on the network operation. Therefore, the maximum utilization can be decreased from 99% to a lower threshold, or the mechanism of sharing load can be applied to reduce the link utilization. This paper provides only some results based on the predefined scenarios to show the characteristics of three-steps algorithms.

An interesting effect is the counter balancing behavior of the three optimization steps. Imagine an unfavorable *Augmentation*, adding a more than necessary number of edges to the graph. This does not necessarily lead to a bad overall result, because the supplementary edges also introduce additional optimization potential to the *Fine-Tuning Reduction*. In this case *Fine-Tuning Reduction* can compensate the bad *Augmentation* performance.

Besides the described directed optimization (network redesign) also the opposite is possible (designing new networks) by assigning all potential and real existing edges to the initial set E_0. Thus, the algorithm has all degrees of freedom to create a completely new solution set. Applying the new design method, our tests show e.g. for the *AT&T* network (Fig. 15, same constraints) a best solution set with minimal cost of over again 10% less compared to the best redesign method.

5.4 Parameter Studies

Different parameters influence the overall performance of the proposed algorithm. This paper presents some results of cost metrics, maximal hop number and potential threshold for network calculation and redesign. The points in the following figures are derived from 50 trials with confidence level 95%. Furthermore, a specified new-design is defined. Only one difference from the redesign is that this new-design takes all active and potential edges as operational edges, i.e. E_0 is extended. Referring to empirical results, both design processes are similar. Hence, the new-design of *AT&T* network is fulfilled to evalute the parameters mentioned above. The new-design has no limitation of node degree, but with maximal hop number and maximal edge utilization.

5.4.1 Reduction–Metric

By evaluating the cost metric for *Reduction*, w_{cost} and $w_{capacity}$ play an important role. As mentioned in *Reduction*, the metrics with low values will be removed with a higher probability. In addition, the weight of cost has more positive influence in the redesign than the weight of capacity. Namely, the expensive edges are preferably removed. In terms of w_{cost} and $w_{capacity}$, Fig. 17 presents optimization results after *Reduction*, *Augmentation* and *Fine-Tuning-Reduction*. A high weight of cost can lead to a better optimization solution, e.g. the right side of the figure w_{Cost} close to 1.0. Several unusual slight increment on the curves, such as Fine-Tuning-Reduction, stand the reason that the setting of w_{cost} and $w_{capacity}$ also depends on other parameters and could lead local optima.

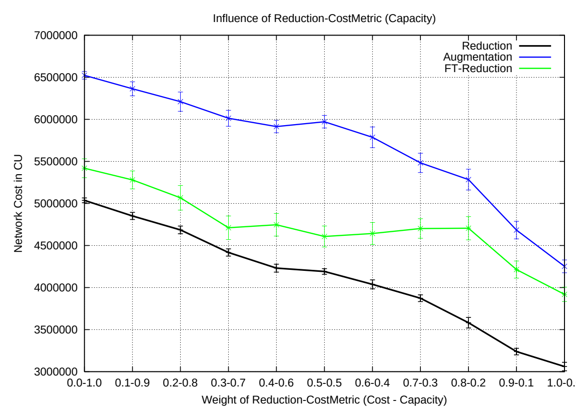


Figure 17. Influence of w_{cost} and $w_{capacity}$ for Reduction–Metric on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{RED} = 20$	$pThres_{AUG} = 10$
$Metric_e^{AUG}$:	$w_{cost} = 0.7$	$w_{flow} = 0.3$
$Metric_e^{FTR}$:	$w_{cost} = 0.7$	$w_{utilization} = 0.3$

5.4.2 Augmentation– and FT–Reduction-Metrics

Instead of $w_{capacity}$, the cost metrics of Augmentation– and FT–Reduction-Metrics take advantage of w_{flow} and $w_{utilization}$. Subject to hard constraints during the design, the candidate edges for both steps are strictly limited. Hence, Augmentation– and FT–Reduction-Metrics have similar characteristics in terms of optimization results. Hereby, only the optimization results of Augmentation-Metrics are presented and analyzed. Fig. 18 compares w_{cost} and w_{flow} of the Augmentation–Metric for three steps. The change of weights w_{cost} and w_{flow} has no explicit improvement for network design. However, the solutions with w_{cost} close to 1 are a little better than those with w_{cost} near 0.

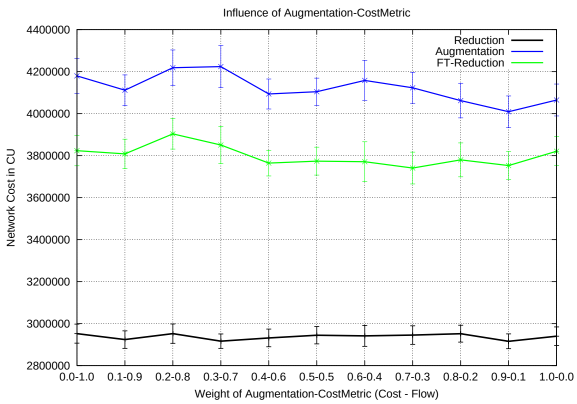


Figure 18. Influence of w_{cost} and w_{flow} for Augmentation–Metric on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{FTR} = 10$	$pThres_{AUG} = 10$
$Metric_e^{Rdkt}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

5.4.3 Maximal Hop Number

The maximal number of hops has been mentioned for *AT&T* network redesign. More test results are shown in Fig. 19 for *AT&T* network new–design. The lower the maximal number of hops is, the more expensive the designed network topology is. In this case more links are required to save hops for point–to–point connections and the node degree (maximal and mean) can increase. Therefore, maximal hop number significantly influences design results. However, if the node degree is limited, the influence of the maximal hop number could be changed.

5.4.4 Potential Threshold

Fig. 20 provides an overview of the optimization results in terms of different potential thresholds for *Reduction*, i.e. $pThres_{RED} \in [0\%, 700\%]$ with constant $pThres_{AUG}$ and $pThres_{FTR}$. The lower thresholds can lead to better solutions. Due to limited candidate edges, different potential thresholds of $pThres_{AUG}$ and $pThres_{FTR}$ have no large change. Hence, they are disregarded in this paper.

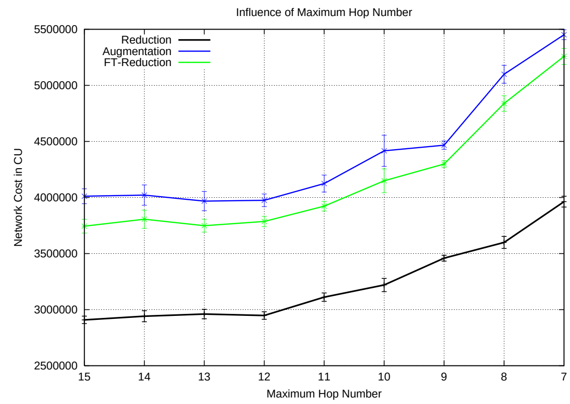


Figure 19. Influence of maximal hop number on network cost:

Constraints:	$max.Util. = 99.999\%$	
Random in %:	$pThres_{RED} = 10$	$pThres_{AUG} = 10$
$Metric_e^{RED}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

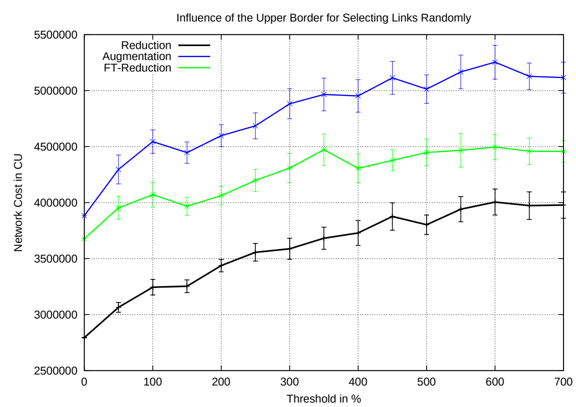


Figure 20. Influence of potential thresholds on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{AUG} = 0$	$pThres_{FTR} = 0$
$Metric_e^{RED}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

5.4.5 Summary

With the different configuration of parameters, several networks have been investigated. Tab. 3 summarizes the most useful parameter set, found empirically during the study.

6 Conclusion

In this paper we proposed a three-steps algorithm for network redesign problems to find a new fully biconnected

Table 3. The best found parameter setting for IP–Network design

Parameters	Values
Reduction:	
$Metric_e^{RED}$	$w_{cost} = 0.9 \quad w_{capacity} = 0.1$
$pThres_{RED}$	10%
Augmentation:	
$Metric_e^{AUG}$	$w_{cost} = 0.8 \quad w_{flow} = 0.2$
$pThres_{AUG}$	$0\% \leftrightarrow 10\%$
FT–Reduction:	
$Metric_e^{FTR}$	$w_{cost} = 0.8 \quad w_{utilization} = 0.2$
$pThres_{FTR}$	0%

cost-effective network topology. Hard constraints, such as hop number, edge utilization and node degree, are considered. In terms of numerical simulation results, the algorithm was verified to be flexibly applicable for redesign existing networks and creating new networks. The implemented algorithm has successfully applied to different network topologies, fulfilling the constraints and considering the biconnectivity, particularly with two-vertex connectivity. The redundant edges can be efficiently removed by the third step - *Fine-Tuning Reduction*, which improves the quality of network redesign in comparison with the previous work. A parameter study helps understanding the sensitivity of the results on algorithm settings.

In real network design situations, keeping the network operational is the most important task. A cost saving of e.g. 5% will always be critically examined by the network operator, since the occurrence of critical transition states during the migration has to be taken into consideration. Furthermore, network requirements continuously change, due to new services, new customers and new technologies. Thus, for a network operator it is not crucial to have (a temporarily) optimal network, but it is more important to know the optimal network. This knowledge helps making the right (most cost efficient) decision for necessary network extensions. The presented strategy supports network planners in these situations. Its high transparency and the possibility to closely interact by adjusting intermediate results manually assures its applicability.

References

[1] C. Minge, R. Zhao, M. Schweigel, "Improved Optical Network Topology Redesign Ensuring Biconnectivity", The Fourth Advanced International Conference on Telecommunications (AICT2008), p.398–403, Athens, Greece, June, 2008

[2] G.M. Schneider, M.N. Zastrow, "An Algorithm for the Design of Multilevel Concentrator Networks", Computer Networks, vol.6(1), p.1–11, 1982

[3] B. Gavish, "Topological design of Telecommunication Networks-the Overall Design Problem", European Journal of Operations Research, vol.58(2), p.149–172, 1992

[4] J. Petrek, V. Siedt, "A Large Hierarchical Network Star-Star Topology Design Algorithm", European Transactions on Telecommunications, vol.12(6), p.511–522, 2001

[5] A. Bley, T. Koch, R. Wessaely, "Large-scale hierarchical networks: How to compute an optimal architecture?", Proceeding of Networks 2004, p.13–16, Vienna, Austria, 2004

[6] M. Pioro, Deepannkar Medhi, "Routing, Flow, and Capacity Design in Communication and Computer Networks", Elsevier Inc. America, 2004

[7] B. Gavish, Topological design of Telecommunication Networks-local Access Design Methods", Annals of Operations Research, vol.33(1), p.17–71, 1991

[8] S. Chamberland, B. Sanso, O. Marcotte, "Topological Design of Two–Level Telecommunication Networks with Modular Switches", Operations Research, vol.48(5), p.745–760, 2000

[9] A. Girard, B. Sanso, L. Dadjo, "A Tabu Search Algorithm for Access Network Design", Annals of Operations Research, vol.106(1–4), p.229–262, 2001

[10] I. Godor, G. Magyar, "Cost-optimal Topology Planning of Hierarchical Access Network", Computers & Operations Research, vol.32, p.59–86, 2005

[11] R. Zhao, S. Goetze, R. Lehnert, "A Visual Planning Tool for Hybrid Fiber-VDSL Access Networks with Heuristic Algorithms", The 5th International Workshop on Design of Reliable Communication Networks (DRCN 2005), p.541–548, Island of Ischia (Naples), Italy, 2005

[12] M.R. Garey, D.S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-completeness", San Francisco, CA, Freeman, 1979

[13] S. Chopra, "Polyhedra of the Equivalent Subgraph Problem and Some Edge Connectivity Problems", SIAM Journal on Discrete Mathematics, vol.5, p.321–337, 1992

[14] T.L. Magnanti, S. Raghavan, "Strong Formulations for Network Design Problems with Connectivity Requirements", Networks, vol.45(2), p.61–79, 2005

[15] T.L. Magnanti, S. Raghavan, "Redesigning network topology with technology considerations", International Journal of Network Management, vol.18(1), p.1–13, 2008

[16] M. Weiss, F. Zeyer, "Redesign of Local Area Networks Using Similarity-based Adaption", Proc. 10th Conference on AI for Applications, p.284–290, San Antonia, TX, USA, 1994

[17] W.D. Grover, "Mesh–Based Survivable Networks", Prentice Hall, New Jersey, 2004

[18] R.H. Jan, F.J. Hwang, S.T. Chen, "Topological Optimization of A Communication Network Subject to A Reliablity Constraint", IEEE Transactions on Reliability, Vol.42, p.63–70, 1993

[19] S. Rai, D.P. Agrawal, "Distributed Computing Network Reliability", IEEE Computer Society Press Tutorial, USA, 1990

[20] T.G. Robertazzi, "Planning Telecommunication Networks", IEEE Press, USA, 1999

[21] L. Nederlof, K. Struyue, C. Shea, H. Misser, Y. Du, B. Tamayo, "End-to-End Survivable Broadband Networks", IEEE Commun. Mag., vol.9, p.63-70, 1995

[22] Y. Liu, D. Tipper, P. Siripongwutikorn, "Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing", IEEE/ACM Transaction on Networking, vol. 13, No. 1, p.198-211, February 2005

[23] R. Diestel, "Graph Theory", 2.ed, vol. 173, Graduate Textbooks in Mathematics, Spring-Verlag, 2000

[24] G. N. Fredericson, J. Jaja, "Approximation Algorithms for Several Graph Augmentation Problems", SIAM Journal on Computing, vol.10(2), p.270–283, 1981

[25] S. Khuller, R. Thurimella, "Approximation Algorithms for Graph Augmentation", Journal of Algorithms, vol.14(2), p.214–225, 1993

[26] S. Khuller, B. Raghavachari, A. Zhu, "A Uniform Framework for Approximating Weighted Connectivity Problems", Proc. 10th Annual SODA (ACM-SIAM), p.937–938, Baltimore (short paper), 1991

[27] I.D. Ljubic, J.J. Kratica, "A Genetic Algorithm for the Biconnectivity Augmentation Problem", Proceedings of the 2000 Parallel Problem Solving from Nature VI Conference, vol.1917, p.641–650, 2000

[28] K.-D. Hackbarth, A. Menèndez, C. Diaz, A.J. Portilla, "An approximative algorithm for fully biconnectivity and its application to network design", Euro-NGI.–D.JRA 3.1.2, p.95–106, 2004

[29] T. Erlebach, "Ausfallsicherheit von Netzen (reliability of networks)", Lecture manuscript, ETH Zürich, 2003

[30] S. Pracht, D. Hardman, W. Horneff, "Echotest in IP-Netzwerken (echo test in IP-networks)", vol.1, p.44–46, 2001

[31] J. Evans and C. Filsfils, "Deploying Diffserv at the Network Edge for Tight SLAs (Part 1)", IEEE Internet Computing", vol.Jan, p.61–65, 2004

[32] L. Kleinrock, "Queueing Systems, Volume II: Computer Applications", John Wiley & Sons, Canada, 1976.

[33] R. Zhao, H.J. Liu, R. Lehnert, "Topology Design of Hierarchical Hybrid Fiber-VDSL Access Networks with ACO", The Fourth Advanced International Conference on Telecommunications (AICT2008), p.232–237, Athens, Greece, June, 2008

[34] R. Zhao, Y. Zhang, R. Lehnert, "Topology Design of Hierarchical Hybrid Fiber-VDSL Access Networks with Enhanced Discrete Binary PSO", The Third International Conference on Access Networks (AccessNets08), Las Vegas, Nevada, USA, October, 2008

[35] K. Menger, "Zur Allgemeinen Kurventheorie", Fundamenta Mathematicae, vol.10, p.96–115, 1927

[36] R. Sedgewick, "Algorithms", 2.ed, Princeton University, Addison-Wesley, 1988

[37] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, "Network Flows: Theory, Algorithms, and Applications", Prentice Hall, United States Ed edition, 1993

[38] D. Jungnickel, "Graphenm, Netzwerke und Algorithmen", 3.ed, BI-Wissenschaftsverlag, Mannheim/Leipzig/Wien/Zuerich, 1994

[39] G. Chartrand, "Introductory Graph Theory", p.45-49, New York, Dover, 1985.

[40] R.E. Tarjan, "Depth First Search and Linear Graph Algorithms", SIAM Journal on Computing, vol.1, p.146–160, 1972

[41] M. Weinkopf, D. Reimer, "Entgeldantrag fuer digitale Standard- und Carrier-Festverbindungen, fuer den Comfort-Service (dSFV) und die Express-Entstoerung (CFV)", Technical Report, 2004

[42] NetWorks, http://www.networks.detecon.com/en/, 2008

[43] R. Baessler, A. Deutsch, “Nachrichtennetze (Telecommunications Networks)”, VEB Verlag Technik Berlin, Berlin, 1989

[44] Germany National Research and Education Network, http://www.dfn.de/, 2008

[45] A. Gous, A. Afrakhteh, “Traffic Engineering through automated optimisation of routing metrics”, Case Study, Cariden Technologies, 2002

Finding Inter-domain QoS Enabled Routes Using an Overlay Topology Approach

Şerban Gerogică Obreja, *IEEE Member*
Communications Department, ETTI
University POLITEHNICA Bucharest
Bucharest, Romania
e-mail: serban@radio.pub.ro

Eugen Borcoci, *IEEE Member*
Communications Department, ETTI
University POLITEHNICA Bucharest
Bucharest, Romania
e-mail: eugen.borcoci@elcom.pub.ro

***Abstract*— The transport of multimedia flows over the Internet needs to manage and control end to end Quality of Services (QoS) at transport level. One problem, which needs to be solved in this case, is finding inter-domain QoS enabled paths. This paper deals with the problem of establishing QoS enabled aggregated multi-domain paths, to be later used for many individual streams. It is proposed a simple but extendable procedure, based on overlay approach, to find several potential inter-domain end to end paths. The problem of QoS path finding is spilled in two phases. First several paths between the source and destination domains are found using the overlay topology. Then, the process of QoS path finding is completed in a second phase, when QoS enabled aggregated pipes are established, by negotiation between the domains' managers, along one of the paths founded in the first phase. The subsystem proposed is part of an integrated management system, multi-domain, dedicated to end to end distribution of multimedia streams.**

Keywords- QoS routing, end to end QoS management, pSLS, overlay network topology.

I. Introduction

The real time multimedia services, delivered on Internet networks, raised new challenges for the network regarding the end to end (E2E) quality of services (QoS) control in order to ensure the proper delivery of the services from content provider (source) to content consumer (destination). But traffic processing in real Internet deployments is still mostly best effort. Several approaches have been proposed, focused on provisioning aspects – usually solved in the management plane - and then in the control plane: e.g., well known dynamic techniques have been standardized, like IntServ, Diffserv, or combinations. The routing or - more generally - QoS enabled path finding and then maintaining, are also a part of the scene. Offering multimedia services in multi-domain heterogeneous environments is an additionally challenge at network/ transport level. Service management is important here, for provisioning, offering, handling, and fulfilling variety of services. Appropriate means are needed to enable a large number of providers to co-operate in order to extend their QoS offerings over multiple domains. To this aim, an integrated management system can be a solution, preserving each domain independency but offering integration at a higher (overlay) layer in order to achieve E2E controllable behavior.

This paper deals with the problem of establishing QoS enabled aggregated multi-domain paths, to be later used for many individual streams. It is proposed a simple but extendable procedure, running at management level, to find (through communication between domain managers) several potential inter-domain end to end paths. Then, using a resource negotiation process performed also in the management plane, QoS enabled aggregated pipes are established. All these function are performed at an overlay level, based on abstract characterization of intra and inter-domain capabilities delivered by an intra-domain resource manager. The subsystem is part of an integrated management multi-domain system, dedicated to end to end distribution of multimedia streams.

The QoS path finding is not a traditional routing process: it is not implemented on routers, and it doesn't choose a route between network devices, but between two or more nodes of an overlay virtual topology described at inter-domain level. Together with the intra-domain QoS routing available inside each network domain we will obtain an E2E QoS routing solution.

The main advantage of this solution is that, by separating the process of path finding from the QoS negotiation, the path searching process doesn't need to work real time. So we can find several paths in very complex overlay topologies. Also, by simplifying the overlay topology, by considering only the domain managers as topology nodes, our solution will work for very complex topologies, being no need for a hierarchical approach.

This paper is organized as follows: the Section 2 contains the state of the art in QoS inter-domain routing; the Section 3 shortly describes the general Enthrone architecture focusing on the service management at the network level. The Section 4 introduces the proposed QoS inter-domain path finding solution. Section 5 presents details about the implementation and Section 6 contains conclusions and possibilities of extensions and open issues.

II. State of the Art

Because our approach deals with QoS path finding and routing, a short overview of the available approaches for QoS routing is presented below [13][14][15][17][18]. We distinguish between intra- and inter-domain QoS problems.

The intra-domain QoS routing solutions could be divided in two major approaches.

Classically, intra-domain QoS routing protocols run on the routers and find paths with QoS constraints from source to destination.

Other solutions are based on a domain central manager, having knowledge of the total resource allocation inside the domain, and use an algorithm to determine QoS routes between source and destination. In this case the QoS routing process is run by a dedicated module of the domain manager, and the resulted route is installed on the network equipments by a network controller. Usually the QoS routing process is triggered by a new request for a QoS path through the domain.

For inter-domain QoS routing also we can distinguish between two kinds of approaches. The first one proposes enhancements for the BGP protocol in order to support QoS features. The BGP advertises QoS related information between autonomous systems (ASes), and the routing table is build taking into consideration this additional QoS information. The Q-BGP protocol, proposed in MESCAL project [20], is such an example.

Another category of inter-domain QoS routing solutions are based on the overlay network idea [13][14]. An overlay network is built, which abstracts each domain with a node, represented by the domain service manager, or with several nodes represented by the egress routers from that domain. Then protocols are defined between nodes for exchanging QoS information, and based on this information QoS routing algorithms are used to choose the QoS capable path. In [13] a Virtual Topology solution is proposed. The VT is formed by a set of virtual links that map the current link state of the domain without showing internal details of the physical network topology. Then a Push and a Pull model for building the VT at each node are considered and analyzed. In Push model each AS advertise their VT to their neighbor ASes. This model is suited for small topologies. In Push model the VT is requested when needed, and only from the ASes situated along the path between source and destinations, path which is determined using BGP routing information. If BGP kept several routes between source and destination than the VTs for each domain situated along the founded paths are requested. Based on this VTs information the QoS route from source to destination is calculated. After that an end to end QoS negotiation protocol is used to negotiate the QoS resources along the path.

One problem with these solutions is that they are based on the virtual available resource topology information obtained from other ASes. This requirement could be not accepted by the actual network providers, due to their confidentiality policy regarding their resource availability.

Also, these solutions based on an end to end QoS negotiation process. After the QoS path is found, the negotiation process is started. The QoS routing process previously performed is increasing the chance of negotiation success, but it implies two QoS searching processes: building the QoS topology and secondly negotiation in order to reserve resources.

This paper proposes a simpler approach by separating the process of path searching from the process of QoS negotiation (QoS searching path). By combining these two processes we will obtain a QoS inter-domain routing solution.

This was developed and integrated in an E2E QoS management system [2][8][9][10]. The system was proposed and implemented by an European consortium in the FP6 European project ENTHRONE [2][3][4][5], and continued with ENTHRONE II [6][7][8]. The ENTHRONE project is an integrated management solution based on the end-to-end QoS over heterogeneous networks and terminals. It proposes an integrated management solution that covers the entire audio-visual service distribution chain, including protected content handling, distribution across networks and reception at user terminals.

The overlay QoS path finding solution is based on the overlay network topology abstracting each pair (IP domain + manager) with a node. The overlay network in this case is only a connectivity one, with no information about the resources available intra and inter-domain. Several alternative inter-domain paths are computes, at overlay level, for each destination domain. Then, the end to end QoS negotiation mechanism is used to reserve resources. Together they will act as a QoS inter-domain routing algorithm.

III. ENTHRONE END TO END QOS MANAGEMENT SYSTEM

As mentioned before the ENTHRONE project, IST 507637 (continued with ENTHRONE II, IST 038463) European project, cover the delivery of real time multimedia flows with end to end quality of services (QoS) guarantees, over IP based networks. To achieve this goal, a complex architecture has been proposed, which cover the entire audio-visual service distribution chain, including content generation, protection, distribution across QoS-enabled heterogeneous networks, and delivery of content at user terminals [2][3][4][5][6][7]. A complete business model has been considered, containing actors (entities) such as: *Service Providers (SP), Content Providers (CP), Network Providers (NP), Customers (Content Consumers – CC), etc.*

A. *Enthrone features*

ENTHRONE has defined an E2E QoS multi-domain *Enthrone Integrated Management Supervisor (EIMS).* It considers all actors mentioned above and their contractual service related relationships *Service Level Agreements (SLA)* and *Service Level Specifications (SLS)* as defined in [2][3][4][5][6][7]. One of the main EIMS components is the service management (SM). It is independent of particular management systems used by different NPs in their domains, and it is implemented in a distributed way, each network domain containing Service Management entities. It is present in SP, CP, NP CC entities, depending on the entity role in the E2E chain. The SM located in NPs should cooperate with each IP domain manager and also with other actors in the E2E chain.

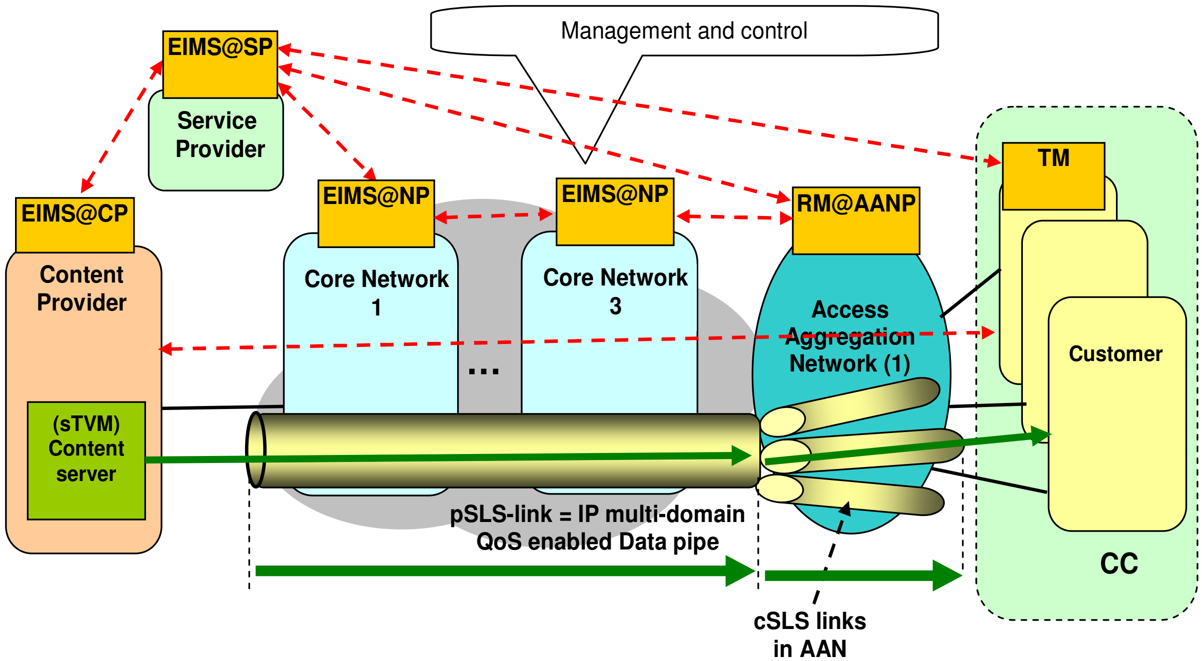


Figure 1. Forwarded cascaded model for pSLS negotiation

Legend:

EIMS@CP, EIMS@SP, EIMS@NP – ENTHRONE Integrated Management Supervisor at – respectively: Content Provider (CP) , Service Provider (SP) and Network Provider (NP)

sTVM –Source TV and Multimedia Processor (Content server)

AAN – Access Aggregation Network; AANP – AAN Provider

RM@AANP – Resource Manager of AAN (it is ENTHRONE compliant)

TM Terminal Manager

ENTHRONE supposes a multi-domain network composed of several IP domains and access networks (AN) at the edges. The CPs, SP, CCs, etc. are linked to these networks. The QoS transport concepts of ENTHRONE are shortly described below.

First, QoS enabled aggregated pipes, based on forecasted data, are established in the core network, part of the multi-domain network. They are logical pipes built by the Service Management entities. The aggregated QoS enabled pipe, called pSLS pipe, is identified by the associated pSLS agreement (Provider SLS) established between the Network Providers, in order to reserve the requested resources. Each pSLS-link belongs to a given QoS class, [20].

Then, slices/tracks of pSLS-links are used for individual flows based on individual cSLA/SLS contracts. An individual QoS enabled pipe is identified by a cSLS agreement, which is established between the manager of a Service Provider (EIMS@SP) and a CC for reserving the necessary resources for the requested quality of service. Several cSLSs pipes are aggregated at the core network level into an aggregated pSLS pipe.

In the data plane of core IP domains, Diffserv or MPLS can be used to enforce service differentiation corresponding to the QoS class defined. In the ANs, the traffic streams addressed to the users (Content Consumers) is treated similar to the *intserv*, i.e. individual resource reservations and invocations are made for each user.

B. Service Management at Network Provider

The EIMS architecture at NP (EIMS@NP) contains four functional planes: the *Service Plane (SPl)* establishes appropriate SLAs/SLSs among the operators/ providers/customers. The *Management Plane (MPl)* performs long term actions related to resource and traffic management. The *Control Plane (CPl)* performs the short term actions for resource and traffic engineering and control, including routing. In a multi-domain environment the *MPl* and *CPl* are logically divided in two sub-planes: inter-domain and intra-domain. Therefore, each domain may have its own management and control policies and mechanisms. The *Data Plane (DPl)* is responsible to transfer the multimedia data and to set the DiffServ traffic control mechanisms to assure the desired level of QoS.

The main task of the EIMS@NP is to find, negotiate and establish a QoS enabled pipe, from a Content Server (CS), belonging to a Content Provider, to a region where potential clients are located. Each pipe is established and identified by a chain of pSLS agreements, between successive NP managers. The forwarded cascaded model is used to build the pSLS pipes [5]. The pipes are unidirectional ones. An E2E negotiation protocol is used, [5] to negotiate the pSLS pipe construction across multiple network domains.

The process of establishing a pSLS–link/pipe is triggered by the SP. It decides, based on market analyses and users recorded requirements, to build a set of QoS enabled pipes, with QoS parameters described by a pSLS agreement. It starts a new negotiation session for each pSLS pipe establishment. It sends a pSLS Subscribe request to the EIMS@NP manager of the Content Consumer network domain. The EIMS@NP manager performs the QoS specific tasks such as admission control (AC), routing and service provisioning. To this aim, it splits the pSLS request into intra-domain respectively inter-domain pSLS request. It performs intra-domain routing to find the intra-domain route for the requested pSLS, and then it performs intra-domain AC. If these actions are successfully accomplished, and if the pSLS pipe is an inter-domain one, then the manager uses the routing agent to find the ingress point in the next domain, and send a pSLS Subscribe request towards the next domain. This negotiation is continued in chain, up to the destination domain, i.e., the domain of the CC access network. If the negotiation ends successfully, the QoS enabled pipe is considered logically established along the path from source to destination.

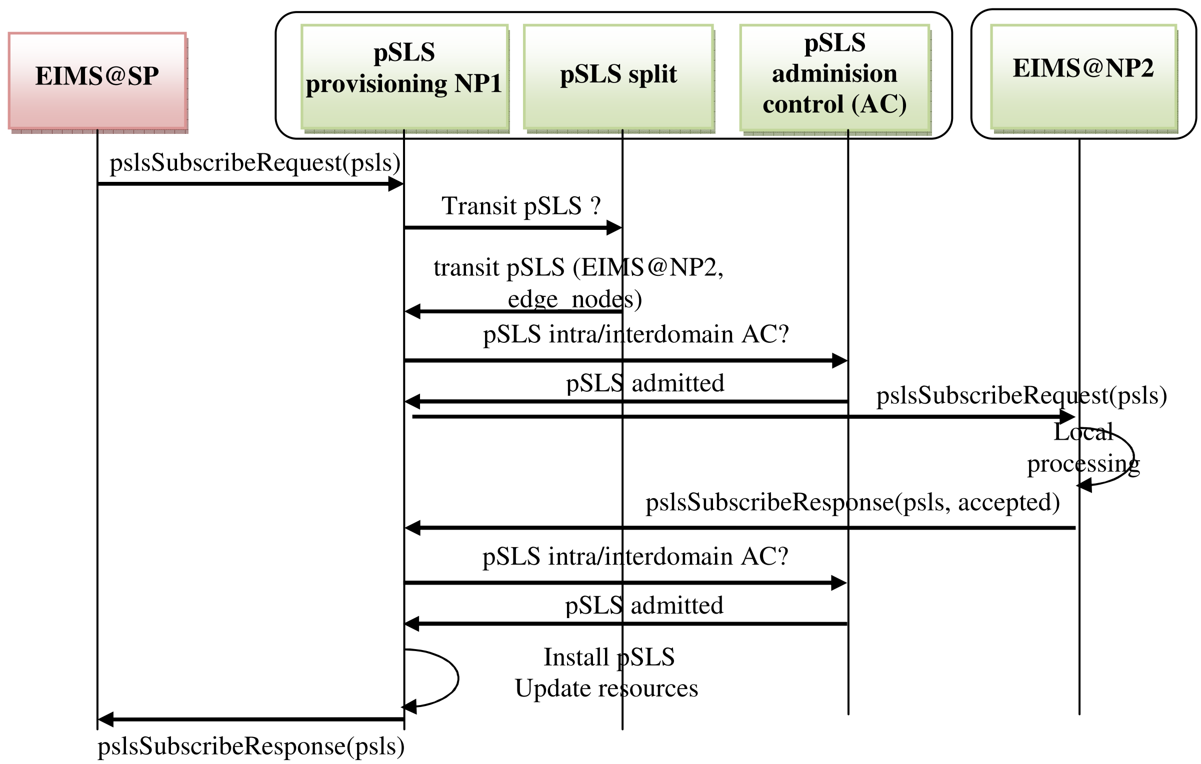


Figure 2. pSLS negotiation for QoS enable path establishment

The actual installation and configuration of routers is considered in ENTHRONE a separate action, and is done in invocation phase in a similar signaling way, plus the "vertical" commands given by EIMS@NP to the intra-domain resource manager.

After the pSLS pipe is active (i.e. subscribed and invoked) the Service Provider is ready to offer the new service to the users from the access network situated at the end of the pipe. Now the process of cSLS individual agreements establishment, for this new pSLS pipe, could be started.

IV. FINDING AN END TO END PATH WITH GUARANTIED QOS

A. General considerations

The main concepts of ENTRONE as stated in [8] are:

- E2E QoS over multiple domains is a main target of EIMS.
- But each AS has complete autonomy, regarding its network resources, including off-line traffic engineering (TE), network dimensioning and dynamic routing.
- Each Network Service Manager (cooperating with Intra-domain network resources manager) is supposed to know about its network resources in terms of QoS capabilities. ENTHRONE assumed that each AS manager has an abstract view of its network and output links towards neighbors, in a form of a set of virtual pipes (called Traffic Trunks in ENTHRONE I, see [5][6]), each such pipe belonging to a given QoS class.

A solution to this problem is to define/use routing protocols with QoS constraints, called QoS routing protocols. They can find a path between source and destination satisfying QoS constraints.

While finding the QoS path is only a first step, then maintaining the QoS with a given level of guarantees during the data transfer requires additional actions of resource management, including AC applied to new calls.

EIMS@NP management system performs these tasks. It is a centralized manager knowing the topology and resources of a domain. Being a central management node for a network domain, a centralized QoS routing solution is appropriate inside the domain.

On the other side, the multiple domain pSLS-links should also belong to some QoS classes and, therefore, inter-domain QoS aware routing information is necessary to increase the chances of successful pSLS establishment when negotiating the pSLSes. Several approaches are possible and they are summarized in [5]:

- NPs advertise their QoS capabilities with their associated scope through different methods (from automated peer-to-peer processes down to conventional techniques). A NP manager can locate and find out the QoS-classes offered by other domains (QoS capabilities, capacities, destination prefixes and costs).
- NPs implement a small number of well-known QoS classes. Inter-domain QoS services are created by constructing paths across those domains that support a particular QoS class. The BGP information is used to find destination prefixes. But QoS capabilities, capacities and costs, can be determined during pSLS negotiations – which may be successful or not.
- NPs advertise their QoS class capability and reachability through a protocol. Inter-domain QoS services are then created by constructing paths (which may not necessarily be the BGP path) across those domains that support a particular QoS class. This is path advertisement through a protocol.

B. The proposed overlay inter-domain QoS path finding solution

We proposed a simplified version [1], which takes into account the following assumption regarding the specific characteristics of the Enthrone system:

- The number of E2E QoS enabled pipes is not very large because they are long term aggregated pipes.
- The number of NP entities is much lower than the number of routers
- The EIMS@NPs are implemented on powerful and reliable machines, having enough computing and storage capabilities.
- The inter-domain core IP topology is rather stable and fixed; new elements are added at large time intervals.

This solution is also based on the idea of Overlay Virtual Network (OVN) [13], but in the first approach of our case, the OVN consists only of network domains (autonomous systems) abstracted as nodes. Each node will be represented by an EIMS@NP in this Overlay Virtual Network. This virtual network contains only information on connectivity between the domains, represented by the EIMS@NP nodes, or additionally static information regarding the inter-domain QoS parameters: links bandwidth, maximum jitter and delay, mean jitter and delay, etc.

This virtual connectivity topology (VCT) can be learned statically (offline) or dynamically.

The statically approach considers that the OVCT is built on a dedicated server – a topology server, like in the Domain Name Service (DNS). When a Network Provider wants to enter in the Enthrone system, then its EIMS@NP should register on this topology server. The topology server will return the Overlay Virtual Connectivity Topology. So, we will consider that each EIMS@NP has the knowledge of this connectivity topology. In the dynamic case each EIMS@NP, if wanting to build the OVCT, will query its directly linked (at data plane level) neighbor domains. It is supposed that it has the knowledge of such neighbors.

Each queried EIMS@NP returns only the list of its neighbors. At receipt of such information, the queerer EIMS@NP updates its topology data base (note that this process is not a flooding one as in OSPF). Then it queries the new nodes learned and so on. The process continues until the queerer node EIMS@NP learns the whole graph of "international" topology.

As we mentioned above the graph contains as nodes the EIMS@NP, which means that is made from the Network Service Managers of Enthrone capable domains.

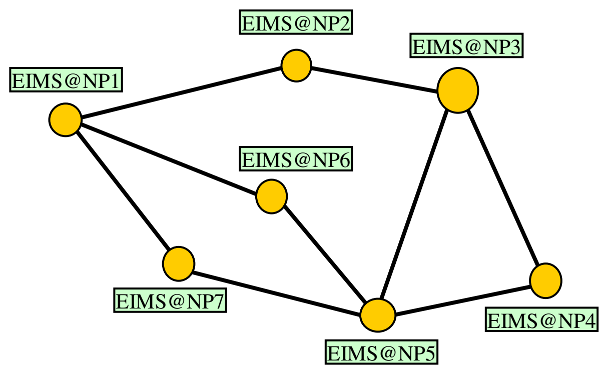


Figure 3. Overlay Virtual Connectivity Network

If the Enthrone system will be implemented at large scale, the number of nodes in the graph will be large, which means that the time required calculating the routing table will be also large. But because the topology changes events (adding new EIMS domains) are sparse ones (weeks, months), the topology construction process could run at large time intervals (once a day for example). In this case the routes calculation is triggered also at large time intervals, which means that it is enough time to determine the overlay paths. Another consequence is that the messages used to build the OVCT will not overload significantly the network. Enthrone capable domains can be separated by normal domains, with no Enthrone capabilities. In this case we consider that static QoS enabled pipes, are built between Enthrone capable domains, pipes crossing the Enthrone non capable domains. These domains (Enthrone non capable) will be transparent for the Enthrone domains.

On the graph learned, each EIMS@NP can compute several paths between source-destination pairs, thus being capable to offer alternative routes to the negotiation function.

The number of hops is used as a primary metric for the path choosing process. By the “hop” term we refer to a node in the Overlay Virtual Topology.

The process of route selection is as follows:

- When a request for a new pSLS arrived at one EIMS@NP, this will select the best path to the destination (the next EIMS@NP node that belong to this path), based on the overlay routing table.
- After the next hop is selected, the EIMS@NP will check if it has an intra-domain QoS enabled path for this route, i.e., between an appropriate ingress router and an egress router to the chosen next hop domain. If there is no such QoS enabled route, the next hope EIMS@NP node is selected from the overlay routing table.
- In case that, in the intra-domain, it is found a QoS enabled route, the EIMS@NP, based on mechanisms defined in Enthrone, trigger a request for a new pSLS negotiation to the chosen EIMS@NP neighbor.
- This process continues until the destination is reached. If the negotiation ends with success, than the pSLS pipe with guaranteed QoS parameters is found. If the process fails, then the EIMS@NP will choose another overlay path to the destination, and will start a new negotiation.

In Figure 4 the messages sequence for pSLS negotiation process, in the case of multiple paths towards the destination, is shown. The Service Provider decides to build a pSLS enable pipe between a source, located in the NP1 domain, and a destination, located in NP5 domain. We consider for this example that the working overlay topology is the one given in Figure 3. One can see that there are four possible routes between NP1 and NP5 domains. The first two of them, in terms of cost value, are the routs through NP6 and NP7 respectively. In Figure 4 it is illustrated the case when the pSLS negotiation along the route NP1-NP6-NP5 fails, due to admission control rejection by NP6 domain, either on intradomain pipe inside NP6 domain, or on the interdomain pipe between the NP6 and NP5 domains.

When it receives the rejection response at the pSLS subscription request the NP1 domain checks for an alternate route towards the NP5 domain. It finds the route through the NP7 domain, and starts a new negotiation using this new route. This negotiation ends successfully, so the QoS enable pipe between NP1 and NP5 will follow the route NP1-NP7-NP5.

This solution has the advantage of being simple, and that it not require at an AS the knowledge of current traffic trunks for the other network domains as in [13].

A drawback of our solution (proposed above) is a larger failure probability in negotiating a segment (therefore a longer mean time for negotiation process), if comparing with solutions which calculate the QoS path before the negotiation process. The latter approach increases the probability that the negotiation finished with success at the first try.

The path finding process described above is not based on BGP information at all. BGP is used only for best effort traffic. The process of QoS routing takes place at service management level. But it is possible in principle to use such BGP information.

V. Design details

A. *Routing tables*

As mentioned before this solution is based on the knowledge of the overlay network connectivity topology. The topology can be kept in a form of a square matrix. The dimension M is equal to the number of nodes in the overlay topology network. Each entry r_{ij}, has an integer value. A zero value means that there is no direct connectivity between the nodes *i* and *j*. A value different from zero, value *1* for example, implies that there is a direct connection between the two nodes:

$$r_{ij} = \begin{cases} 1 & if\ \exists L_{ij} \\ 0 & if\ not\ \exists L_{ij} \end{cases} \qquad (1)$$

L_{ij} represents the link between nodes i and j. Because the matrix is a sparse one, it can be easily compressed in order to be stored, in case that the dimension M is large.

Based on this overlay topology each EIMS@NP builds a routing table which contains, for each destination node in the network, the several possible paths to this destination node, and the costs associated with each of these paths. Because in the routing table several entries will exists for each destination, the QoS negotiation process will be able to be carried successively on multiple paths, increasing the probability that a path fulfilling the QoS requirements to be found.

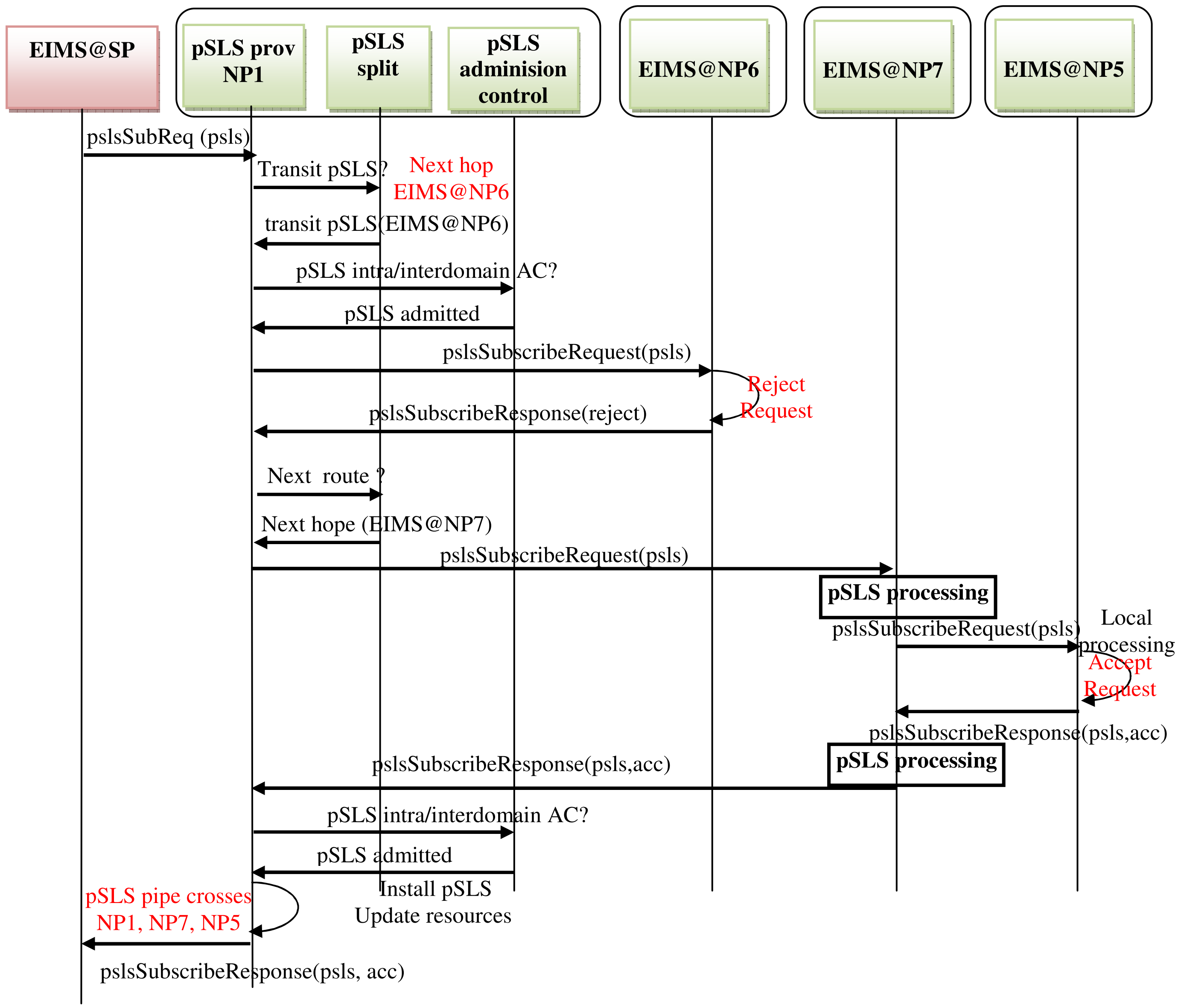


Figure 4. Overlay Virtual Connectivity Network

Because the number of possible paths from source to a certain destination could be high, we have limited it to the first four ones, with the lowest costs. If the neighbors number are less than four, than the number of possible routes towards a destination is limited to this number. It is used the same principle as in the case of distance vector protocols. In the case when there are several paths to the same destination EIMS@NP node, using as first next hop the same node, in the routing table it will be stored the best cost of all the possible paths going through that node.

This is not a limitation because in our case the routing decision is taken hop by hop so the source node has no idea what route to the destination will be chosen at the node where the paths are splitting. An EIMS@NP does not need to keep the whole path information (but the total cost only) because it cannot influence the route chosen decision at the next hops along the path.

Let's suppose that the EIMS@NP$_k$ node has the neighbor nodes EIMS@NP$_m$, EIMS@NP$_n$, EIMS@NP$_p$. The routing table from EIMS@NP$_k$ node to EIMS@NP$_l$ node will be:

TABLE I. ROUTING TABLE AT NODE *K* FOR NODE *L* DESTINATION

Destination	EIMS@NP$_l$	EIMS@NP$_l$	EIMS@NP$_l$
Nex Hop	EIMS@NP$_m$	EIMS@NP$_n$	EIMS@NP$_n$
Cost (Nb of hops)	5	0	3

The EIMS@NP at node *k* builds such a record for each node in the overlay network. This process, of searching several possible paths for each possible destination, in this overlay network topology, is an expensive one in terms of calculation. But based on the assumptions presented above, which are realistic ones, if such a management system will be implemented in the network domains, this routing table building process will be run only on topology updates, which means at very long time intervals. Such a process will put low computing overhead on the Service Manager. Also, it could be scheduled to run on intervals with low management activity [5]. Taking this in consideration, it could be considered that the routing table is a static one, and the route search process reduces to a simple database search one. It does not need to run the searching algorithm for each pSLS subscription request. It is enough to search, in the routing table, the route with the smallest cost, and forward the request to the chosen next node. If the negotiation for QoS parameters along this path failed, then it will chose the next path, in terms of cost, from the routing table.

B. Possible improvements

It is said that the solution did not take into consideration any QoS parameters, in the first phase, for path building process. This task left for the QoS negotiation process.

A possible improvement is to take into account some general data about the QoS parameters, in the path finding phase. For example, based on agreements with Service Managers of some domains, or based on some general QoS parameters of the domains, the Policy Based Management module could associate different costs for the links in the topology matrix. It is supposed that domains agreed to share these parameters, such as: the min/mean/max delay and jitter, introduced by the domain. In such a way the Policy module could influence the routing decision process. In this case the matrix element r_{ij} could be expressed as in (2):

$$r_{ij} = \begin{cases} c_{ij} & if\ \exists\ L_{ij} \\ 0 & if\ not\ \exists\ L_{ij} \end{cases} \qquad (2)$$

The value c_{ij} is the cost for the link L_{ij}, and could be established by weighting appropriately the general QoS parameters mentioned above. These weights could be established by the domain administrator and transmitted to the Policy module.

Also the cost of a link could be modified based on statistics regarding the acceptance or rejection rate of previous negotiated pSLS pipes. For example, if some domain with a good link cost rejects several times our requests we could modify the costs of the links crossing that domain.

Also, when the path cost is computed, it could be taken into account the existence of resource price agreements between some domains. These agreements could be negotiated using pull model, based on some statistics. For example, an EIMS@NP node has two different paths towards a destination with similar path costs. It chose the path with a better cost, but it also could periodically request resource price information from both neighbor nodes crossed by the two paths. If the second node has available resources and is interested to carry traffic from the source domain, it will propose a better resource price as a response to resource price requests. So the EIMS@NP source node could modify the routing table by improving the path cost for the second path, and the future pSLS pipe requests will be routed through the second path. Such a resource price communication could be easily implemented because the EIMS@NP managers are built as web-services, which implies very flexible communication capabilities.

C. Overlay topology building

For our solution we have chosen to build the overlay topology by means of successive interrogations of all the available nodes. The node, which decides to build/refresh the overlay topology, starts to interrogate all the other overlay nodes about their neighbors. It starts with its direct connected neighbors, and then continues interrogating the new found neighbors, and so on.

For the EIMS@NP implementation we have used the webservice technology. The interfaces between the EIMS@NP modules are implemented using WSDL language. The interdomain path finding WSDL interface it is used by EIMS@NP to interact with other EIMS@NPs, in order to build the overlay topology.

The inter-domain path finding WSDL interface has defined the following messages:

- ***getEimsNeighborsRequest*** ()
- ***getEimsNeighborsResponse***(EimsNeighborsArray eimsNeighbors)

- ***getDomainQoSRequest*** ()
- ***getDomainQoSResponse***(DomainQoS qos)

The first two messages are used by the Overlay Path Building module from EIMS@NP subsystem to build the overlay topology. The response message contains an array with all the neighbors of the interrogated domain, and their associated data about the webservices addresses, identification, and IP addresses.

The next two messages are used to get general information about the QoS parameters of the domain: min/max/mean delay and jitter, mean transit cost, max bandwidth. These values refer to the transit parameters for the domain. We have considered that such information could be offered by each domain without affecting its confidentiality policy. These parameters are used to establish the cost associated with a link between two neighbor domains. For establishing the cost we have weighted the normalized values for these parameters. The weights were chosen arbitrarily, such as their sum to be one. No studies have been done to find the optimal weights values.

The format of messages parameters are given in table 2.

TABLE II. DATA TYPE SECTION FOR THE INTERDOMAIN PATH FINDING WSDL INTERFACE

```
<wsdl:types>
     <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://webservice.enthrone.org/eims/
                   /InterdomainPath/datatype">
<xsd:complexType name="EndPoint">
<xsd:sequence>
        <xsd:element name="IPAddress" type="xsd:string"/>
         <xsd:element name="NetMask" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Neighbor">
<xsd:sequence>
           <xsd:element name="id" type="xsd:string"/>
    <xsd:element name="OverlayPathWebserv" type="xsd:string"/>
             <xsd:element name="Address" type="tns:EndPoint "/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="DomainQoS">
<xsd:sequence>
               <xsd:element name="minDelay" type="xsd:int"/>
               <xsd:element name="maxDelay" type="xsd:int"/>
              <xsd:element name="meanDelay" type="xsd:int"/>
              <xsd:element name="minJitter" type="xsd:int"/>
              <xsd:element name="maxJitter" type="xsd:int"/>
              <xsd:element name="meanDelay" type="xsd:int"/>
               <xsd:element name="meanCost" type="xsd:int"/>
             <xsd:element name="maxBandwidth" type="xsd:int"/>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
</wsdl:types>
```

In order to be able to perform the pSLS negotiation and to obtain the overlay topology, we have defined several database tables used to store the data required by the above mentioned operations. These tables are shortly described next:

- **Overlay_topology** table – it contains data about each EIMS node in the topology, such as the addresses of the web-services available, the IP address, the domain identifier, and the QoS parameters. It is updated by the Inter-domain Overlay Path module at each overlay topology building cycle. It is used by the overlay routing process to build the overlay topology matrix, used in the overlay route searching process.
- **Eims_neighbors** table – stores information about the neighbors for each EIMS node contained in the *overlay_topology* table. It is also updated by the Inter-domain Overlay Path module, at each overlay topology building cycle.
- **Overlay_interdomain_routes** table – is used to store several alternative routes towards a destination overlay node. The number of alternative routes is limited to four. It is managed by the overlay routing process.
- **Local_eims** table – stores information about the local NetSrvMngr@NP such as: IP address, web services ports, domain Id. It is managed by the system administrator.
- **Border_routers** table – stores informations about the local domains border routers. It contains the border routers IP address, and neighbor EIMS@NP reached through this border router. It is managed by the system administrator.
- **Access_networks** table – stores informations about the access networks for the local domain. It contains the access network IP address and the border router IP address. It is managed by the system administrator.
- **Local Eims_neighbors** table - stores information about the eims neighbors for the local domain. It contains information about the border routers used to connect the local domains and the neighbors, border router IP address, web service port addresses, etc. It is managed by the system administrator.
- **Domain_qos_parameters** table – it is used to store global QoS parameters about the domain. It is managed also by the system administrator.

D. *Functionality tests*

This solution was implemented on the test-bed build at our university in the Enthrone project framework [21] [22]. The test-bed consists of three Autonomous Systems, each managed by a Network Service Manager (EIMS@NP). The EIMS@NP managers are implemented using web services technology. Between domains the BGP protocol is used to route the best effort traffic. A Network Manager is used to install the pSLS pipes on network devices. Also the test-bed has a Service Provider EIMS Manager, and the other modules required by the Enthrone system. The connectivity tests involved only the Network Provider managers and Service Provider manager.

The EIMS@SP was used to trigger pSLS subscribe requests, between a Content Provider and one of the available Access Networks, until the resources on the lowest cost path between the chosen source and destination, were exhausted. Then, we triggered additional requests between the same source and destination. These new requests were admitted but the pSLS pipes were built along the next cheapest path between the chosen end points.

Because the test bed is a small one, is difficult to evaluate the performances of the proposed solution for a large number of domains. We have measured how fast a request for getting the neighbors EIMSs from a network domain is served. We have obtained a mean time less than 0.1s per request. If we take for example a topology consisting of 1000 domains then, because we can consider that the total processing time is increasing linearly with the number of domains, the total processing time requires to obtain the overlay topology is about 100s. We can increase it with 50% to take into account that at a large number of domains the local processing time, between two interrogations, could be higher. So, we could consider that for 1000 domains the topology building process takes about 150s, which is an acceptable value. Also, the solution used to build the overlay topology, implies a large number of messages to be exchanged in order to build the topology. Each node should communicate with the other nodes. But the messages exchanged are small, because each of them contains only a few data about the neighbors of the interrogated node. If it have been adopted a link state like protocol to build the topology, then the messages would have been much bigger, in case of large number of domains, so the amount of signaling data in the network would have been bigger too. Also, in our case we don't have convergence problems.

It has not been evaluated till now the time needed to compute several paths towards all the destinations nodes in the overlay topology.

The test bed used is not appropriate to test the scalability for the path finding process performed in the first phase. It was only used to see that the routing table is built correctly, containing several paths towards each destination domain in the topology. Then, several requests for QoS enabled pSLS pipes were triggered. These pipes were built along the first path specified in the routing table. When the resources on this path were exhausted, during the negotiation process, the next route was used for the following pSLS pipe. These tests proved that the solution is able to find QoS enabled pipes, in a multi domain environment.

VI. ESTABLISHMENTCONCLUSION AND FUTURE WORK

It has proposed a simple solution for solving the problem of QoS enabled inter-domain path finding, in the presence of a Network Service Management system, capable of QoS enabled pSLS pipes negotiation.

Because it does not require at a domain the knowledge of other domain resources, it could be accepted by the actual network providers. Another advantage is, that it does not burden a given domain manager with the need of knowing the available traffic trunks of other network domains. Also, by separating the process of path finding from the QoS negotiation, the path searching process does not need to work real time. So we can find several paths in very complex overlay topologies. By simplifying the overlay topology, considering only the domain managers as topology nodes, our solution will work for very complex topologies, being no need for a hierarchical approach.

The solution has the main disadvantage that it does work only in the presence of a QoS negotiation system capable. It is based on this feature to check the QoS constraints along the paths founded in the overlay topology. Another disadvantage is that, it may not find the best QoS enabled path, as could be the case with other solutions.

But it is simple, and is well suited for ENTHRONE Integrated Management System. The solution is also naturally extensible for more sophisticated techniques in QoS capable paths finding.

Further studies and simulations will be done in order to validate this solution for a real network environment. Also, it has been suppose that, because the path finding process could be run offline, and the topology is a simplified one, a non hierarchical solution could be adopted for Internet. Simulations should be done to establish the amount of resources need by such a process.

ACKNOWLEDGMENT

This work was supported by the FP6 project ENTHRONE.

REFERENCES

[1] S.G. Obreja,, E. Borcoci, "Overlay Topology Based Inter-domain QoS Paths Building". AICT apos;08. Fourth Advanced International Conference on Telecommunications, Volume , Issue , 8-13 June 2008 Page(s):64 – 70. Greece, Athens.

[2] ENTHRONE I Deliverable 05 "IMS Architecture Definition and Specification", June 2004.

[3] A. Kourtis, H. Asgari, A. Mehaoua, E. Borcoci, S. Eccles, E. Le Doeuff, P. Bretillon, J. Lauterjung, M. Stiemerling, "Overall Network Architecture", D21 ENTHRONE Deliverable, May 2004.

[4] T.Ahmed - ed. Et al., "End-to-end QoS Signal-ling & Policy-based Management Architectures", ENTHRONE IST Project Public Deliverable D23F, Sep-tember 2005, http://www.enthrone.org.

[5] H. Asgari, ed., et.al., "Specification of protocols, algorithm, and components, the architecture, and design of SLS Management", ENTHRONE IST Project Public Deliverable D24F, July 2005, http://www.enthrone.org

[6] P.Bretillon ed., et. al, ENTHRONE II Deliverable D01: Overall system architecture – version 2, 2007

[7] P.Souto ed., et al, ENTHRONE II Deliverable D03f: EIMS for ENTHRONE 2

[8] E. Borcoci, Ş. Obreja eds., et al, ENTHRONE II Deliverable D18f: Service Management and QoS provisioning.

[9] Project P1008, "Inter-operator interfaces for ensuring end-to-end IP QoS", Deliverable 2, Selected Scenarios and requirements for end-to-end IP QoS man-agement, January 2001.

[10] P.Trimintzios, I.Andrikopoulos, G.Pavlou, P.Flegkas, D. Griffin, P.Georgatsos, D.Goderis, Y.T'Joens, L.Georgiadis, C.Jacquenet, R.Egan, "A Management and Control Architecture for Providing IP Differentiated Ser-vices in MPLS-Based Networks", IEEE Comm. Magazine, May 2001, pp. 80-88.

[11] E.Marilly et. al, "SLAs: A Main Challenge for Next Generation Networks", 2nd European Conference on Universal Multiservice Networks, ECUMN'2002 April 8-10, 2002.

[12] T.Engel, H.Granzer, B.F. Koch, M.Winter, P.Sampatakos I.S. Venieris, H.Hussmann, F.Ricciato, S.Salsano, "AQUILA: Adaptive Resource Control for QoS Using an IP-Based Layered Architecture", IEEE Communications Magazine, January 2003, pp. 46-53. See also http://www-st.inf.tu-dresden.de/aquila/

[13] Fabio L. Verdi, Maurıcio F. Magalhaes "Using Virtualization to Provide Interdomain QoS-enabled Routing", Journal of Networks, April 2007.

[14] Z. Li, P. Mohapatra, and C. Chuah, "Virtual Multi-Homing: On the Feasibility of Combining Overlay Routing with BGP Routing," University of California at Davis Technical Report: CSE-2005-2, 2005.

[15] Z. Wang and J. Crowcroft, "Quality of Service Routing for supporting multimedia applications", IEEE Journal of Selected Areas in Communication (JSAC), 14 (7) (1996), pp. 1228-1234.

[16] D. Eppstein, "Finding k-shortest paths", SIAM Journal on Computing, 28 (2) (1998), pp. 652-673.

[17] D. Griffin, J. Spencer, J. Griem, M. Boucadair, P. Morand, M. Howarth, N. Wang, G. Pavlou, A. Asgari, P. Georgatso, "Interdomain routing through QoS-class planes", Communications Magazine, IEEE, Feb.2007, Volume: 45, Issue: 2, page(s): 88-95.

[18] S.P. Romano, ed., "Resource Management in SLA Networks", D2.3 CADENUS Deliverable, May 2003.

[19] T.Ahmed, A.Asgari, A.Mehaoua, E.Borcoci, L.Berti-iquille, G.Kormentzas "End-to-End QoS Provisioning Through an Integrated Management System for Multimedia Content Delivery", Computer Communication Journal, Volume 30, Issue 3 (February 2007), pages 638-651.

[20] E.Borcoci, A.Asgari, N.Butler, T.Ahmed, A.Mehaoua, G.Kourmentzas, S.Eccles "Service Management for End-to-End QoS Multimedia Content Delivery in Heterogeneous Environment", AICT Conference, Lisbon, Volume, Issue, 17-20 July 2005 Page(s): 46 - 52. Portugal, Lisbon.

[21] T. Ahmed, ed. et al., ENTHRONE II Deliverable D27 "Pilot and services integration and tests", March 2008

[22] ENTHRONE II Deliverable D28 "Trials and evaluation", November 2008

Emulation of Wireless Multi-Hop Topologies with Online Mobility Simulation

Anders Nickelsen
Department of Electronic Systems
Aalborg University, Denmark
Email: an@es.aau.dk

Hans-Peter Schwefel
Department of Electronic Systems
Aalborg University, Denmark
and Forschungszentrum Telekommunikation Wien - FTW
Vienna, Austria
Email: hps@es.aau.dk

Abstract—**Communication in wireless networks is affected by uncontrollable disturbances in the channel. Effects of these disturbances are exacerbated in networks with dynamic topologies and multiple hops. Lack of control of the channel complicates testing applications in such networks as test conditions are hard, or impossible, to reproduce. This paper describes a test-bed to create reproducible test conditions for applications by emulating the wireless links. Emulation is performed by a topology emulator to which end-nodes are connected using wired links. In real-time, the emulator drops or delays packets in traffic between end-nodes. These imposed link properties are based on simulations of node mobility, loss and delay models. Two versions for performing the simulations are described; an offline version in which the mobility traces and link properties are calculated beforehand, and an online version where geographic trajectories are depending on the outcome of the communicating applications. Evaluation confirms that both versions of the test-bed are capable of emulating links in real-time and transparently to upper layer protocols. Additional delays from packet processing and bandwidth limitations introduced by using the emulator are shown meet the transparency requirements, also when the emulator is heavily loaded with packet flows.**

Keywords-topology emulation; real-time; scalability; online simulation;

I. Introduction

Wireless technologies are deployed in increasingly many types of mobile devices. The vast popularity of these mobile devices, and thus the increased availability of wireless technologies, makes applications for the mobile domain of great interest. Evaluation is an integral part of developing such applications. Evaluation can be performed in fully virtual simulation models or in experimental lab or field prototypes. Emulation provides a hybrid of simulation and experimental setups by allowing real applications to operate in environments with simulated link properties. The control of the simulation allows for accurate reproductions of the test conditions to repeat test runs. A disadvantage of using emulation is that real applications operate in real-time and thus enforce the link emulation to run in real-time. This paper describes an extension of the work on a topology emulation tool described in [1].

Simulation tools, such as ns-2 [2], provide a high level of detail in the networking layers. The application under test is, however, typically simplified as the real application cannot easily be used in the simulated environment. Hence, the simulation results represent an application model and not the real-life implementation of the application itself.

Experimental setups can be used to test the real application implementation. However, disturbances in the environment may have a huge impact on the test results. Moreover, the characteristics of these disturbance are not always controllable, making it very difficult to repeat even simple test runs [3].

The objective of this work is to develop an emulation test-bed to evaluate applications in wireless multi-hop topologies. In the test-bed the wireless link is replaced by a wired link. The wired link is less exposed to uncontrollable disturbances and thus more controllable than a wireless link. The properties of a wireless link, such as *packet drops* and *delay*, are then imposed on network traffic on the wired link in a controllable and reproducible manner based on simulation models.

Several types of applications benefit from being tested in an emulation test-bed. Performance of implementations of multi-hop routing protocol can be verified under realistic, yet reproducible, conditions. Other types of applications such as platooning [4] or driving assistance [5] to be deployed in car-to-car scenarios can be tested in the risk free environment of the emulator. In the latter case, the network quality experienced by the application is mapped into changes of node movement or transmission parameters. An example of this is a platooning application that reacts to experiencing a high packet loss rate by slowing down the speed of the car to avoid cars colliding. The described applications illustrate two important aspects of the test-bed. In the first case, a routing application optimizes *network* performance based on the emulated link properties, i.e., affects parameters in layers above the emulator. In the other case, a platooning application optimizes *geographic trajectories* of the nodes based on the link properties, i.e., directly affects the input parameters of the test-bed. This paper describes how the test-bed supports both types of applications.

Several additional requirements must be fulfilled by the test-bed in order to emulate properties of wireless links successfully. Emulating *transparently* as seen from the network application is inherent to create results comparable to those of real experimental setups. Transparency is composed of several requirements. The emulator must operate in real-time to resemble access and transmission performance of a real wireless link, as seen from the applications. It must also present link-layer interfaces to the network layer (and upper layers) similar to a real wireless interface. The emulator should be non-intrusive meaning that as little software as possible should be added to the end-nodes where the applications are deployed. In addition, the test-bed must employ *accurate communication models* in the link simulation to resemble influences from a real environment. Lastly, simple and scalable deployment, use and result processing is required to ensure usability of the emulator. Scalable deployment means that it should be easy to connect enough nodes to the test-bed to create a network of realistic size for the application under test.

The contributions of this paper are: 1) description of the design of the topology emulator, a test-bed capable emulating dynamic multi-hop topologies by changing time-varying link properties in real-time; 2) development of both an offline simulation version for complex models to be simulated before emulation and an online simulation version for simple models of which parameters can be affected during emulation; 3) evaluations of both versions of the topology emulator to illustrate that they are capable of emulating dynamic multi-hop topologies while accomplishing the specified requirements.

II. Related work

Many solutions emulate wireless networks by manipulating link properties of connections. In general, the solutions can be put in two categories; with a *central* architecture or with a *distributed* architecture. The central architecture eases deployment, configuration and control of the emulation process. However, the typical problem with a central approach is performance limitation. Tools with distributed architectures address this limitation by distributing the simulation and emulation tasks onto more nodes in the network. On the other hand, these tools are typically challenged by having to coordinate events between the involved nodes and require an additional piece of software to be installed on every involved node. The latter requires that there exists a version of the emulation software for all platforms used in the application, which is difficult to support.

An extension to ns-2, called ns-2e [6], can emulate simulated link properties from a traditional ns-2 simulation onto real traffic. Through so-called *network objects* real-world traffic is redirected through the simulation. Incompatibilities between real and simulated traffic are handled using *tap agents* that effectively tunnel real traffic inside special simulation packets. In its original form, ns-2e is completely transparent toward end-nodes. However, it cannot guarantee timely delivery of real traffic. This limitation has been addressed in [7]. ns-2e has the ability to simulate dynamic topologies through traditional scripting. However, the number of end-nodes connected to the emulator is limited by the number of available network interface cards on the emulating node. This is a limiting scalability factor.

Another widely used emulation tool is NIST Net [8]. NIST Net is described as 'network-in-a-box' capable of emulating an entire network in a single hop. Similar to ns-2e NIST Net employs a central architecture with a single node to handle all end-node connections, simulation and emulation. This means that it suffers from the same deployment limitations as ns-2e (individual network interface per attached node and the need for context switches in the operating system to handle interrupts for all packets on these interfaces). Also, as indicated by the term 'network-in-a-box', NIST Net emulates entire networks and not links. This means that to use NIST Net to evaluate networking protocols they need to be implemented in the emulator, as is the case when using a simulator. Thus, NIST Net is not transparent to network protocols, as required.

The approach of Seawind [9] is similar to the two described above. Seawind has mostly be applied for wide-area wired and wireless networks.

Recently developed tools with centralized architecture include WNINE [10] and Qomet [11]. WNINE employs a *two-stage emulation* process meaning that it completely separates simulation of the link properties and the emulation. In the simulation process it offers detailed models, but for emulation it relies on dummynet [12], which is a tool for traffic shaping on intermediate nodes in a network. Thus, WNINE also suffers from the scalability limitations. Qomet follow the same concept as WNINE, and evaluation of Qomet only been performed using StarBED [13] and not in a general network scenario.

In contrast to the centralized approach, each attached node in a distributed emulation calculates its own view on the link properties and these views must be synchronized between all nodes. This approach is used in NetEM [14], EMWIN [15], JEmu [16]. Testing real applications with these distributed tools has several challenges. The end-node environment is deployed as a virtual machine and not a native environment. This means that the possibility of running the real application in a deployment environment is reduced. Moreover, the communication overhead of synchronization may affect the performance of the protocols under evaluation. Lastly, and perhaps most importantly, this type of emulator is not transparent in the sense that in each case a (potentially small) piece of code must be running on the attached node.

The tool presented here, called the 'topology emulator', employs a central architecture and thus falls in the category of ns-2e and NIST Net. Compared to NIST Net, the tool emulates individual layer 2 links between nodes instead of layer 3 paths. This is done in order to facilitate testing of networking protocols without having to implement them on a software platform in the involved nodes as well as in the NIST Net emulator. Compared to ns-2e, the architecture of the topology emulator uses one central network switch to aggregate node traffic instead of using a number of network interface cards in the central emulating device. This has the advantage that more nodes can be added without changing the hardware set-up of the emulator. When new nodes are added it is only the internal link models that need to be updated accordingly.

III. Topology emulator

The work presented in this paper builds on the scalable centralized emulator that is transparent to real applications on real end-nodes and meets real-time requirements. This version of the emulator is described in detail in [1] and summarized in the following. The version of the emulator described here extends the features of the original emulator to support what we call *online simulation*. To distinguish to the two versions, we refer to the original emulator as the *offline version* and to the extended emulator as the *online version*, which is explained in the following.

In the offline version, all properties of the network are assumed to be known beforehand. By knowing distances between nodes, the specific link properties (packet drop probability and packet delay) for transmitting packets can be calculated. To calculate such link properties, models of the transmission conditions (propagation conditions, transmission power, coding schemes, etc.) must be specified. All simulation models can be dynamic models. This way geographic trajectories of nodes can change during simulation, as well as transmission parameters such as transmission power. Any parameter can be changed in any model, as long as it is known beforehand. Once the emulation is running, the parameters cannot be affected anymore in the offline version.

In the online version, we introduce online simulation which means that the parameters of the transmission models can be affected during emulation. To facilitate changing the parameters online, a control channel from the applications to the emulator is introduced in the architecture.

Next, the basic architecture of the scalable topology emulator is described. This architecture is designed for both offline and online simulation. Then, the details of the core parts of the offline version are described. Finally, the main implementation choices to realize the online version are described.

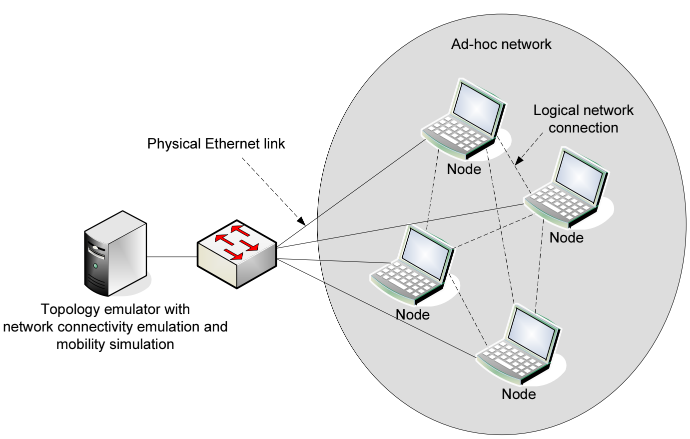


Figure 1. Architecture of topology emulator.

A. Architecture

The network architecture of the topology emulator is illustrated in Figure 1. It consists of one centralized node called the *emulator node* and a central network switch to which all *end-nodes* are connected. End-nodes contain the applications or networking protocols for evaluation. When applications are evaluated in real setups, the wireless networking interface on the end-nodes is used. When the topology emulator is used for evaluation, the wired networking interface must be used to connect to the central network switch. This change of network interface is the only requirement to use the topology emulator for evaluation. The network switch allows for connecting many nodes to the topology emulator and thereby helps ensure the scalability of the test-bed.

The emulator node receives and forwards all frames transmitted between end-nodes. No end-node receives frames before they have been forwarded by the emulator node. This node separation and traffic concentration is obtained by use of 802.11q virtual LAN (VLAN) tagging [17] on the switch. This concentrates all frames from one end-node in one unique virtual LAN per end-node. The VLAN-tag can thereby be used by the emulator node to identify the sources of the frames.

The emulator node creates virtual links between the end-nodes by controlling bridges between the VLANs. The virtual links allow end-nodes to transmit frames to each other only if enabled by the emulator node. By selectively dropping or delaying frames on the virtual links, the emulator node controls the properties of all virtual links between end-nodes. Ultimately, as the emulator node functions as an enhanced switch and forwards layer 2 frames, its existence is transparent by design to any layer 3 protocols (and higher) used on the end-nodes.

The software architecture of the emulator node consists of three processes; a simulation process, an emulation process and a property updating process. The simulation process simulates link properties (packet drop probabilities and delays) for all virtual links and saves them to a trace. The emulation process emulates the links in real-time by deciding if packets should be dropped or not while end-nodes communicate. If not dropped, the emulator determines a delay for each packet and transmits it once the delay expires. The property updating process binds the simulation process and the emulation process together by periodically feeding link properties into the running emulation process.

In Figure 2 the architecture of the offline version of the topology emulator is depicted. In contrast to the offline version, the online version in Figure 3 also contains the mobility simulation and the link simulation as part of the real-time execution. In the following the different parts of the architectures are described in detail.

B. Offline version

In this section, the simulation process, the emulation process and the property updating process are described for the offline version. The emulation process is the same for the offline and

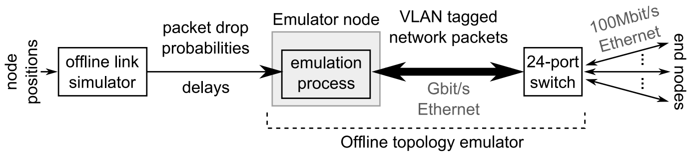


Figure 2. Emulation with offline simulation.

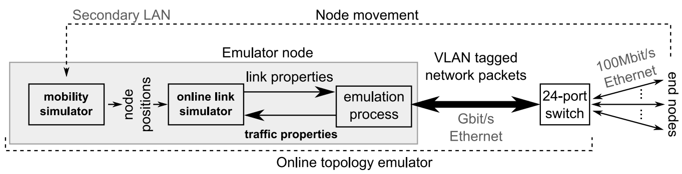


Figure 3. Emulation with online simulation.

online version and therefore it is only described in this section.

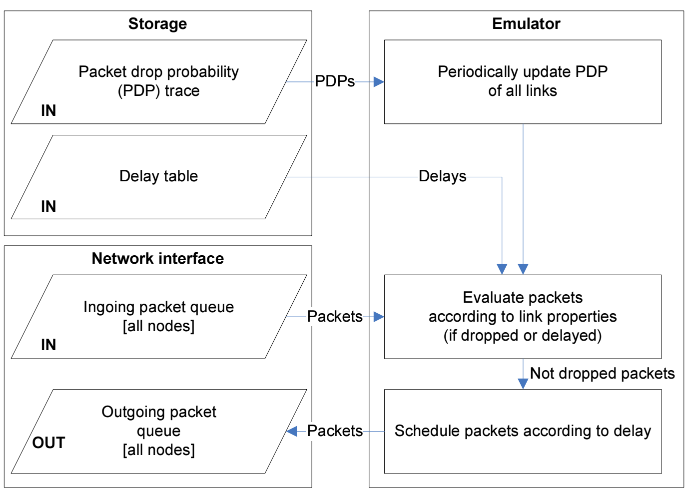


Figure 4. Emulation process

1) Simulation process: In the offline version simulation is carried out before the emulator is started.

The simulation process is a function that outputs packet drop probability and packet transmission delay for all virtual links in time-slices. As input it takes a trace of node positions, a transmission model and input parameters for the transmission model. Two examples of transmission models and parameters are described below. These are also the models used to verify the performance of the two emulator versions. This verification is described further in section IV.

One model is a *simple* unit-disk model that determines packet drop probability of the link between nodes i and j at time t by use of a threshold D on distance $d_{i,j}$ between the nodes:

$$PDP_{i,j}(t) = \begin{cases} 0 & \text{if } d_{i,j}(t) < D \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

Here the delay is constant on all links, which would represent simply a medium access delay where the channel is assumed error free and without other contenders when a packet is transmitted.

To use this model in the simulation process with a trace of node positions as input, the model parameters needed are the distance threshold and the constant delay.

More realistic (and thus more complex) transmission models than the unit-disk model are supported by the simulation process. In [18], a stochastic model is used which is detailed in the following.

To model packet drop probability, channel properties, physical layer properties and link layer properties are used to relax some of the simplifying assumptions of the simple model. Rappaport's shadowing model [19] as seen in (2) is used to model attenuation of transmitted power to received power (α is the path-loss coefficient, σ is the shadowing parameter which is the standard deviation of the zero-centered Gaussian random variable X_σ).

$$P_r(d) = P_t - 10 \cdot \alpha \cdot log_{10}(d) + X_\sigma \quad (2)$$

The mapping between received power and bit error rate in the physical layer is modeled by Mangolds OFDM model [20], assuming use of IEEE 802.11a. To calculate a frame error rate from the bit error rate the assumption of independent, identically distributed losses is used, leading the following expression (where L_{frame} is size of frame in bytes):

$$FER = 1 - (1 - BER)^{8 \cdot L_{\text{frame}}} \quad (3)$$

The frame error rate is then mapped one-to-one into a packet drop probability in this model. To model the delay, we use the Bianchi IEEE 802.11 DCF model [21], assuming that the only causes for frame losses are collisions due to other nodes transmitting and thus the only factor in delay is the time spent backing off from the channel. Propagation delay is considered as a constant and independent of distance. To model the number of nodes contending for a channel, we model how many nodes are likely to transmit packets within a time-slice by using a threshold on the packet drop probability for all virtual links. If a link between two nodes has a packet drop probability below a threshold P, we consider the two nodes as being *neighbors* and consider the transmitted packets from them probable to collide. The more neighbors a node has, the more likely it is for packets transmitted by this node to have a high delay.

In this paper the latter complex model is used for the evaluations of the offline version. For comparison of the offline and online version, the simple link model is used in order to keep processing power needs of the emulator node low (and to avoid having to performance optimize the implementation of the complex model).

When performing offline simulation the distances can be calculated from node positions which are input to the simulation process over time t as (x, y)-coordinates. These positions can be based on realizations of either recorded traces of real movement, deterministic paths that simulated nodes follow or based on simulations of stochastic models such as random walk or random way-point [22].

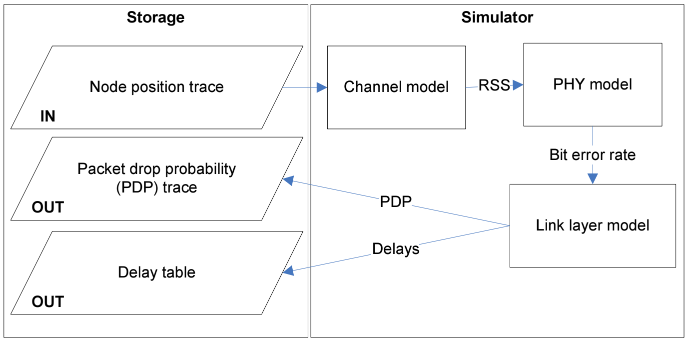


Figure 5. Offline simulation process

Packet drop probability of a link is calculated based on distances between nodes and the transmission models of the channel, the physical layer and link layer as shown in Figure 5.

The output of the calculations is a trace of packet drop probabilities over time $p_{i,j}(t)$ on every link (between nodes i and j) in the topology. Note that the symmetry of the probabilities on a link depends on the simulation model as the topology emulator supports asymmetrical links where upstream and downstream links have unequal properties.

The delays are calculated as a static table of inverse cumulative distribution functions. The table is used primarily to enable fast lookup of delays as it need to be determined for each packet when emulating. As the traffic patterns of the nodes are not known prior to the emulation, a parameterized family of delay distributions for $K = 1, 2, ..., n$ (number of simultaneously transmitting nodes) is used instead of just samples of delays. n is the total number of nodes in a simulation and is scenario-specific. The reason several delay distributions are used is the fact that the distribution of the delay depends on the number of nodes able to transmit in a channel. Therefore we generate a delay distribution for each possible number of next-hop neighbors in a network. When nodes move around or the transmission conditions change, the number of next-hop neighbors will change and the real-time emulator will access the corresponding distribution in the table. This also means that the table containing the delays is generated once during simulation and not continuously updated during emulation.

Using the simple model with constant delay for all numbers of next-hop neighbors the delay table is a column vector of height $n-1$. In the complex model with stochastic delay the table is a $(n-1) \times r$ matrix, where r is the chosen discretization of the inverse CDF in the table.

Based on the probability threshold P, the number of next-hop neighbors $n_i(t)$ of node i at time t is calculated from $p_{i,j}(t)$ for all t. This is used as index to find the appropriate delay distribution during emulation.

All simulation models must compensate for the fact that all calculations are performed before running the emulation. For the delay, this means that the patterns of generated traffic by applications connected to the emulator must match the traffic models used in the simulation models. As an example, the Bianchi-model assumes a saturated channel, which may not always be the case for the applications. Also, in the offline version, it is not possible for the applications to dynamically change communication parameters, such as modulation scheme, when emulation is running. If in need of such changes, they must be implemented in the simulation models when simulating.

2) Property updating process: In the offline version the trace of link properties is saved to a file in the simulation process. To be able to use the trace-file for emulation, the *property updating process* reads the file and updates arrays accordingly which are then read by the emulation process when determining packet drop probabilities and number of next-hop neighbors. The static delay table is only loaded once. The entries in the trace are time stamped relative to the beginning of the simulation and ordered ascending. When the real-time clock reaches the next time-stamp, the next entry in the file is loaded into the emulation process. The property updating process supports update frequencies up to 10Hz.

3) Emulation process: All nodes are physically connected to a Cisco Catalyst 2950T VLAN-aware 24-port switch. Two of the ports have a capacity of 1Gbit/s traffic whereas the remaining ports support 100Mbit/s. One of the gigabit ports is used in trunking mode, such that all traffic received on other ports is concentrated in this port. The emulator node is then connected to the trunk port. This node has an Intel Core 2 Duo 1.86GHz processor and has Linux kernel v2.6.18 installed in our setup.

Packets received on the emulator node are filtered twice before reaching the emulation process; on layer 2 by *ethernet bridging* controlled by *ebtables* and on layer 3 by *netfilter* controlled by *iptables*. The layer 2 bridging is necessary to prevent transmission of link layer datagrams, e.g from ARP, when there is no virtual link. If emulation was only enforced using netfilter, link layer datagrams would not be filtered. Netfilter is used to deliver packets to the emulation process.

The details of the emulation process resemble NIST Net, however, as the packet flow using the VLANs was not easily integrated into the existing solution, a new emulation process was developed.

On layer 2, the packet control uses a binary value $l_{i,j}(t)$ indicating if a link exists or not between source and destination of a packet identified by the VLAN tag. If $l_{i,j}(t) = 1$ then there is a link and the packets are forwarded to layer 3. If $l_{i,j}(t) = 0$ then there is no link and the packet is dropped. $l_{i,j}(t)$ is calculated as

$$l_{i,j}(t) = \begin{cases} 1 & \text{if } p_{i,j}(t) < P \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

On layer 3, the packets are forwarded to the emulation process. For each uni-directional link between two nodes, parameters are stored in tables representing packet drop probability, the number of next-hop neighbors, and packet delay distributions. To determine if a packet is dropped the emulation process draws a uniform random number in [0,1] and compares it to $p_{i,j}(t)$. The process also determines the delay by using the number of next-hop neighbors of the source node i, $n_i(t)$, and the static delay table. To determine a delay, $n_i(t)$ is used to find the appropriate distribution (row in the table). A uniformly distributed random number is drawn and used as index in the tabularized form of the inverse CDF which then indicates the delay of that particular packet. Once the delay has been determined, the packet is scheduled for delayed transmission.

The packet scheduler checks for and transmits packets every $122\mu s$. In order to achieve such a high time resolution, the Linux Real-Time Clock chip (RTC) was used to trigger a scheduler in the emulation process. This clock supports triggering processes at frequencies up to 8192Hz $(1/8921Hz = 122\mu s)$, which is much higher than normal timer resolution of Linux of 10-100Hz. Once triggered, the emulation process advances a ring buffer containing a list of packets to be transmitted in the current time slot and transmits all the packets in the list.

C. Online version

For the online version the simulation is performed in real-time during the emulation. An overview of this process related to emulation is given in Figure 3. The main difference to the offline version is that here parameters of the simulation models can be changed online. This means that instead of having all inputs to the simulation process being known and specified beforehand, these properties can be actively updated during the emulation. This directly enables application to affect mobility parameters, namely to be able to affect the position of a node as a consequence of the experienced network quality. With the online version, this becomes possible as all input parameters to the simulation process can be updated, including the node movement.

As illustrated in Figure 3, the online version requires the simulation process to be deployed on the emulator node. In turn, if applications affect the mobility of the nodes, updated position information must be communicated to the link simulation process. As the example applications in the vehicular scenarios do not manipulate (x,y)-coordinates but rather node physics (such as speeder or brakes), there must exist a function capable of transforming application output to link simulation input. In the car scenario this function would transform braking and speeding information into updated (x,y)-coordinates which would be loaded into the link simulator. To handle this transformation, an additional mobility simulation entity is needed in the network between end-nodes and the link emulator.

Online link simulation during emulation can be performed on several levels; an event-based level (where link properties are calculated per packet) or periodically (where link properties are calculated per time-slice) as in the offline version. The event-based level is very precise as the networking environment is adapted according to each packet. The periodic level may be less precise as it only represents an average of properties over the specific time-interval. On the other hand, simulating complex models per packet can become very processing-intensive when a large amount of packets is sent through the emulator process. Simulating periodically is not influenced by the incoming packet rate and may therefore be more suitable for more complex models as high packet rates can still be supported. In this work, the online simulation is performed periodically as the event-based level is very processing-intensive and not necessary to illustrate support of online simulation. As described in section III-B, a simple simulation model was developed to reduce implementation complexity and processing-intensity. The simple model is sufficient as main purpose of the model is to evaluate the performance of the online simulator compared to the offline simulator.

The connection between the online simulation process and the emulation process is quite similar to the offline version. In the online version the link properties are transmitted from the simulation process to the updating process using a socket. This structure has been chosen as it enables the simulation process to be located on a secondary, resourceful computer. To be able to use the properties for emulation, the *property updating process* for the online version reads the incoming data on the socket and updates the emulator arrays. The arrays are updated in an event-driven manner whenever new data is received by the updating process. As in the offline version, this updating process supports update frequencies up to 10Hz.

IV. Evaluation

The topology emulator has been evaluated on several levels; 1) evaluating that the emulator is capable of emulating dynamic multi-hop topologies and 2) verifying that the performance requirements are met. Results from both evaluations are presented and discussed in the following. As the emulation process is the same for both offline and online simulation, the emulation capability has only been evaluated once by using the offline version. Meeting performance requirements is a different task depending on simulation version and therefore has been evaluated for both offline and online version.

A. Functionality evaluation

To evaluate the functionality of the emulator, a scenario is specified where mobile nodes create a dynamic topology in which end-to-end communication is supported by ad-hoc routing over multiple hops. This scenario illustrates the intended use of the emulator, namely to test prototypes of networking algorithms such as routing or addressing schemes or even transport or service protocols for highly dynamic networks.

The movement of the nodes in the scenario is illustrated in Figure 6. In the scenario, the R2 node moves out of range of the destination node and is replaced in the end-to-end path by R1. Throughout the scenario packets are dropped and delayed on the links according to the simulated parameters.

The objective of the scenario is to illustrate how the ad-hoc routing algorithm is performing in a dynamic multi-hop topology. The link properties model an IEEE 802.11a wireless link incorporating shadowing and fading in the channel, as previously described. The parameters of the physical model are set to $\alpha = 2$ and $\sigma = 4$. For the IEEE 802.11a link the background noise is set to -100 dBm, the modulation scheme to 64-QAM, transmission power to 100 mW, L_{frame} to 1500 bytes and the number of retransmissions in the medium access to 7. The specified movement is used as input to simulate

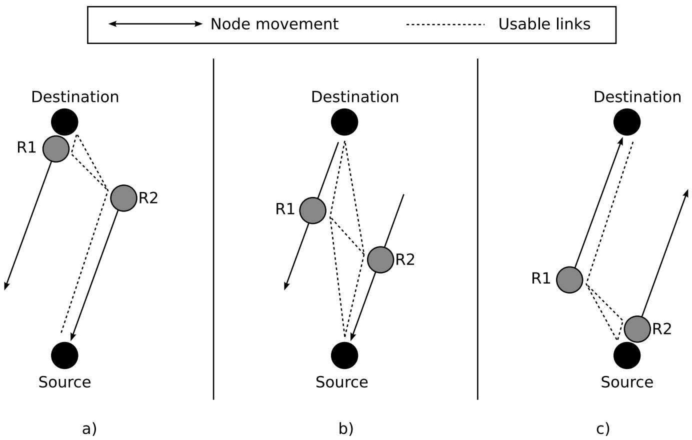


Figure 6. Scenario with relay nodes R1 and R2 move periodically back and forth.

PDP and delay. The PDP of the links is illustrated in Figure 7 as the probability varying over time on each link in the network topology. A 3-hop path is always present (R2-R1-Destination). Also a 2-hop path is always present, however it changes over time. This effectively illustrates the dynamic multi-hop topology. The trace of simulated PDP and the delay table are then used as input for the emulation process to impose the topology information onto real traffic. *ping* is set to continuously ping from source to destination while OLSR [23] is used as ad-hoc routing algorithm between the nodes.

The resulting traffic recorded by the emulator is illustrated in Figure 8. From the figure, we see that the emulator is capable of emulating a dynamically changing multi-hop topology as seen from the end-nodes. This is seen as the flow of packets is redirected to use the available links when the currently used link becomes unavailable. Moreover, the packets sent using the R1-R2 link clearly demonstrate the multi-hop emulation capability.

B. Performance evaluation

As previously described, the emulator must appear transparent to end-nodes meaning that lower network bandwidth and higher link delay (besides the emulated delay) compared to a wireless environment is not tolerated. Evaluations of bandwidth and service time have thus been performed to ensure that the topology emulator meets these requirements.

Bandwidth limitations in a network occur at processing or communication bottlenecks. In the topology emulator, two

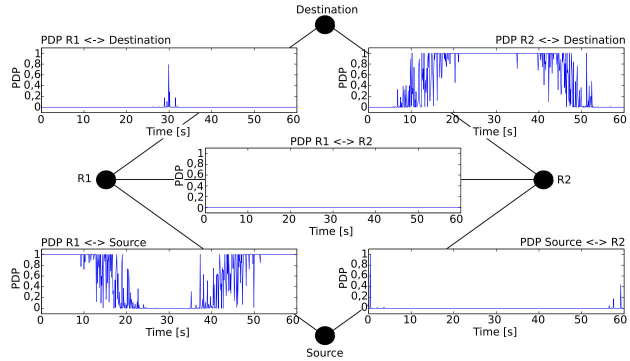


Figure 7. Packet drop probability (PDP) trace on links. The path changes from using *R2-Destination* to *Source-R1*.

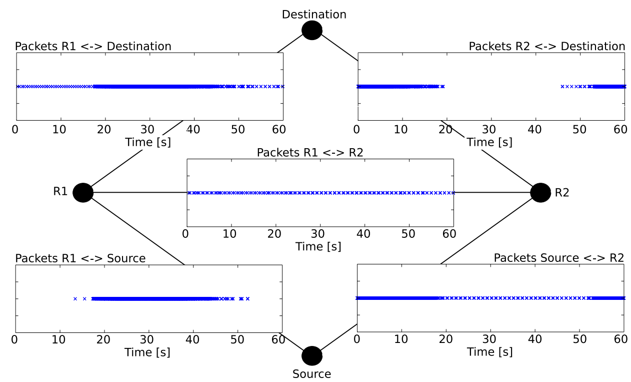


Figure 8. Traffic on links during emulation shows that different available links are used.

such potential bottlenecks exist; in the switch and in the emulator node. As the complexity and processing need in the emulator node is far greater than that of the switch, the emulator node is considered the significant bandwidth bottleneck in the setup. Hence, only the bandwidth capabilities of the emulator node were evaluated.

To evaluate available bandwidth of the emulator node, the traffic generator D-ITG [24] is used. By use of 7 nodes in a fully connected emulated topology, each sending and receiving streams of 100MBit/s asynchronously to other nodes, the emulator node was heavily loaded. D-ITG is also capable of recording the received bandwidth on the nodes which should amount to 100Mbit/s per node deducting a small overhead percentage from transport and network layers. In total this amounts to 700Mbit/s theoretical load on the emulator (experiments showed the real max to be 644Mbit/s).

Considering that the expected maximum throughput of IEEE 802.11a in a real wireless channel is 54Mbit/s, the emulator node is capable of supporting 12 separate channels within the 644Mbit/s limit. This limitation of 12 channels is equal to 24 connected, fully loading nodes in separated networks of only two nodes and is in effect only when the nodes experience a traffic- and noise-free channel.

The measured bandwidth capability of the emulator is considered acceptable due to the following reasons. First, currently it is only possible to connect 20 nodes to the switch. Second, for these 20 nodes to fully load the emulator node on links between them with up to 54Mbit/s, the simulation scenario would need to produce 10 separate channels with only two nodes each and these channels would need to be free of any external noise. This is of course a possible situation when testing the wireless application, although not considered very likely.

Service time of the entire topology emulator is also evaluated. Service time is defined to be the time it takes from a packet is sent from one end-node until it is received at another end-node. This includes transmission time from one end-node to the switch, between switch and emulator node both ways, and finally between switch and the other end-node. To be able to use the emulator to emulate any delay transparently, the service time must not exceed the time it takes to send a packet between two nodes on a real wireless link. The Linux network tool `ping` was used to measure the service time for a link between two nodes by sending out packet probes. `ping` measures the round-trip time when sending and acknowledging a packet. To measure the delay between two nodes, half of the round-trip time was used.

The evaluation is performed on a emulated link (with emulated delay = 0) and compared to both a direct wired link and measurements of an IEEE 802.11a wireless link. This is done to establish if the maximum service time of the topology emulator is less than smallest delays expected on a real wireless link. As the service time includes processing time of a packet in the switch and in the emulator node, and this time is dependent on the packet size, several packet sizes are used. Layer 3 packets of sizes 0-5000 bytes, with Ethernet maximum frame size of 1500 bytes payload, are used. Having a maximum layer 2 frame size smaller than the maximum layer 3 packet size will result in fragmentation, which in turn will result in more frames to traverse the topology emulator, which ultimately will result in longer service times. The measurement of the wireless delays was done using two Linux-equipped laptops with IEEE 802.11a network interface cars in ad-hoc mode standing next to each other. The measurement of the wired link used the same laptops, but connected through a single cross-over Ethernet cable. The delays were measured using `ping` and in the wireless case multiple times during a week. The results of all measurements are illustrated in Figure 9. Here it is illustrated that transmitting a packet through the entire topology emulator uses approx. $250\mu s$ more than when transmitting using a direct wired link. In addition, the figure shows that the excess $250\mu s$ are well below the transmission times of the tested wireless link. This means that the service time of the topology emulator is acceptable as it is below the values of the tested IEEE 802.11a link.

A comparison between the service times of the offline and the online version was also performed. This was done to establish if the transparency requirement is met in the online version as well. As the simulation process consumes

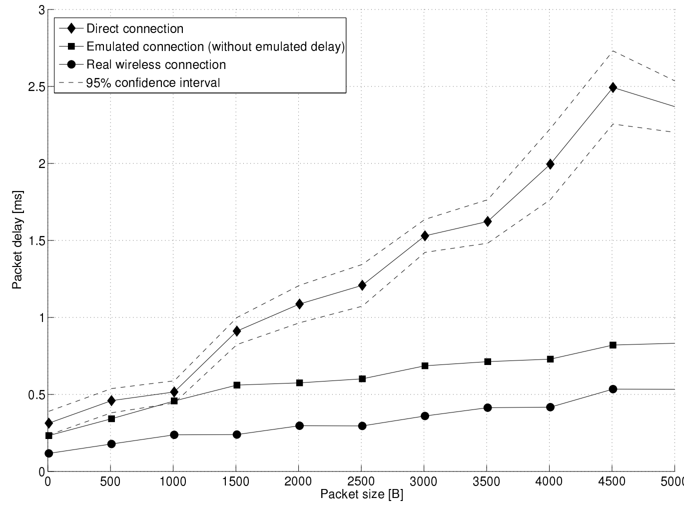


Figure 9. Service time from various packet sizes on: A real wireless IEEE 802.11a link, a direct emulated link (with delay = 0) and a direct wired Ethernet link. The confidence interval is shown for the wireless measurements only, as the variations in the remaining measurements are insignificant.

processing power besides the emulation process, it is important to confirm that the performance of the online version is not different from the offline version.

The evaluation was performed during the bandwidth test described previously using *ping* to probe the delay from one node to another while loading the emulator process heavily. The emulator was loaded traffic ranging from 0 to 700 Mbit/s and *ping* probes of size 1500 bytes measured delay 30 times during the load period. From these measurements the mean one-way delay was calculated. The offline and the online versions were set up using equal simulation models and equal property update rates. The simulated link model was using equal property update frequencies at maximum (10Hz).

The results are shown in Figure 10 and Figure 11. It was possible to measure the delay between load values of 0 to 500 Mbit/s. Above 500 Mbit/s no `ping`-packets were acknowledged within the measuring period. As a consequence load values above 500Mbit/s are not depicted.

As illustrated in Figure 10 the service time of both versions are well below 1 ms, which can be read from Figure 9 to be sufficient to be transparent for packet of 1500 bytes. In Figure 11 it is seen that the service time starts to vary significantly more and increase once the emulation process is loaded with more than 350Mbit/s. This is shown to be the case for both versions. This means that the emulator cannot reliably be considered transparent toward end-nodes above 350Mbit/s, which is then considered the bandwidth performance limit. If the simulated delay is above the service time of 30ms seen in Figure 11, then an algorithm could be applied to take the service time (that changes with the load) into account when delaying packets. As this is out of the scope of this work, the bandwidth performance limit is kept at 350Mbit/s.

The increased service time is most likely due to overloading of the CPU in the emulator node causing buffering on either

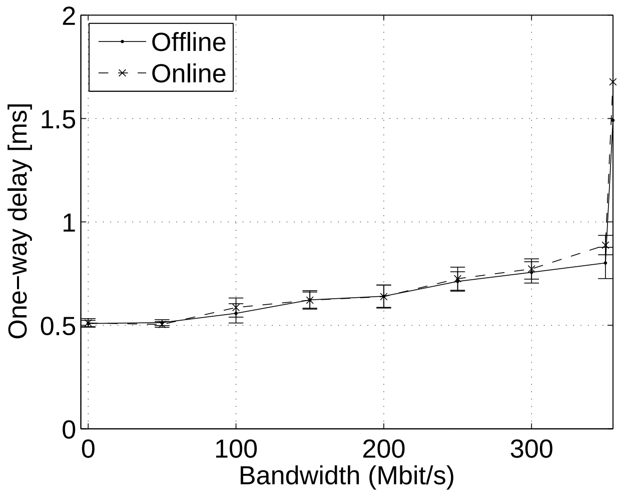


Figure 10. Service delay at different bandwidth (0-350 Mb/s)

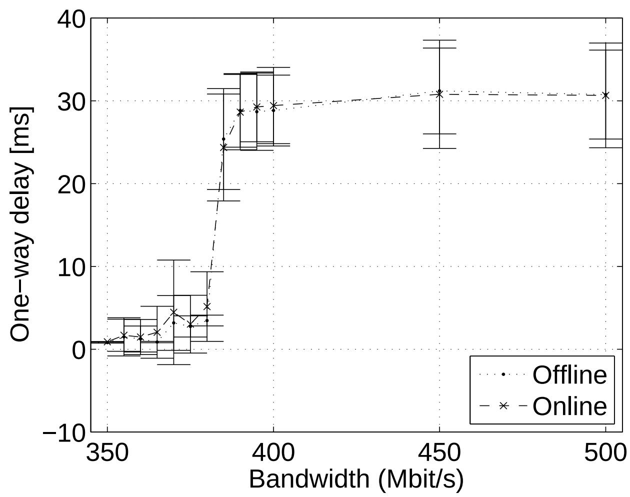


Figure 11. Service delay at different bandwidth (350-500 Mb/s). Continuation of Figure 10 with increased y-scale.

the networking interface card or in the operating system. One immediate solution to this issue would be to distribute the emulation process into two threads working on incoming packets using both cores of the processor.

V. Extensions

Information about node positions or link quality is not communicated to end-nodes. To distribute this information to the end-nodes would require an additional control channel. Such a control channel could be deployed using the existing connections between end-nodes and emulator node. In doing so, the resources consumed by the control traffic on the links should be accounted for in the simulation models. An alternative solution to reusing connections is to deploy a separate network, which would then require additional networking interfaces in all nodes, but would not interfere with the application traffic on the emulated links. Moreover, only packet drop probability and delay are calculated, meaning that the upper layers do not have access to traditional link layer information such as received signal strength indication (RSSI). The setup requires all upper layer technologies to be independent of the link layer and the physical layer, which is a reasonable assumption to test many distributed applications. Approaches to actively deliver such information through an emulated virtual network interface on the end-node are being investigated. Transporting such information from the emulator to the end-nodes requires test-bed specific software to run on the end-nodes during emulation.

VI. Conclusion and Future Work

In this paper, we presented a new network emulation tool capable of emulating dynamic multi-hop topologies. This is especially important when developing wireless applications, as field tests of such applications become cumbersome or even impossible to reproduce. As an advance to existing tools, the topology emulator features real-time emulation of dynamically changing multi-hop topologies that are resulting from node movement in a pre-specified scenario. Moreover, the architecture of the topology emulator is designed to be scalable and modular to facilitate extensions without any modification to the deployed version.

The functionality is divided into two parts; a simulation part and an emulation part. The simulation part simulates a complex wireless network from node movement resulting in packet drop probabilities and delays on each link between nodes. The emulation part imposes these properties in real-time to received packets on a central emulator node. End-nodes, that are usually in the wireless domain, are connected to the emulator node via wired links through a central switch. All frames sent from end-nodes are forwarded to the emulator node and based on the link property traces the emulation part decides if frames should be forwarded further. If so, a delay is determined and the frames are scheduled for transmission to the receiving end-node. This two-part model has been deployed in two different execution environments; offline simulation and online simulation. The offline version simulates the link properties prior to any emulation whereas the online version simulates the properties in real-time during emulation. In the online version it is thereby possible to change the parameters of the simulation model to reflect changes in the environment and allow end-nodes to affect their own positions based on the experienced network properties or on the content of the communicated messages. By designing the connecting point as a switch, the network architecture allows for up to 20 real end-nodes to be connected to the emulator. Evaluations of the functionality and the performance of the emulator shows that it is capable of emulating dynamically changing topology properties toward all connected end-nodes. The results also show that the end-nodes do not experience limitations in bandwidth or longer transmission delays when using the emulator. This has proved valid for both the offline and the online emulator. The maximum capacity of the emulator was determined to be 350 Mbit/s. This limit has shown sufficient to support up to 14 nodes in WLAN scenarios using at maximum 54 Mbit/s per channel.

Future work includes optimization of the performance of the online simulation process. Approaches to deliver simulated link information, such as RSSI-values, layer 2 packet loss rates or even position coordinates, to end-nodes are currently under investigation. An interesting investigation is to establish the maximum possible complexity of the simulation model in the online version. As highly complex model requires many resources during simulation, the processing power remaining from the emulation process can be used to determine the maximum level of complexity for calculating link properties.

Acknowledgments

This work was partially supported by the EU IST FP6 project 'HIghly DEpendable ip-based NETworks and Services – HIDENETS', see www.hidenets.aau.dk and the EU ICT FP7 project 'Open Pervasive Environments for iNteractive migratory services – OPEN', see www.ict-open.eu. The Telecommunications Research Center Vienna (ftw.) is supported by the Austrian Government and by the City of Vienna within the competence center program COMET.

References

[1] A. Nickelsen, M. Jensen, E. Matthiesen, and H. Schwefel, "Scalable emulation of dynamic multi-hop topologies," in *Proceedings of the 4th International Conference on Wireless and Mobile Communications (ICWMC 2008)*, 2008.

[2] S. McCanne and S. Floyd, "Network simulator ns-2." The Vint project, available for download at http://www.isi.edu/nsnam/ns, May 6, 2009.

[3] H. Waeselynck, Z. Micskei, M. Nguyen, and N. Riviere, "Mobile Systems from a Validation Perspective: a Case Study," *Parallel and Distributed Computing, 2007. ISPDC'07. Sixth International Symposium on*, pp. 14–14, 2007.

[4] T. Tank and J. Linnartz, "Vehicle-to-vehicle communications for AVCS platooning," *Vehicular Technology, IEEE Transactions on*, vol. 46, no. 2, pp. 528–536, 1997.

[5] R. Bishop, R. Consulting, and M. Granite, "A survey of intelligent vehicle applications worldwide," in *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pp. 25–30, 2000.

[6] K. Fall, "Network emulation in the VINT/NS simulator," in *Computers and Communications, 1999. Proceedings. IEEE International Symposium on*, pp. 244–250, 1999.

[7] D. Mahrenholz and S. Ivanov, "Real-Time Network Emulation with ns-2," in *Proceedings of the The 8-thIEEE International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2004)*, 2004.

[8] M. Carson and D. Santay, "NIST Net–A Linux-based Network Emulation Tool," *Computer Communication Review*, vol. 33, no. 3, pp. 111–126, 2003.

[9] M. Kojo, A. Gurtov, J. Manner, P. Sarolahti, T. Alanko, and K. Raatikainen, "Seawind: a Wireless Network Emulator. In proceeding of 11th GI," *ITG Conference on Meaurement, Modelling and Analysis (MMB 2001)*, pp. 151–166, 2001.

[10] T. Perennou, E. Conchon, L. Dairaine, and M. Diaz, "Two-stage wireless network emulation," in *Proceedings of the Workshop on Challenges of Mobility held in conjunction with 18th IFIP World Computer Congress (WCC)*, pp. 57–66, Springer, 2004.

[11] R. Beuran, L. Nguyen, K. Latt, J. Nakata, and Y. Shinoda, "QOMET: A Versatile WLAN Emulator," in *Proceeding of the 21st International Conference on Advanced Information Networking and Applications*, 2007.

[12] L. Rizzo, "Dummynet FreeBSD network emulator." http://info.iet.unipi.it/~luigi/ip_dummynet, May 6, 2009.

[13] J. Nakata, S. Uda, R. Beuran, K. Masui, T. Miyachi, Y. Tan, K. Chinen, and Y. Shinoda, "StarBED2: Testbed for Networked Sensing Systems," in *Networked Sensing Systems, 2007. INSS'07. Fourth International Conference on*, pp. 142–145, 2007.

[14] S. Hemminger, "Network Emulation with NetEm," in *Proceedings of Linux Conf Au 2005*, 2005.

[15] P. Zheng and L. Ni, "EMWIN:: emulating a mobile wireless network using a wired network," *Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*, pp. 64–71, 2002.

[16] J. Flynn, H. Tewari, and D. O'Mahony, "A Real-Time Emulation System for Ad Hoc Networks," *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference*, 2002.

[17] D. McPherson and B. Dykes, "VLAN Aggregation for Efficient IP Address Allocation." RFC 3069 (Informational), Feb. 2001.

[18] A. Nickelsen and M. Jensen, "Evaluation of routing dependability in manets using a topology emulator." Master Thesis, 2007.

[19] T. Rappaport, *Wireless communications*. Prentice Hall PTR Upper Saddle River, NJ, 2002.

[20] S. Mangold, S. Choi, and N. Esseling, "An Error Model for Radio Transmissions of Wireless LANs at 5GHz," in *Proc. Aachen Symposium*, pp. 209–214, 2001.

[21] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 3, pp. 535–547, 2000.

[22] C. Bettstetter, H. Hartenstein, and X. Pérez-Costa, "Stochastic Properties of the Random Waypoint Mobility Model," *Wireless Networks*, vol. 10, no. 5, pp. 555–567, 2004.

[23] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)." RFC 3626 (Experimental), Oct. 2003.

[24] S. Avallone, A. Pescape, and G. Ventre, "Distributed internet traffic generator (D-ITG): analysis and experimentation over heterogeneous networks," *ICNP 2003 poster Proceedings, International Conference on Network Protocols, Atlanta, Georgia*, 2003.

Deducing a User's State of Mind from Analysis of the Pictographic Characters and Emoticons used in Mobile Phone Emails for Personal Content Delivery Services

Kazumasa TAKAMI† Ryo YAMASHITA† Kenji TANI†
Yoshikazu HONMA† and Shinichiro GOTO‡
†Faculty of Engineering, Soka University
‡NTT Information Sharing Platform Laboratories, NTT Corporation
†k_takami@soka.ac.jp ‡goto.shinichiro@lab.ntt.co.jp

Abstract - *As the ubiquitous environment is taking root, there are calls for services that deliver content appropriate for the individual user's personal interests and preferences. However, it is difficult to deduce the ever-changing preferences of people who live in a complicated society. In this paper, we focus on mobile phones, whose users are growing in number and which offer many sophisticated functions besides the ability to talk. We propose a method of deducing the state of mind of the user by analyzing the pictographic characters and emoticons used in his or her emails. Moreover, we have proposed a method of selecting an appropriate piece of music based on a music type, which is represented by the "number of chords", "sound strength", and "melody pattern" in a piece of music. We have developed the algorithm to deduce the user's state of mind from an email and applied the algorithm to the selection of music, which is considered to be close related to people's feelings.*

Keywords; *content delivery service; interests and preferences; deduction of state of mind; mobile phone email; pictographic character;*

1. Introduction

As the ubiquitous society develops, the demand for personalized services is growing. For example, there is a Web mail service that analyzes the text of each email received, and delivers advertisements related to the words contained in that email [2]. As terrestrial digital broadcasting and one-segment TV services become widespread and as the memory capacity of mobile phones increases, people expect to see personalized video or music delivery services.

To provide personalized services, it is necessary to capture the preferences and dynamic state (physical/ mental/ emotional) of each user, and to determine the type of service to provide in light of the user's state so captured. There have been a variety of studies that address these needs [3][4][5]. As regards capturing the user's state, methods of determining the user's current location using GPS [4][5], and of capturing the user's state of mind by text mining have been proposed [6][7][8]. However, since text mining requires a vast computing time and vast resources, it is unfit for capturing the ever-changing state of mind of the user in real time.

By March 2008, the number of mobile phone users exceeded one hundred million and is still growing in Japan [9]. The mobile phone has evolved into a sophisticated mobile information terminal, capable of email, Web access, credit card transactions, pre-paid card functions, GPS-based navigation, functioning as a digital camera and so on, in addition to voice communication. This means that a mobile phone is a comprehensive treasure trove of the behavior and preference information of its user. In particular, more than 80% of mobile phone users use pictographic characters in order to convey a variety of emotional concepts that are difficult to express in text [10][11]. Recently, a progress has been made to standardize pictographic characters among different mobile phone providers. Mobile phone emails have the following characteristics:

- There is a strict restriction on the number of characters that can be included in each email, so most emails are short, making it relatively easy to analyze them. This means that mobile phone emails can be analyzed at low cost.
- The style of writing is relatively close to spoken language. A wide assortment of means of expression is used, such as symbols, pictographic characters, and emoticons. These make it easy for the user to write an email that directly reflects how he or she feels at the moment.

We are studying how to deliver content appropriate for the user's current state of mind by analyzing his or her interests and preferences in daily life [1]. In particular, this paper proposes a method of deducing the user's state of mind by analyzing facial pictographic characters and emoticons contained in emails exchanged using a mobile phone. We have chosen to focus on music because user's feeling is the major factor in the selection of music. We have chosen to use the "number of chords", "sound strength", and "melody pattern" as music elements that characterize a piece of music. Based on these elements, we define the relationship between a piece of music and a user's feeling. We also propose a method of searching for an appropriate piece of music. Section 2 presents the information delivery system assumed, and the service it provides. This system and service information has been used to identify research topics. Section 3 proposes solutions for these topics. Section 4 describes the prototype system developed to evaluate the proposed algorithm, the experiments conducted and the evaluation results. It also describes the evaluation of the algorithm by having a group of students use the system, and the evaluation of the music selection method. Finally, Section 5 presents the conclusions and issues for future study.

2. Content delivery system that involves the analysis of mobile phone emails

This section describes examples of email that reveal the user's state of mind and interests, and example of use of the information content delivery service. Then, the research topics are addressed.

2.1. Emails containing pictographic characters and emoticons to express interests and state of minds

The words and pictographic characters and emoticons in emails reveal the user's interests and preferences, and the user's expressions (choice of words) and types of pictographic characters and emoticons reveal the user's current state of mind and the extent of the user's fatigue. An example of an email that reveals the current state of mind and interests is shown in Figure 1. The idea, "The Japanese team don't seem to be able to score", can be expressed in different ways, as shown by the two emails in Figure 2. Email (a) shows that the user is irritated while Email (b) shows that the user is discouraged. So, the same sentence can covey a different state of mind depending on the pictographic character used.

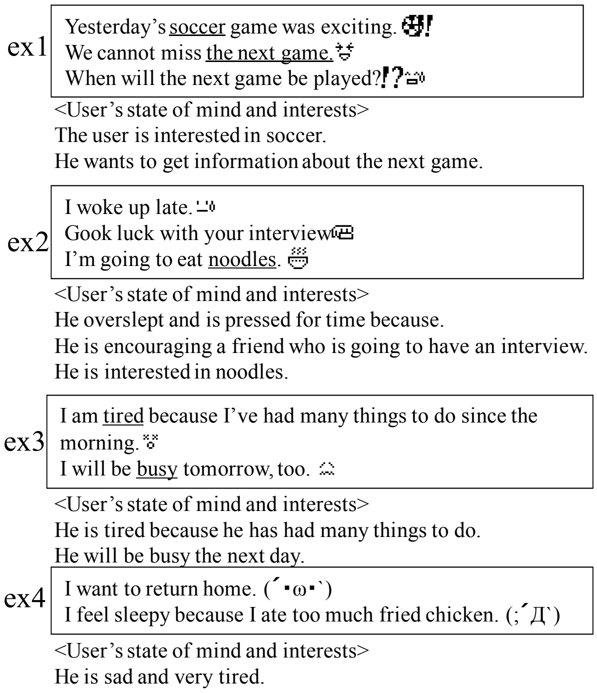


Figure 1. Examples of email that reveal the user's state of mind and interests

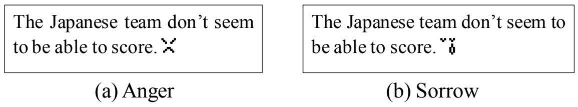


Figure 2. The same sentence can convey a different state of mind.

2.2. Content delivery system

With this system, we aim to determine the user's interests, preferences, and the extent of his or her fatigue by consecutively analyzing emails containing pictographic characters that he or she exchanges daily with friends and family members, and to deliver content that is appropriate for his or her interests and current state of mind. An example of potential use of the service to be provided by this system is shown in Figure 3. The service proceeds as follows:

1) The user exchanges emails with friends as usual.
2) An application extracts and analyzes words, pictographic characters and emoticons contained in these emails.
3) The application determines the user's interests and current state of mind and saves that data in the mobile phone.
4) When the user wants to receive content appropriate for his or her interests and state of mind,

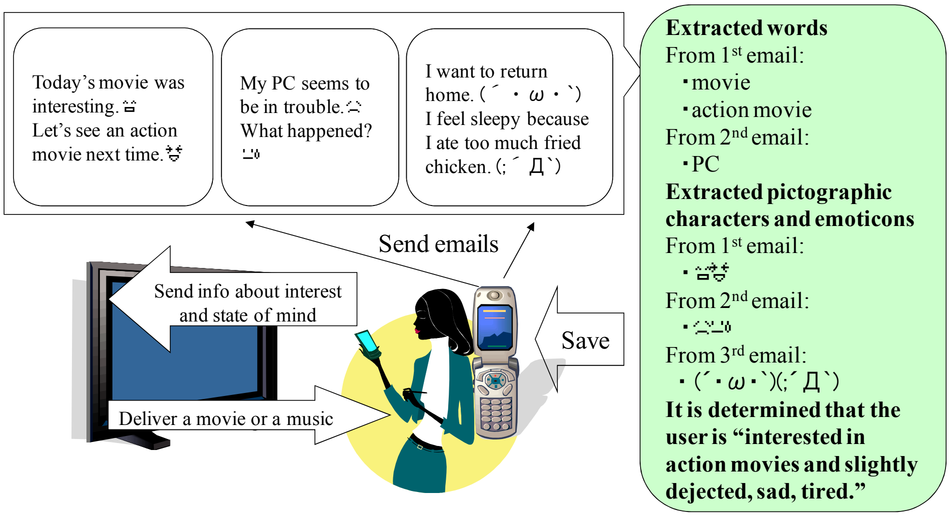


Figure 3. Example of use of the information content delivery service

the analysis result in 3) is sent to his or her TV (which relays the result to the delivery server).

5) The delivery server delivers content appropriate for the user.

2.3. Research topics

(1) Classify and characterize pictographic characters and emoticons.

In order to allow the deduction of the users' interests and state of mind from the pictographic characters and emoticons used in his or her emails, it is necessary to determine the deeper meaning that each pictographic character and emoticon conveys [12].

(2) Establish an algorithm to determine the user's interests and state of mind.

It is necessary to establish an algorithm to determine the user's interests and state of mind from the words, pictographic characters and emoticons contained in his or her email.

(3) Establish an algorithm to determine the appropriate content to be delivered.

It is necessary to establish an algorithm for selecting the content appropriate for the user, based on his or her interests, preferences and state of mind, which have been deduced from his or her email.

3. Proposed algorithms

This section describes our solutions to topics (1) (2) and (3) identified in Section 2.3.

3.1. Algorithm for determining interests and preferences

Figure 4 shows the algorithm for determining interests and preferences from words, pictographic characters and emoticons contained in an email, and for deducing the state of mind and extent of fatigue from the expressions, pictographic characters and emoticons used.

The interests and preferences are deduced from words that indicate interests, such as baseball and tennis, and pictographic characters that indicate interests, such as and , and also from past analysis results. Similarly, the state of mind and extent of fatigue are deduced from expressions used (choice of words), the percentage of pictographic characters and emoticons used in email text, and the types of pictographic characters and emoticons used. It can be assumed that a person in a poor state of mind tends to use plain or blunt expressions, that a person in a better mood tends to use pictographic character and emoticons more often, and that a person uses different types of pictographic character and emoticon depending on his or her state of mind. To confirm these assumed tendencies, we conducted a questionnaire survey with nine frequent users of email. More than half of the respondents said that they use more pictographic characters and emoticons when they are in a better mood, and almost all of them said that the pictographic characters and emoticons they choose reflect their state of mind.

We propose the algorithm for deducing the state of mind and extent of fatigue from pictographic characters and emoticons shown in Figure 4.

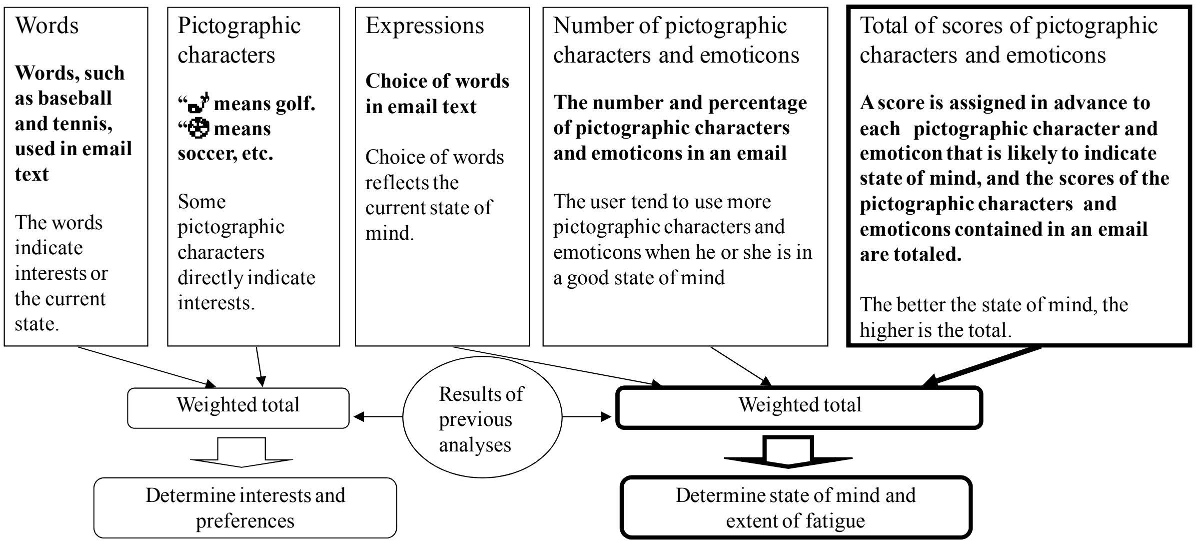


Figure 4. Algorithm for determining interests and preferences

3.2. Algorithm for deducing the state of mind from pictographic characters and emoticons

In this paper, we focus on the pictographic characters offered in NTT DoCoMo's [13] mobile phone service and emoticons, including 2-byte characters. Today, more than 200 pictographic characters and more than 1000 emoticons are used. In order to narrow down the list and select only those pictographic characters and emoticons useful for deducing the state of mind, we classified and characterized them in the following sequence of steps.

3.2.1. Selection and classification of pictographic characters and emoticons. We selected 43 pictographic characters that were found to express facial expressions and feelings. From among frequently used emoticons pre-installed in mobile phones and other emoticons, including 2-byte characters, we selected 64 emoticon that were found to indicate feelings clearly.

3.2.2. Classification of state-of-mind elements and weighting factors. We selected six state-of-mind elements: happy, angry, sad, optimistic, tired and affectionate. The newly selected element, "affectionate", is usually not relevant to the selection of an item of content, but is used only when its value is extremely high or extremely low. Since one pictographic character or emoticon can convey a variety of feelings, each is defined as a combination of vector values of several state-of-mind elements. The vector value of each element ranges from 0 to 5. For example, for one pictographic character "", a vector value 5 may be assigned to "happy", 0 to "angry", 0 to "sad", 3 to "optimistic", 0 to "tired", and 2 to "affectionate".

We conducted a questionnaire survey with 36 students (29 males and 7 females) in our university to validate our selection of the six state-of-mind elements, and to determine the vector values of each pictographic character. We asked the students to select the states of mind that are associated with each pictographic character or emoticon. The distribution of state-of-mind elements for each pictographic character is shown in Figure 5. The value of each state-of-mind vector was normalized by dividing the total votes in the survey by the number of respondents, and multiplying the result by 50. The one decimal place of this value was rounded off. A small number of respondents suggested the inclusion of "surprised" and "worried" as possible state-of-mind elements, but most claimed that the six states-of-mind elements were appropriate and sufficient for mapping the pictographic characters and emoticons. Examples of vector values for some pictographic characters are shown in Table 1.

With some pictographic characters, a few students associated states of mind that are quite different from, or even opposite to, those of other students. The differences were particularly pronounced between male and female students. However, the majority gave consistent associations between pictographic characters and emoticons and states of mind.

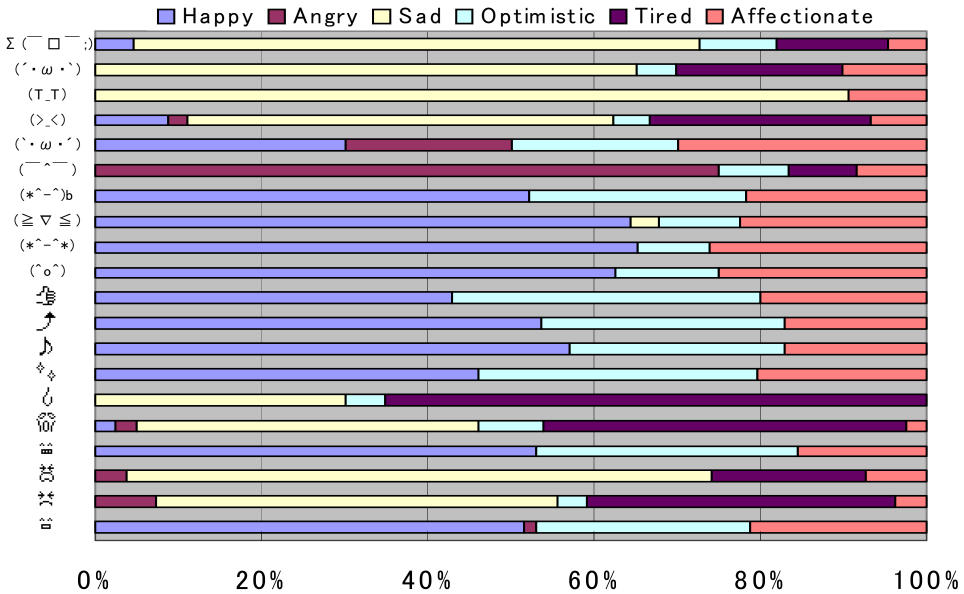


Figure 5. Result of questionnaire survey on the relationship between pictographic characters, emoticons and states of mind

Table 1. Example of state-of-mind vectors of individual value

State of mind elements / Characters	Happy	Angry	Sad	Optimistic	Tired	Affectionate
[illegible]	5	0	0	3	0	2
[illegible]	3	0	0	2	0	5
(^-^)	5	0	0	2	0	3
(>_<)	2	0	5	1	3	2

3.2.3. Use the regular expression for searching. In order to find emoticons efficiently, we chose to use pattern matching of character strings in an email. A new emoticon is often created by adding a character(s) to an existing emoticon. For example, the emoticon (^o^) (smiling face) is expanded to ＼(^o^)／ or v(^o^)v. If the regular expression is used, a pattern matching operation can find not only (^o^) but also ＼(^o^)／ and v(^o^)v. Since the meanings of these derivatives are not so different from the original emoticon, this search method is effective in finding emoticons that have been created by adding parts to an existing emoticon.

3.2.4. Algorithm for determining states of mind from pictographic characters and emoticons. We assume that the state of mind of a person can be expressed by a finite number of state-of-mind elements. Therefore, we express the state of mind using k-dimensional vectors.

Suppose there are m pictographic characters and emoticons in an email text, M. Then, the state-of-mind vector $\varepsilon(p_i)$ of an individual pictographic character or emoticon p_i can be expressed as:

$$\vec{\varepsilon}(p_i) = (e_{ji})_{1 \le j \le k} \qquad (p_i \in M, 1 \le i \le m)$$

Where e_{ji} indicates the intensity of the specific state-of-mind element. The state-of-mind vector that can be deduced from the pictographic characters used is defined as:

$$\vec{F}(M) = \frac{1}{m} \sum_{i=1}^{m} \omega_p \vec{\varepsilon}(p_i)$$

ω_p is a weighting factor. It reflects such factors as the ratio of the number of pictographic characters to the number of emoticons or the ratio of the number of males to the number of females.

The algorithm for calculating the state-of-mind vector F(M) is as follows. Note that the state-of-mind vector of each pictographic character or emoticon (such as the one in Table 1) is pre-registered in the database.

Step 1: Extract a pictographic character or emoticon p_i by analyzing the mail text.

Step 2: Find the state-of-mind vector values for each of m pictographic characters and emoticons p_i in the database.

Step 3: Obtain F(M) by calculating the average vector value of each state-of-mind element for m pictographic characters and emoticons.

3.3. Composition of the music database and database search method

In order to determine the music type (number of chords, sound strength, and melody pattern) that the user is likely to want to hear at present, we analyze the state-of-mind elements. We have developed an algorithm for a music search and delivery system.

We created 12 MIDI (Musical Instrument Digital Interface) files in different music categories with different chord patterns. We investigated the relationship between music types and the states of mind with 24 students. We asked them to write ○ when the music type they heard matched their state of

Table 2. Related data base of rhythm and state of mind

	Category	Key	Rhythm				State of mind					
			Chord	Ratio	Volume	Melody	Happy	Angry	Sad	Optimistic	Tired	Affectionate
Music 1	Blues	B♭major	B♭7	6.0	Large	passing tone	2.9	-2.3	-1.7	4.4	0.2	1.0
			E♭7	3.0								
			Cm7	1.5								
			F7	1.5								
			Edim	1.0								
Music 2	Bop	E♭major	B♭m7	1.0	Small	passing tone	1.7	-0.4	-1.7	0.8	-1.7	-0.4
			Cm7	1.0								
Music 3	Country	Cmajor	F	1.0	Small	auxiliary tone	-0.2	-0.4	1.7	1.0	1.9	0.6
			G	1.0								
			Em	1.0								
			Am7	1.0								
Music 4	Pops	Cmajor	F Bb	1.0	Large	passing tone	3.1	0.0	1.3	1.7	1.0	1.7
		Fmajor	G C	1.0								
			Em Am	1.0								
			Am7 Dm	1.0								
Music 5	Classic	D♭major	D#m	7.0	Small	passing tone	1.0	0.0	-2.0	4.0	2.0	3.0
			C#	2.0								
			B	5.0								
			D#sus4	4.0								
			G#m	1.0								
			C#sus4	1.0								
Music 6	R&B	Cmajor7	CM7	1.0	Large	auxiliary tone	2.7	-1.7	-0.8	2.3	-0.2	1.0
			C6	1.0								
			Dm7	1.0								
			Dm6	1.0								
Music 7	Rock	Cmajor7	FM7	1.0	Small	auxiliary tone	0.8	-1.3	1.3	1.0	0.6	0.8
			G	1.0								
			Em	1.0								
			G9	1.0								
Music 8	Latin	Cmajor7	CM7	2.0	Small	auxiliary tone	1.7	-1.7	-1.9	3.3	-0.4	0.2
			Dm7	1.0								
			D♭M7	1.0								
Music 9	Jazz samba	Emajor	D	1.0	Large	passing tone	2.9	-1.3	-0.4	1.5	-1.0	0.6
			E	1.0								
			F#m	0.8								
			G#m(♭5)	0.5								
			A	2.8								
			Bm	1.8								
			A#aug	1.0								
Music 10	Ballad	Cmajor	A7sus4	2.5	Small	auxiliary tone	-0.6	-0.4	2.5	0.4	2.5	1.7
			D9	1.5								
Music 11	Techno	Aminor	DM7	1.0	Large	auxiliary tone	-0.6	-0.6	0.4	0.2	-0.4	-0.6
			E6	1.0								
Music 12	Rock	Dmajor	D	5.0	Large	auxiliary tone	-3.0	1.0	2.0	-1.0	2.0	0.0
			G	4.0								
			A	2.0								
			Bm	1.0								

mind, × when they did not match, and nothing (space) when neither of them was true. If ○, 1 is added to the value. If ×, 1 is subtracted from it. If nothing, no operation was made. The state-of-mind value was calculated by dividing the total by the number of people, and multiplying it by 5. The results for each music type are shown in Table 2.

We created music metadata that consists of music title, category, key, chord, and state-of-mind elements. Music type data was extracted from each MIDI file. The state-of-mind data was obtained by searching the basic element database. The process of deriving state-of-mind element data from a MIDI file (MU_i) is as follows.

Step 1: Extract the number of chords, volume, and melody pattern from the MIDI (MU_i) file.

Step 2: Compare these with data in the basic element database (Table 2), and determine the music type that best matches the extracted data.

Step 3: Obtain the state of mind element values from the matched music type, and register them as the state of mind values of MU_i.

Step 4: Repeat steps 1 to 3 for MU_0 to MU_j. Build a music database for each of MU_0 to MU_j.

Step 5: Compare the user's actual feeling data with the feeling element data in the music database, and determine that music that best match each other.

4. Development of a prototype system and evaluation

This section describes our prototype system to evaluate the proposed methods and the experimental results also.

4.1. Prototype evaluation system

We implemented the state-of-mind deducing algorithm and the state-of-mind element database for the proposed pictographic characters and emoticons on a PC. In order to collect sample mails and evaluate the algorithm, we built a website with an input form written in JavaScript. The form enabled the user to input his or her personal information, such as name, gender, age, and the prefecture in which he or she had been brought up, as well as mail sentences and his or her subjective feeling values. We used a set of "i-pictographic characters [14]" for the input of pictographic characters, and a piece of free software called "Emoticon Helper Mini [15]" for the input of emoticons. The prototype software was written in Perl. The state-of-mind database was built using MySQL. We registered the state-of-mind elements and scores of 43 pictographic characters and 64 emoticons in the database. Figure 6 shows the configuration of the prototype system. The system operates as follows.

Step 1: The user inputs an email that contains pictographic characters and emoticons using the input form. The user also inputs his or her subjectivity values.

Step 2: Extract pictographic characters and emoticons that express user's feelings from the email text.

Step 3: Access the MySQL database to extract the score of each state-of-mind element of the pictographic characters and emoticons.

Step 4: Calculate the state-of-mind element score, and deduce the state of mind. Identify the three strongest emotions in each state of mind.

Step 5: Save the email text and the deduction result in text form.

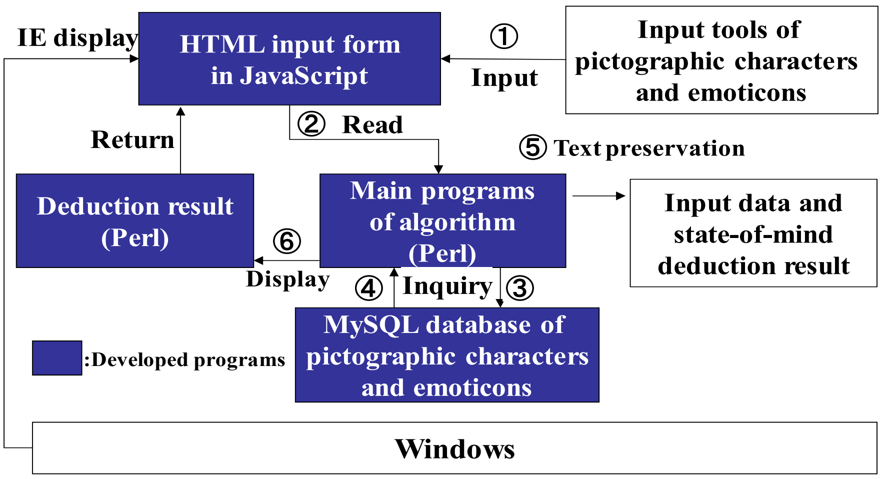


Figure 6. Block diagram of the system and programs developed

Step 6: Display the extracted pictographic characters and emoticons, the user's subjectivity feeling value, and the state-of-mind deduction result.

4.2. Evaluation

We collected one sample mail from each of 64 students (43 males and 21 females) by using a prototype system. We calculated the state-of-mind value of each person and compared it with each person's subjective feeling value. Using this result, we excluded the item of data with the highest state-of-mind value and the item of data with the lowest value, leaving 62 items of data for evaluation.

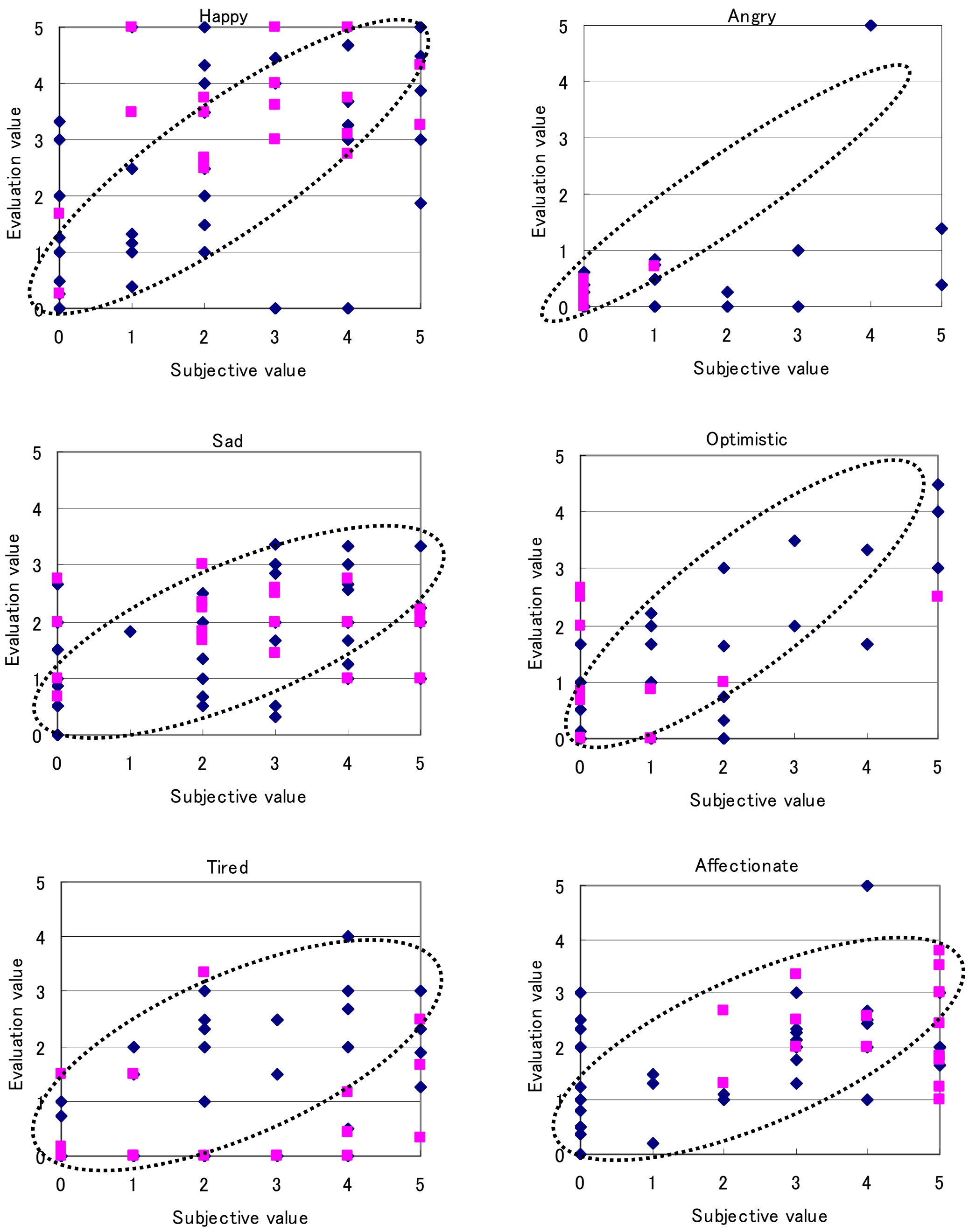


Figure 7. Correlation between the subjective value and the evaluation value (Note：◆Male ■Female)

The correlation between the calculated state-of-mind value and the subjective self-declared feeling value for each state-of-mind element is shown in Figure 7. The correlation for “angry” was the highest (the correlation coefficient = 0.66), and that for “affectionate” the lowest (the correlation coefficient = 0.34). The correlation coefficients for “happy”, “sad”, “optimistic” and “tired” were 0.53, 0.65, 0.35 and 0.49, respectively. The average was 0.5. These results were more or less what we had expected. The definitions of "optimistic" and "affectionate" were vague and differed between genders and between individuals. The definitions of these states of mind made by the students may have been different from what we had in mind, and this fact would explain the low correlation for “anger” and “affectionate”. Overall, the correlation coefficients of the female students were lower than those of male students. This is probably because there were more male students than female students among our subjects.

Figure 8 shows the statistics of the correlation between subjective feeling values and calculated values of all state-of-mind elements. The average correlation coefficient value for males and females were 0.63 and 0.57, respectively, and the overall average was 0.61, which is sufficiently high.

We evaluated whether an appropriate item of music can be selected from the state-of-mind values by referring to the music database. We selected three students (two males and one female) who recorded a h igh correlation value. Specifically, w e compared the strong state of mind obtained from the deduction result with data in the database shown in Table 2, and selected three pieces of music. We checked whether the pieces of music selected matched their feelings. We asked each student to write ◎ if the first piece matched their feeling, ○ if the second piece matched their feeling, △ if the third piece matched their feeling, and × if none of the three pieces matched their feeling. The results are shown in Table 3.

The piece number in Table 3 correspond to the piece number in Table 2. The piece encircled by ○ is the one that matched the subject’s feeling. The state-of-mind element to which the woman gave the highest value was "affectionate." However, since that value

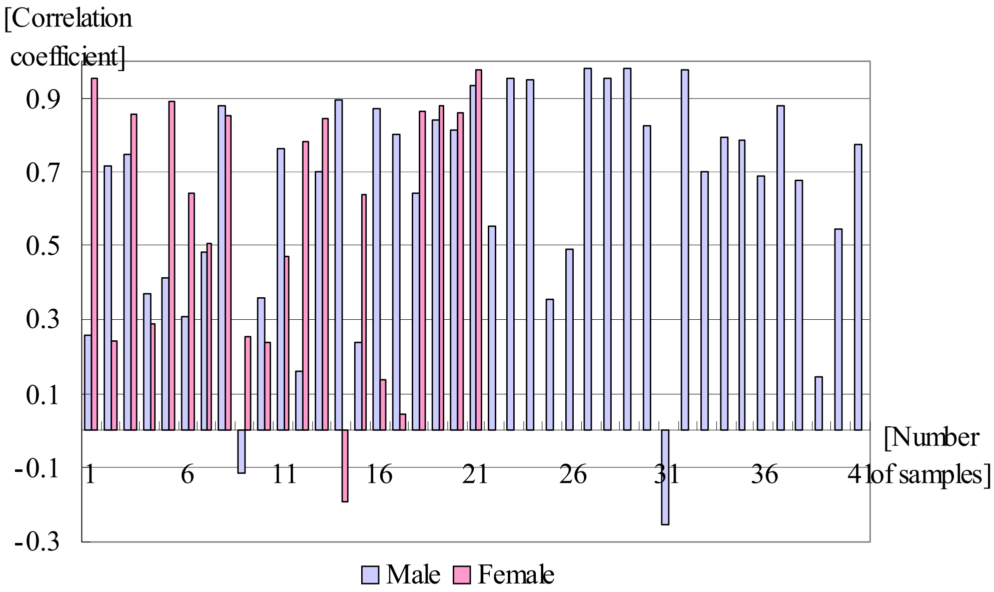


Figure 8. Statistical graph of individual correlation value

Table 3. Evaluation of music recommendation method

Items / Gender	Strong state-of-mind element presumed			Individual's correlation coefficient	Selected piece of music	Evaluation
	First element	Second element	Third element			
Male 1	Sad	Tired	Angry	0.705	3,12,10	△
Male 2	Sad	Tired	Happy	0.828	10	◎
Female	Affectionate	Happy	Optimistic	0.975	4,1,9	△

was not pronounced, we selected pieces mostly from "happy" and "optimistic". We found that at least one of the three pieces selected matched the subject's mood. Some answered that the second or the third piece best suited his or her mood. We find that the evaluation result was relatively good.

5. Conclusion and future work

We have proposed an algorithm for deducing the user's state of mind from the pictographic characters and emoticons contained in emails, and evaluated its feasibility. We have studied the state-of-mind elements associated with pictographic characters and emoticons, and their weighting factors, and have introduced specific values for the weighting factors. We have studied the algorithms for extracting pictographic characters and emoticons from an email, and for deducing the user's state of mind from the state-of-mind elements of the extracted pictographic characters and emoticons. We have developed a program that implements this algorithm, and a prototype evaluation system. Using this system, we have verified the effectiveness of the proposed algorithms.

We have also presented an algorithm for extracting the number of chords, volume and melody from a piece of music. We have identified the relationship between the user's state of mind and the music type he or she is likely to want to hear, and evaluated the relationship.

Future issues include the method of sending the obtained state-of-mind information to the delivery server, and the method of sending the appropriate content from the delivery server to the user. It is necessary to study how to automate the processes for determining and providing content appropriate for the user based on the obtained state-of-mind information. To sum up, future issues include the following:

- Algorithm for deducing user's state of mind from multiple emails that the user has sent in the last few hours in order to take the immediate trend of the user's state of mind into consideration.
- Algorithm for searching for and analyzing more types of emoticons and detailed parts of emoticons, such has eyes, mouth, cheek and hand.
- How to deduce changes in the user's state of mind as the exchange of emails progresses.
- How to automate the processes for determining and providing content appropriate for the user, based on the obtained state-of-mind information.
- Algorithm for deducing more detailed state-of-mind data instead of a simply state-of-mind element value calculated by the system, in order to achieve a stronger linkage with the service of content delivery.

References

[1] Kazumasa TAKAMI, Yoshikazu HONMA and Shinichiro GOTO: "A method of deducing a user's state of mind from an analysis of the pictographic characters used in mobile phone emails", in proceedings of UBICOMM2007, pp.83-88, 4-9 November 2007.
[2] Gmail, http://mail.google.com/
[3] Ono, C., Motomura, Y. and Asoh, H., "Study of Movie Recommendation System Considering Both Users' Personality and Situation", IPSJ SIG Technical Report 2005-DPS-125, pp.79-84, 2005
[4] Kikuchi, T., Sakai, H., Sueda, Y. and Murakami, K., "Context-aware human-activity support based on activity affinity-level measuring method", FIT 2003, pp.361-362, 2003.
[5] Mase, T. and Nakayama, Y., "Design and Implementation of Information Provision System Based on Position and User's Preference", FIT 2004, pp.157-158, 2004.
[6] H. Yokono, "Categorizing model expressions of dialogue sentences for emotion presumption," IPSJ SIG Technical Report 2005-NL-170, pp.1-6, 2005.
[7] T. Kumamoto and K. Tanaka, "Extracting Feelings from Newspaper Accounts on the Web," IPSJ SIG Technical Report 2005-NL-165, pp.15-20, 2005.
[8] Nakayama, N., Eguchi, K. and Kando, N., A Proposal for Extraction of Emotional Expression, IPSJ SIG Technical Report 2004-NL-164, pp.13-18, 2004.
[9] Telecommunications Carriers Association, Number of contracts according to entrepreneur, http://www.tca.or.jp/japan/database/daisu/yymm/0803matu.html
[10] Workshop on the Fortune-telling with Pictograph, "The Fortune-telling with Pictograph", JIMOS(2005, in Japanese).
[11] Shimizu, Y. and Akama, H., Various Meanings of Mobile Icons - Their Understandability Depending on Semantic Categories -, Proceedings of the 2005 Spring Conference of JSKE, pp.70-73, 2005.
[12] H. Cho, R. Inaba and T. Ishida, "Semantics in Pictogram Communication," IPSJ SIG Technical Report 2006-ICS-145, 2006/10/25.
[13] NTT DoCoMo, http://www.nttdocomo.co.jp/service/imode/make/content/pictograph/index.html
[14] i_pictographic characters Ver.1.21, http://www.nttdocomo.co.jp/service/imode/make/content/pictograph/tool.
[15] Emoticon helper mini Ver.2.01, http://www.vector.co.jp/soft/dl/win95/writing/se102620.html

System of Development Patterns in Service-Oriented Software

Jaroslav Král and Michal Žemlička
Charles University, Faculty of Mathematics and Physics
Department of Software Engineering
Malostranské nám. 25, 118 00 Praha 1, Czech Republic
kral@ksi.mff.cuni.cz, zemlicka@ksi.mff.cuni.cz

Abstract

Service orientation is the leading paradigm of contemporary software. Each paradigm has specific practices and a specific set of design and development paradigms. For service orientation it holds too. We show that service orientation is a quite complex trend: There are several types of service-oriented architectures (SOA). The various SOA types may have different domain of application, different patterns and antipatterns, they can use different modeling and development techniques. Proper selection of SOA type can be a crucial task significantly influencing likelihood of project success. The applicability of individual SOA variants depends on requirements and on general business circumstances like staff knowledge, planned business alliances, and the need to reuse existing software. The proper selection of a SOA variant is an important pattern often made by the way. It is important that some patterns can depend on the effects of the other ones. Patterns should therefore be orchestrated. We discuss here mainly the patterns for the variant of SOA called confederation where communication partners need not be looked for. Most important patterns for confederations are user (business) oriented service interfaces, reuse of legacy systems and third-party products, and the use of so-called architecture services. Architecture services can serve as message transformers, heads of composite services, process managers, and integration constructs for the integration in the large. All architecture services discussed in this paper can be viewed as instances of one generalized concept from Petri nets.

keywords SOA types, SOA development patterns, user-oriented service interfaces, generalized Petri place, specification patterns, easy prototyping, interdependency of patterns.

1. Introduction

Service orientation is a paradigm having many aspects. It is manifested by the fact that the notions "service orientation" and "service-oriented architecture" are overloaded. Different people may assign different meanings to these notions. Although there are multitudinous meanings, we will focus only on a few – probably the ones being most important in practice. We shall discuss the variant of SOA being a virtual peer-to-peer network of peers behaving like real-world (human) services. It means that any service can offer capabilities as well as it can require them.

There are several variants of such SOA. They differ in the degree of autonomy of peers and their "size".

Various SOA variants have different structure of the collection of patterns they use. The selection of the patterns depends on the "importance" of individual patterns. It can depend on the immediate technical effects of the patterns as well as on the business circumstances and plans like the use of third-party products, legacy systems, etc.

Some patterns can be blocked by business politics and business conditions like market alliances or the level of staff training. In this case it is good to know the possible losses caused by the rejection of a given pattern.

Some patterns are a precondition of the applicability of some other patterns. The most important case is the pattern building service interfaces so that they are "usable" or user-oriented. It enables/implies the pattern "coarse-grained interfaces" and prototyping via redirecting the destination of messages. Such a pattern is not generally known yet. The development of the collection of patterns cannot be any one-step process. This fact is often neglected.

Business needs and practical experiences led to an important change in the use of SOA-related communication protocol – SOAP (Simple Object Access Protocol, [34]). The shift is characterized by the increasing use of SOAP document-literal (SOAP-D) and decreasing use of SOAP-RPC (Remote Procedure Call) protocols [6]. In other words SOAP is now used as a XML-document carrying tool. Note the documents can be semantically rich and well understood by users. It is, they can be usable. Usability is no matter of choice now. It has substantial consequences for the applicability of some powerful development techniques discussed below. Such effects are not generally known.

We discuss SOA patterns according their importance for the project success.

The structure of the paper is the following: A variant of SOA called Confederation is specified. It is shown that it is preferable to use in confederations the services of two types: application services having coarse-grained user-oriented interfaces and architecture (integration) services facilitating integration of other services. Several variants of the architecture services are presented. It is shown that all the variants are instances of one concept called Generalized Petri Place. It is shown that architecture services can be used to provide user-oriented interfaces or to support powerful prototyping. It is also possible to use architecture services to compose other services and processes. It is discussed how service orientation influences the specification patterns.

2. Choice of SOA Type

Crucial service oriented (SO) pattern is the choice of a proper variant of SOA. It should follow just after the decision whether SOA will be applied or not. We must – using the system environment – apply the variant of SOA best fulfilling the requirements. The type of SOA implies what further patterns are applicable, e.g., whether ESB (Enterprise Service Bus [5]) is good for the given system.

The concept of service orientation is in this paper pragmatically understood in the following way (see [23] for more exact definition[1]): A software system is service-oriented if it is a (virtual) peer-to-peer (p2p) network of loosely related (autonomous) components called services. The services somewhat behave like the services of real world. For example they are permanently ready to accept a request to do something. They can communicate with each other. Technically they communicate primarily by asynchronous message exchanges; synchronous communication can be an option. We can then say that such a system has a *service-oriented architecture* (SOA).

Service-oriented systems can have different architecture details depending on main goals of the systems and contexts in which the systems are used. Typical cases are:

1. e-commerce;
2. e-government, health-care systems, etc.;
3. small and middle-sized enterprises;
4. large enterprises, especially global decentralized organizations;
5. process control systems (soft real-time systems, some hard real-time systems);
6. systems logically having some features of SOA:
   (a) distributed applications being logical monoliths not allowing to apply full SOA. Common feature: communicating autonomous software components.
   (b) batch systems – autonomous software systems communicating offline or applying bulk data communication.

[1]We, however, believe that this definition is too complicated and, may be, too restricting for applications of service orientation in some areas, e.g., in small or middle-sized enterprises, or in process control (i.e., real-time systems).

The autonomy of components is strongest in e-commerce systems and typically weakest in process control systems. In the cases 2, 3, and 4 the systems are formed by a core network of not too large number of services providing the basic capabilities of the systems (e.g., the services being wrapped information systems of individual offices of a state administration) and "peripheral" services providing e.g., portals on web.

SOA in large enterprises can consume large resources – money, people, and so on. There usually are powerful supervising authorities. The developed services forming the SOA are therefore in fact in large enterprises less autonomous than in small or medium firms. Large enterprises can moreover afford to develop the system from scratch or buy system like ESB and train people to use the new system.

Service-oriented systems integrate legacy systems, third-party products, and newly developed software artifacts. It is often required (see below) that the services have interfaces mirroring the languages of user knowledge domains. It is the main reason why SOAP-message literal is widely used.

In the case of the process control systems the system need not be open, the messages can have therefore formats based on remote procedure call (RPC) and middleware can be a proprietary one (e.g., based on system bus and primitives provided by operation systems).

2.1. Web Services

This version of service orientation endorsed by W3 Consortium focuses on web-oriented standards. These standards concern many aspects of service development and use.

The main idea is that the individual services should be strictly standardized to be able to communicate (or serve) to anyone (any other service) in the web world. The computerization should go so far that selection of cooperation partners can be done by services themselves.

It is very promising. But there is also an opposite side of the solution: Implementation of all the standards (that are very complicated and moreover changing) is very complicated and can be reasonably done by quite large teams only.

There are several further issues with SOA based on web services. Web services must use universally applicable standards and such standards are difficult to be used properly by (human) users unless the use is based on libraries provided by large software vendors. But it leads to a dangerous situation called Vendor Lock-In Antipattern [4]. The standards like SOAP [34] and SOAP-related solutions tend to support the point-to-point communication rather than more complicated communication protocols and have a limited power to support orchestration of the services.

Web services seem to be the best solution for SOA systems like business-to-customer (B2C) e-commerce where communication partners must be looked for at the start of the cooperation. We call such systems *alliances* [18] for short.

The standardization of the semantics of communication messages is easier if the communication protocol is SOAP-RPC-based. It is a quite common practice that communication partners know each other permanently so the partners need not be looked for. In this case it is better to use a communication protocol based on exchange of documents as it enables the use of user-oriented interfaces discussed below. It is due to the fact that the documents can have syntax and semantics close to the language and knowledge domain of users. Such messages can be designed to be well understood by users. We say that they are *user oriented* or user friendly. It has many desirable consequences (compare e.g., [6]). An issue is that the semantics of the messages cannot be at present fully standardized and almost proprietary solutions must be invented. The above mentioned shift to document-based communication is an indication that confederations deserve substantial focus.

2.2. Software Confederations and Software Unions

Software confederation is a peer-to-peer network of loosely coupled services knowing each other. The communication between the services can be (and practice usually is) the high-level (declarative and coarse-grained) one. It makes sense to use semi-proprietary communication protocols like SOAP-D similar to inter-human communication specific for given problem domain. If the communication is SOAP-D-based, the documents should be in XML dialects.

Many existing service-oriented systems are confederations. Examples are information systems (ERP – Enterprise Resource Planning) of decentralized enterprises, advanced forms of CRM (Customer Relationship Management, [7]) and SCM (Supply Chain Management, [22]), e-government, health-care systems, etc. The principle to use user-oriented messages and user-oriented interfaces is probably the most important design and development pattern in certain SOA types. We call it *User-Oriented Interfaces* (UOI).

User-oriented interfaces have the following advantages. They are:

1. stable (rarely modified),
2. declarative (hiding implementation details),
3. good for agile development,
4. enabling easy integration of legacy systems and agile business processes
5. enabling easy implementation of screen prototypes,
6. improving system usability.

The main disadvantage of user-oriented interfaces is that they are usually not well standardized.

UOI is therefore a crucial SO pattern (a good practice, see [11] for definition). UOI covers functions like Facade[2] [11], but it substantially changes the properties of the system as a whole. Neglecting the use of UOI is itself an SO antipattern (a practice having usually undesired consequences; see [4] for definition). It is especially dangerous as people having object-oriented skills usually (according our long-time experience) do not see the prospects and opportunities of UOI. It is the SO antipattern "Well, What's New?" from [3].

The use of UOI is especially easy and desirable in confederations having a small number of highly autonomous core services. It is the case when legacy systems or third-party products are used. A good example is e-government, SOA-based information systems of small enterprises and municipal offices. We call such systems having such properties *unions*.

Unions are now frequently used by software vendors integrating large software artifacts being open source, legacy systems, third-party products, web services, and newly developed components.

3. User-Oriented Interfaces

The requirement that systems should have user-oriented (usable) interfaces implies that the interfaces of the services forming SOA must have specific property – they must be user-oriented (usable) as well. It is typical for the implementation of agile business processes (see below).

[2] Roughly speaking Facade is in object-oriented world an object providing common uniform interface to several other objects.

SOA, if used properly, is the first broadly used software development philosophy allowing a seamless integration of existing software system into new aggregates. It allows existing systems to be reused and it is not difficult to see that it is the only way to build information system of e-government [17] as well as the systems supporting global enterprises, health-care systems, and others. The resulting aggregates are typically confederations.

Let us discuss the case of e-government in details: The system of e-government must be as a rule constructed as an integration (interconnection) of the information systems of autonomous offices. The systems must be integrated with their local interfaces and without any substantial change of their already existing functions. The only feasible way of achieving it is to connect the systems to a middleware (in this case usually web – either Internet or a private network) enabling the communication between the systems. Business processes are in e-government called admministrative processes. Technically the admministrative processes do not differ from usual business processes.

The construction of the systems in such way has from managerial point of view substantial advantages – it allows saving of immense investments into existing (legacy) software. The proposed solution of the integration of the components (e.g., information systems) can be implemented almost unnoticeably by their existing local users. It saves investments into training and the expenses and loses caused by errors of end users during their adaptation to a new system. The resulting system tends to be a confederation, usually a union.

The service-oriented architecture is then a principle allowing the integration of software artifacts providing basic capabilities. They are often wrapped applications. We shall call them application services. The integration can be supported by architecture services discussed below. This attitude is a crucial SOA pattern.

Application services provide basic user domain capabilities. Application services integrated into a service-oriented system must be usually designed so that they can be easily used in business processes. The business processes must be agile – they should allow on-line users' involvement to be able to react to emergency situations. The users should be responsible for the business consequences of the processes. It is difficult to achieve if the interfaces are not user oriented.

User-oriented interface is as a rule coarse-grained and rather declarative, i.e., specifying rather what to do something than how to do it. It brings a pleasant benefit – the reduction of the load of the communication channels. Application services must and should be integrated as black boxes (like in e-government).

User-oriented interfaces of application services must be as a rule developed in close cooperation of developers and users. The system documentation can consist of the interfaces of application services only. It indicates that during the development of such systems many features of agile design and development of software systems can and should be used. At the same time SO enables the use of agile principles in the development of quite large systems [21]. It is typical for unions.

A very important advantage of user-oriented interfaces is that they mimic the interfaces of real-world services. The interfaces of real world services are often successfully used for a long time; some of them for decades and some even for centuries. Such real-world inspired service interfaces have a good chance, if formalized properly, not to be modified frequently. User-oriented interfaces enhance the usability [26] of the application services and also the usability of the entire system. Application services (e.g., legacy systems) are usually integrated together with their already existing local user interfaces. THe concept of usability of interfaces should be applied inside the system. It is not easy for IT experts to accept it as they must be able to be a bit skilled in user problem domain. Usability is advantageous for requirements specifications as well as for the agile software development processes.

4. Service Roles

Crucial property of services in SOA is that they all have technically the same properties. It is, they are all peers of a virtual peer-to-peer network. In confederations, however, using a logical view we need services providing functions supporting the integration of the application services. The services can provide the capabilities of Enterprise Service Bus or enhance them. They can provide broker services.

It can happen that the current interfaces of an application service are not user oriented, see the antipattern Chatty Services from [3]. Chatty services require many "tiny" messages per one meaningful action in the sense of users. The antipattern Chatty Services can be refactored (avoided) by the use of specific services called *front-end gates* (FEG). FEG is a service transforming user-oriented semantically rich messages produced by users or other services into series of fine-grained (implementation-oriented) messages required by the interface of a given application service and vice versa.

Front-end gate is one of the so called *architecture services* being the units facilitating the construction of service-oriented architectures. The existence of architecture services is an important feature of service-oriented philosophy. It can be viewed as a development as well as design pattern. Application services are often legacy systems. The pattern "Integration of Legacy Systems" is an important SOA pattern. It is, however, similar to the object-oriented antipattern Legacy Systems [4]. It is an indication that the service-oriented philosophy is substantially different from the object-oriented one. FEG provides capabilities similar to Facade or Proxy patterns known from [11] but it has substantially different overall properties. FEG[3] is more an architecture pattern than a design pattern. For details see the section Generalized Petri Places below.

Application service is usually integrated as a black box whereas infrastructure services are usually newly developed and therefore integrated as white boxes.

To summarize the services in software confederations can of two basic types:

1. Application services (typically wrapped legacy systems or third-party products) providing basic capabilities (operations) of the system can be legacy systems, third-party products, or newly developed systems.

2. Architecture (or infrastructure) services supporting the integration of application services into a service-oriented system. (Note that the term "infrastructure service" has here the meaning different from the meaning used in ITIL methodology [14, 15], so we will not use it.) Besides the front-end gates we shall discuss the following architecture services: portals, data store services, process managers, screen prototypes, and generalized Petri places. All the services can be developed using very similar techniques and tools.

The acceptance of the concept of architecture services is crucial for the applicability of the patterns discussed below. The architecture services should be applied in all contexts except alliances and process control systems.

4.1. Architecture Services

Architecture services provide capabilities enabling various forms of integration of application services. The capabilities include enhancement of services interfaces and communication protocols, business process control services, or services acting as routers.

The use of architecture services is a very important development and design pattern of (confederative) service-oriented systems. It is possible that this principle can be successfully used in some alliances too.

Front-End Gates

If we want to reuse legacy systems or simply applications in SOA, the first issue to be solved is the reconstruction of their interfaces. The original interfaces are usually too fine-grained, disclosing implementation details, and often too developer oriented. A very flexible solution is based on the technique of front-end gates (service adapters). The front-end gates provide capabilities provided in object-oriented world by object adapters (facade), proxies, and so on. The capabilities of front-end gates are, however, substantially more powerful than the ones offered by the object-oriented techniques.

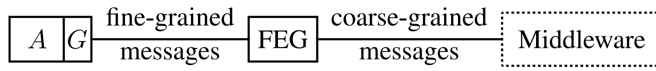


Figure 1. Connection of a front-end gate

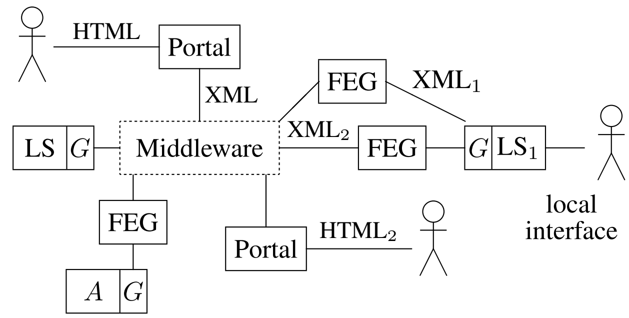


Figure 2. Multiple front-end gates

The interface of an application A can be provided by none, one, or more front-end gates (FEG). A is accessible only through its FEG(s). FEG is a generalization of the concept of connectors in Enterprise Service Bus [30].

It is crucial that FEG is a peer of the virtual peer-to-peer network too. It is in fact an adapter service – compare object adapters in object-oriented world.

The resulting service-oriented system then can have the logical structure from Figure 2. Different FEG of a service can be used for different groups of its communication partners.

The development of FEG has a lot of common with the development of portals of the system as in both cases the task is to develop an automaton transforming k-tuples of input messages into m-tuples of output messages and sends them to (distinguished) destination services. The destination service consumes/processes them. We can use XSLT [33] or tools known for compiler construction for it. So we can conclude that the interface of (application) services can always be user-oriented, if necessary. The condition is that the software component providing services have properly designed "boundaries".

Data Store Services

Practical experience with SOA indicates that service-oriented systems must integrate batch systems and therefore

[3]FEG is a software service working as an interface adapter. It is therefore similar to the concept "service adapter" but its philosophy has specific features; see details below.

services requiring an implementation of data stores on communication channels. Some parts of such systems have then features of functional decomposition. Another application of data stores is the support of sophisticated communication protocols, sometimes more complex than the publish-subscribe one.

Data store services enable us to implement a part of service-oriented system in the way known from structured design – i.e., to apply main principles of functional decomposition. Note that functional decomposition is an object-oriented antipattern [4] but here it is an important pattern enabling a seamless integration of batch applications.

Examples when data store services are needed:

- Business process control service (see Process Manager service below) can use a data store to maintain and interpret business process control data.
- Some algorithms are too complex to be executed on-line. The components implementing such algorithms must then work in batch mode and their results must be stored in a data store possibly implemented as a specialized data-oriented service. A good example is scheduling algorithms in manufacturing systems [19]. The manufacturing scheduling algorithms are performed on enterprise level in batch mode as the algorithms are too complex to be started online. The schedules must be sent to workshop level in bulk mode and then possibly modified by a workshop manager or dispatcher.
- The communication must be supervised and possibly committed/blocked by users. It is typical for business processes if we need that they can be used in and agile way. Note that the agility is necessary if it is needed that the process owner is responsible for process consequences. Agile processes in small-to-medium enterprises must be agile to respond properly on business condition changes [20].
- Data store services can be used to implement complex communication schemas, e.g., the publish-subscribe one if not provided by the middleware yet. In this case the data memory component stores a set of messages. It can be used to solve the point-to-point antipattern [3, 16].
- There can be reasons to change dynamically the destination of a message due the facts known to users only (e.g., machine tool failures, incomplete data used by scheduling algorithm, etc.). The changes can be performed by a user or by an application/service or by activities of destination services.
- Data store services can be used to implement functions necessary for debugging.

Data stores can therefore be used to enhance middleware functions, they can be used to integrate batch applications and can serve as a powerful enhancement of business processes control, especially in the cases when a given function can be provided by several application services that need not provide the same collection of elementary services or provide the same elementary services but with different quality (see [19] for details).

Process Managers

Business processes must often require on-line involvement and supervision of process owners into their operation. This feature is known as agility. Agility is a very desirable feature of business processes – especially in small and middle-sized enterprises. It is the condition for the requirement that process owner should be responsible for business losses caused by process steps as well as for on-line process changes performed by the owner. The reasons are (compare [20]):

1. The process model/definition is based on data that need not be for various reasons timely, accurate, or complete. The business conditions may also change.
2. The process owner can be obliged to commit some risky process steps.
3. The information on the process should be understandable for experts (not necessarily IT ones), e.g., at a court judging a business case.
4. The process model M should be stored as a part of business intelligence and updated by users.
5. It is desirable to be possible to have process model in different languages, if necessary, e.g., in BPEL [2], Aris, [13], workflow [36], or in a semistructured text.

As it is not desirable to have centralized services in peer-to-peer systems (compare experience with UDDI [32] and with UDDI-based world-wide repositories) we can use the following solution (see Figure 3):

1. During the process enactment a new service instance called *Process Manager* (PM) is generated on the request of the process owner O. During the generation of PM a process model M (if any) is transformed into a process control data C using parameters provided by O. O can possibly generate C directly without M. M can be copied from a data store. A solution when M contains no data or when O is a proper textual document used by a process owner, is also possible. C can be stored inside PM.

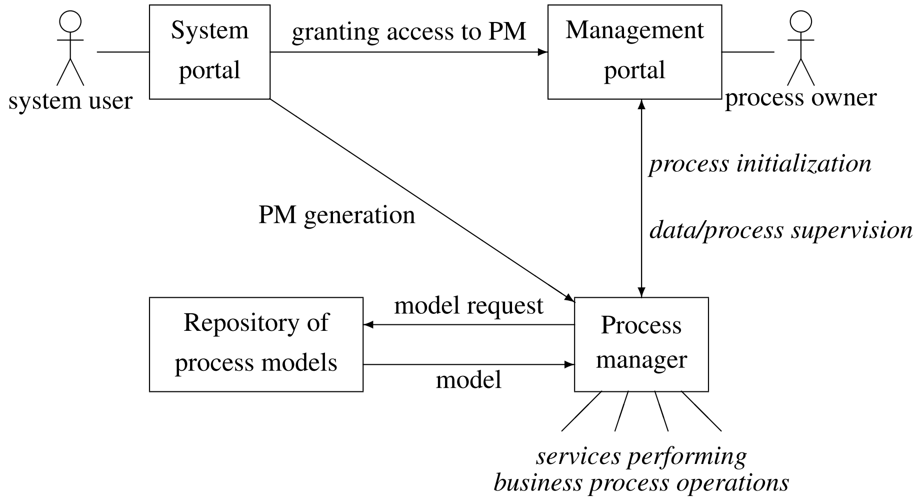


Figure 3. The use of Process Manager

2. During the operation of the process PM generates (using C) service calls. The calls can be synchronous (call and wait for answer) or asynchronous (just send a message). C can be modified on-line by process owner, if necessary.

3. It is important for the reasons discussed above that if the process owner can supervise the process run and the process run is understandable by non IT experts, then the services should have interface based on the languages of user knowledge domains – it is, the interfaces are user oriented. We have seen that user-oriented interfaces have many software engineering advantages. Note that Process Managers are from logical point of view portals of subnetworks of services providing the operations of the business process. The subnetwork behaves like a portal SOA.

Portals

It is advantageous to design a portal (system user interface) as a service (peer of the network) providing the user interface to the some functions of the system to a specific group of users. Any system can have several portals.

In the case of process managers it is meaningful to generate not only the manager but also a portal (user-interface) for it. Another implementation can be via portlets plugged into a portal. According to our experience it is a less flexible solution.

Portal SOA A service in SOA can in principle communicate with any other service. This possibility can be reduced according to service role it plays in the system according the principles of the design and implementation of a given variant of SOA.

The simplest version of SOA has services of only two types: application services and (usually) one portal. The application services can communicate with and using the portal only (Figure 4). Such solution is called *portal SOA*. Note that the application services can be structured, they can be again (virtual) networks of subservices.

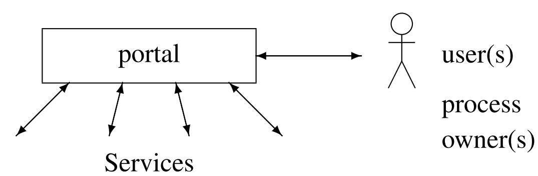


Figure 4. Portal SOA: logical view

A SOA system can have subnetworks possessing different SOA construction principles. An example is inclusion of multiple e-commerce (sub)systems in a large ERP. Another example is the subnet providing support of a business process. The head of the subnet can be a proper service – for example a process manager service PM discussed above. Often the services can be composite services.

Portal SOA is recommendable in the situation when the application services are very autonomous and the agility of business processes is desirable, the application services can be equipped by user-oriented interfaces and the system response times need not be too short. Such conditions are fulfilled quite often. The implementation of portal SOA is simpler than the development of general SOA. It is therefore important to detect whether the system to be developed can have the architecture of portal SOA.

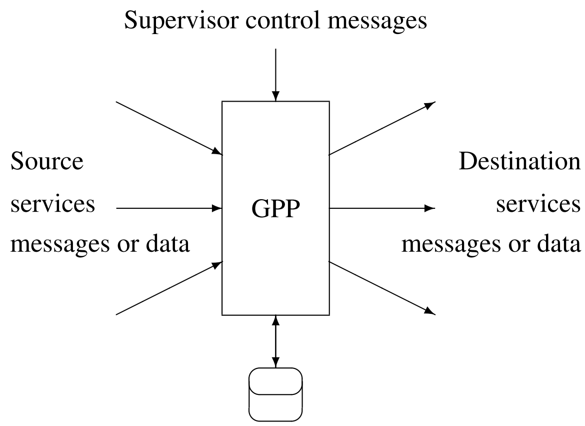


Figure 5. Generalized Petri place (simplified)

Generalized Petri Places

All the above discussed architecture services (with a partial exception of portals) can be viewed as specific variants of the service type called *Generalized Petri Place* (GPP). GPP transforms tuples of input messages into tuples of output messages. It can have its local data store (Figure 5). The functions of GPP, e.g., message routing, can be influenced by a (human) supervisor. GPP is a generalization of the concept "place" in colored Petri nets [27]. It is possible to use tools like an XSLT [33] engine to generalize the implementation of a front-end gate to have m "inputs" and n "outputs" (Figure 5). In other words: a front-end gate can be transformed into a generalized transducer transforming m-tuples of input messages from several sources into n-tuples of messages sent to several destinations.

The functions of GPP are similar to the ones of Facade [11] but GPP is a more general concept as the syntax of messages can be substantially changed by GPP and the messages themselves can be declarative and therefore not procedure-call oriented. The links from Figure 5 can be dynamically changeable at runtime. GPP can easily implement the functions of many other patterns from [11] like Build or Abstract Factory. The functions and behavior patterns are easily changeable at run-time as almost no source code changes and recompilations or no relinking is necessary to change the behavior. Redirecting of messages is needed e.g., for screen prototypes discussed below or in emergency situations. Redirecting can be set up on a request of a human supervisor.

On the other hand the power of GPP can be a dangerous tool in hands that are not skilled enough as there is a little syntax overhead. It is, however, well known that coding (programming) is no bottleneck of software development. The bottleneck is the requirements specifications [31]. The root reason of the problems is the snags in cooperation with users and it is almost not needed here.

A specific variant of it is the use of GPP as an entry point of a subnetwork of services (to assemble into a composite service). GPP then plays the role of a FEG of this composite service. It suffices to require that any message sent to any service of the subnetwork must pass the GPP being FEG of the composite service. This technique is then a service composition tool. We call such a GPP the Head of Composite Service.

GPP can be used to integrate several service-oriented systems. GPP then can serve as a hub enabling the integration. GPP can also connect subnets having properties of alliances, e.g., if the peers of the subnet are web services.

The systems using GPP can therefore have a very rich structure.

4.2. Operations on Generalized Petri Places

We have noted that all above discussed variants of architecture services can be viewed as modifications or special cases of generalized Petri places. Let us now systematize the types of modifications. Variants of communication protocol:

Standard (or public) protocols. The protocols used as a basic variant of communication in given SOA.

Bulk communication. This is a variant of communication used for the communication with batch systems.

The role of data store. We have discussed the following cases:

- massive data store filled in bulk mode,
- data store of messages,
- log memory,
- data store (repository) of the models of business processes.

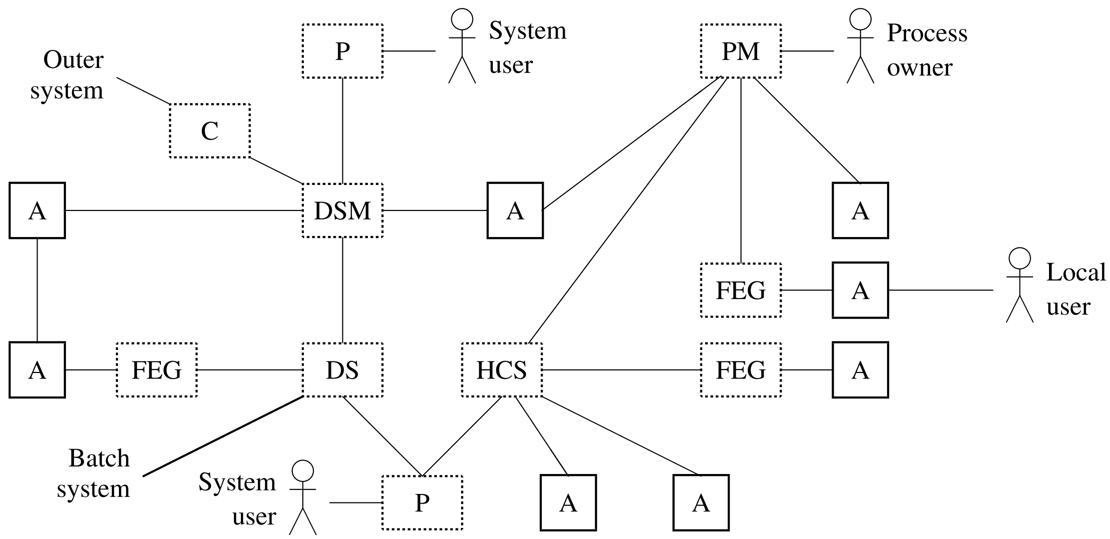
Message paths. We discussed the case when messages must go through a Head of Composite Service and the case of Connector. In fact in Portal SOA the application services communicate via Portal only.

The roles of human interfaces:

- observing and administration only,
- intelligent human interfaces.

The lifetime of services – generable and destroyable vs. permanent ones.

The transformations can be combined, so we can have a broad set of architecture services. It is a topic of further research to find out whether it can lead to further variants of architecture services.



DS – data store, DSM – data store of messages, FEG – service adapter, A – application, P – portal, PM – process manager, C - connector, HCS – head of composite service

Figure 6. Structure of SOA with architecture services

4.3. Architecture Services in Action

The architecture services discussed above enable us to design flexible service-oriented system having flexible and open logical structure that can be easily changed. Example of such a service-oriented system is in Figure 6.

The virtual p2p network in Figure 6 has two subnetworks. One comprises a service providing business operations for the architecture service Process Manager (PM). The second is a composite service headed by a Head of Composite Service (HCS).

Note, however, that the services providing operations for the business process controlled by PM can provide capabilities for other processes if appropriate.

Figure 6 shows how flexible and powerful is the service-oriented paradigm. It can be, however, dangerous if not used with caution.

A very important SOA pattern is a proper use of universal middleware components and the proper use of architecture services. ESB is not broadly used in unions. The reasons are not clear yet. Note, however, that ESB can imply Vendor Lock-In antipattern.

4.4. Fuzzy Tiers in Confederations

If we summarize the above discussion, we can distinguish the following tiers in a confederation: basic message-transport middleware, partly programmable middleware (typically Enterprise Service Bus – ESB), middleware enhancements (FEG, some Data store types, etc.), application services, composite services, orchestration, and system portal(s). Orthogonal to these tiers are local user interfaces. So the system has the structure from Figures 7 and 8. Problem is that there are cases where the services provide functions for more than one layer (tier) – for example data stores. These services can serve as a middleware enhancement as well as a process manager. It is a service orchestration tool. GPP can serve as a service orchestration tool as well as a service composition tool.

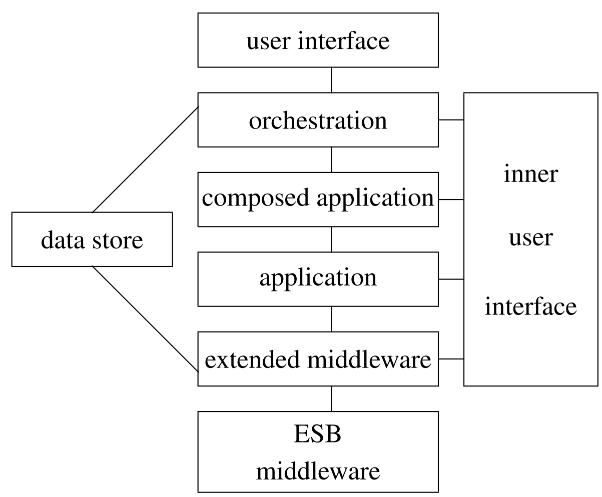


Figure 7. Tier view of a confederation

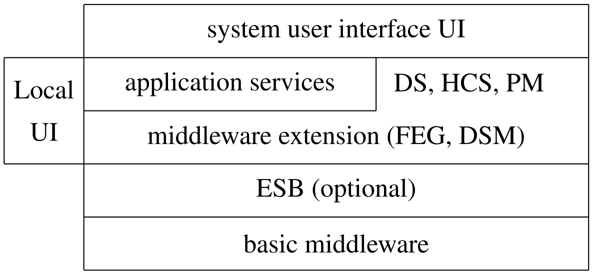


Figure 8. Tiers in SOA

4.5. Screen and Simulation Prototypes

Native service-oriented development process is the incremental one. An issue is how to simulate a service S not implemented yet. As it suffices to simulate the communication with S, we use the following procedure:

1. The middleware or a GPP can change the destination of (redirect) some messages. The messages being preferably in XML format to be sent to S are redirected to user interface UI (portal).

2. UI responses like the S would have responded.

This solution can be generalized to test response times in service-oriented process control system (real-time control systems). In this case the messages are sent to a Simulator. The Simulator can be either a hardware device or a simulator program written in a simulation language or in a language similar to C. The simulation program uses a Calendar of Coming Events (CCE) known from discrete event simulation languages. CCE can be programmed in e.g., C++ or even a discrete simulation language can be used. Details can be found e.g., in [19].

The implementation of the screen prototype is substantially simplified if the interfaces of the application services are user-oriented and uses XML documents (see SOAP-D above).

4.6. Crucial Vision and Specification Pattern

We understand SOA as a virtual peer-to-peer network of software components behaving in some sense like real-world services. The peers are called (software) services. If there is no danger of misunderstanding, the term *service* will mean software service. This broad definition covers quite different systems, the main goals/visions of which can be quite different. Different visions then imply different marketing and technical (engineering) properties, the system of patterns inclusive.

Every SOA system[4] (SOA in the above sense) consists of (compare [20]):

- application services providing basic "atomic" business capabilities, atomic means a software system providing "basic" business capabilities viewed as a black box. An example of an application system is an (encapsulated) legacy system. Application services are usually integrated as black boxes. The development from scratch can be used if necessary or appropriate.

- middleware providing tools for transport of messages between (i.e., supporting the communication of) the peers

- architecture services enhancing the capabilities of middleware. Examples are service adapters and portals. Architecture services are usually developed from scratch, i.e., they are white boxes.

The properties of all the three tiers substantially depend on the global system goals. It then implies what patterns are applicable. Let us give some examples.

1. e-commerce systems. The aim is to support worldwide business activities. It follows that a world-wide network must be used to implement the middleware. It is feasible only if no proprietary standards are used. The use of architecture services is very limited. Systems are very open. The use of web services is appropriate.

2. Process control systems. The main aim is very secure software developed almost entirely from scratch. The number of services is limited. The details of communication protocols can be agreed, the architecture services are usually not used. An exception is the use of portals (client/user tier). The service interfaces can be fine-grained and IT developer oriented (e.g., using R-PC philosophy).

3. System supporting large (partly) decentralized organizations (e.g., e-government, municipal authorities, health institutions and networks, etc.) and small to medium enterprises. The main aim is the reduction of the maintenance of such systems and integration of autonomous. The system consists of large application software services. It is preferable to use architecture services enabling a seamless integration of the services supporting organization units or the integration of third-party products. Some parts of such systems can support e-commerce. These subsystems then have the properties specified in the point 1. Such systems are quite frequent and are in fact the engine of global economy.

[4] SOA system is an abbreviation for "system having SOA".

4. Portal SOA. The communication of services must be controlled or supervised by users. In this case it can be good to generate service requests by portal The capabilities of middleware are used not too much. Service adapters (front-end gates, FEG) can be useful. Such an arrangement can be used for the implementation of agile business processes (see [28] and below). The crucial pattern is to start with the decision what variant of SOA is to be used and what application services and architecture services should be used. For the reasons discussed below the software artifacts implementing application services are as a rule large and the service interfaces are coarse-grained.

4.7. Further Notes on the Choice of a Proper SOA Type

The crucial decision is the proper selection of optimal SOA type. We often have no choice. We must use different solutions for large critical systems than for an information systems supporting a small enterprise.

On the other hand the properties of SOA systems supporting business in a small enterprise and in a very large one must be surprisingly similar although due different reasons:

SME have limited resources, so it must (re)use legacy systems as much as possible. The functions of services must be user-oriented to be used properly as there are few, if any, available IT experts able to understand user needs. User-oriented interfaces are necessary for agile business processes. They are welcome if the responsibility of business process owners for their processes is required.

Large enterprises have more resources but large changes can be too time consuming and implying too high burden on end users. Agility of processes is desirable and responsibility are needed. User-oriented interfaces support information hiding in the sense of software engineering. It is good for in- and out-sourcing.

We conclude that the proper detection of the SOA type is crucial architectural and requirement pattern. It is crucial in the sense that not applying it implies fatal antipattern. Note that in business the unions are often the only possibility.

5. Conclusion

The most important property of SOA is that it is a virtual peer-to-peer architecture. It is a quite broad definition, broader than all the SOA variants defined in standards by e.g., OASIS and W3 Consortium. It includes also quite dynamic structures not having all the properties required by the standrads. For example, it allows to integrate batch systems, web services (either complying or not to the W3C web services standards), and other systems not satisfying the requirements of the standards. Too strict definitions of SOA and software services could be the reasons of partial dissatisfication with SOA observed in the last year.

We have shown that we can adapt useful SOA solution not fulfilling the requirements of complex definition typical for many SOA-oriented standards. Such solutions can successfully support the development of information systems supporting ERP of small or middle-sized enterprises. We can even apply solutions not leading to pure peer-to-peer networks. We can use tools like MQ by IBM or solutions offered by operating systems like named pipes. It is a big promise and challenge that is often missed.

We require only that all the services (peers) of a SOA have structurally similar properties. The overall logical structure of SOA is an implicit consequence of the inner functionality of the particular services and their communication rules in the way discussed in this paper. We believe that the importance of this almost obvious property is significantly underestimated and often not taken into account at all.

It is still open what further architecture services should be invented and used.

SOA could be useful in business only if the user involvement is taken into account during development and allowed during use of the system. It implies the use of user-oriented interfaces of services and application of certain features of agile development. User involvement and ROI imply that the main SOA development and even specification patterns in business should be the integration of legacy systems.

If we look into the history, we see that the problem of year 2000 (Y2K – problems with changes in immense number of COBOL programs necessary due to century change) was the consequence of the use of legacy systems written in COBOL and used for decades with almost no maintenance. It caused the problem that there were no COBOL programmers able to make the changes. It is desirable to develop systems needing almost no maintenance. The use of legacy systems is the way to the true reusability (substantial than in the object-oriented environment – compare [10]). It is difficult to assume that the modern software should not use legacy services requiring almost no maintenance. SOA enables it.

The vendors must, however, adapt their marketing strategies to the challenges of SOA revolution. Users must develop new skills and develop new business processes able to benefit from the power of the service-oriented paradigm.

The main contributions of the paper are the following:

1. It is shown that it is good to study SOA variants used especially in small firms and called confederations and unions.

2. The criteria for the selection of the variants are specified.

3. The detection of the dependencies among patterns used for the development of confederations and unions.
4. Analysis of the importance of user-oriented service interfaces and proposal how to implement them.
5. Powerful system development prototyping.
6. Concept of services and design of the most important ones.
7. The development of the concept of generalized Petri place and treatment of the architecture service as instances of the generalized Petri places.

Acknowledgement

This research was partially supported by the Program "Information Society" under project 1ET100300517 and by the Grant Agency of Czech Republic under project 201/09/0983.

References

[1] J. Král and M. Žemlička. Crucial patterns in service-oriented architecture. In *Proceedings of ICDT 2007 Conference*, page 24, Los Alamitos, CA, USA, 2007. IEEE CS Press.

[2] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana. Specification: Business process execution language for web services version 1.1, 2003. http://www-106.ibm.com/developerworks/library/ws-bpel/ 2009-05-14.

[3] J. Ang, L. Cherbakov, and M. Ibrahim. SOA antipatterns, Nov. 2005. http://www-128.ibm.com/developerworks /webservices/library/ws-antipatterns/. 2009-05-14.

[4] W. J. Brown, R. C. Malveau, H. W. S. McCormick, III, and T. J. Mowbray. *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*. John Wiley & Sons, New York, 1998.

[5] D. A. Chappell. *Enterprise Service Bus*. O'Reilly, 2004.

[6] F. Cohen. Discover SOAP encoding's impact on web service performance. *developerWorks*, Mar. 2003. http://www-106.ibm.com/developerworks/library/ws-soapenc/ 2009-05-15.

[7] J. Dyché. *The CRM Handbook: A Business Guide to Customer Relationship Management*. Addison Wesley Professional, Boston, 2002.

[8] T. Erl. *Service-Oriented Architecture – A field Guide to Integrating XML and Web Services*. Prentice Hall, 2004.

[9] T. Erl. *SOA principles of Service Design*. Prentice Hall, 2008.

[10] L. Finch. So much OO, so little reuse. *Dr. Dobb's Journal*, May 1998.

[11] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley, Boston, MA, 1993.

[12] Gartner Inc. Gartner says the number of organizations planning to adopt SOA for the first time is falling dramatically, Nov. 2008. http://www.gartner.com/it/page.jsp?id=790717 2009-05-15.

[13] IDS Scheer. Aris process platform.

[14] International Standards Organization. ISO/IEC 20000-1:2005: Information technology – service management – part 1: Specification, 2005.

[15] International Standards Organization. ISO/IEC 20000-2:2005: Information technology – service management – part 2: Code of practice, 2005.

[16] S. Jones. SOA anti-patterns, 2006. http://www.infoq.com /articles/SOA-anti-patterns 2009-05-14.

[17] J. Král and M. Žemlička. Electronic government and software confederations. In A. M. Tjoa and R. R. Wagner, editors, *Twelfth International Workshop on Database and Experts System Application*, pages 125–130, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[18] J. Král and M. Žemlička. Software confederations and alliances. In *CAiSE'03 Forum: Information Systems for a Connected Society*, Maribor, Slovenia, 2003. University of Maribor Press.

[19] J. Král and M. Žemlička. Service orientation and the quality indicators for software services. In R. Trappl, editor, *Cybernetics and Systems*, volume 2, pages 434–439, Vienna, Austria, 2004. Austrian Society for Cybernetic Studies.

[20] J. Král and M. Žemlička. Implementation of business processes in service-oriented systems. In *Proceedings of 2005 IEEE International Conference on Services Computing*, volume II, pages 115–122, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

[21] J. Král, M. Žemlička, and M. Kopecký. Software confederations – an architecture for agile development in the large. In P. Dini, editor, *International Conference on Software Engineering Advances (ICSEA'06)*, page 39, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[22] B. Lowson, R. King, and A. Hunter. *Quick Response: Managing the Supply Chain to Meet Consumer Demand*. John Wiley & Sons, New York, 1999.

[23] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz. Reference model for service-oriented architecture 1.0, committee specification 1, 19 July 2006, 2006. http://www.oasis-open.org/committees /download.php/19361/soa-rm-cs.pdf 2009-05-15.

[24] E. A. Marks and M. Bell. *Service-Oriented Architecture – A Planning and Implementation Guide for Business and Technology*. John Wiley & Sons, Hoboken, New Jersey, USA, 2006.

[25] J. McKendrick. Gartner: SOA sinking into trough of disillusionment, Nov. 2008. http://blogs.zdnet.com/service-oriented/?p=1211 2009-05-15.

[26] J. Nielsen. *Usability Engineering*. Academic Press, New York, 1993.

[27] C. A. Petri. Kommunikationen mit automaten (Communication with automata, in German). *Schriften der IIM*, (2), 1962.

[28] A. Schatten and J. Schiefer. Agile business process management with sense and respond. In S. C. Cheung, Y. Li, K.-M. Chao, M. Younas, and J.-Y. Chung, editors, *ICEBE*, pages 319–322. IEEE Computer Society, 2007.

[29] D. Sholler. 2008 SOA user survey: Adoption trends and characteristics, Sept. 2008. http://www.gartner.com/DisplayDocument?id=765720 2009-05-15.
[30] Sonic Software. Enterprise service bus, 2004. http://www.sonicsoftware.com/products/sonic_esb/ 2009-05-15.
[31] Standish Group. Chaos: A recipe for success, 1999.
[32] UDDI Initiative. Universal definition, discovery, and integration, version 3, 2002–2003. An industrial initiative, http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm#uddiv3 2009-05-15.
[33] W3 Consortium. XSL transformations (XSLT), 2007. W3C Recommendation. http://www.w3c.org/TR/xslt20 2009-05-15.
[34] W3 Consortium. Simple object access protocol, 2000. A proposal of W3C consortium. http://www.w3.org/TR/SOAP 2009-05-15.
[35] W3 Consortium. Web services activity, 2002. http://www.w3.org/2002/ws/ 2009-05-15.
[36] Workflow Management Coalition. Workflow specification, 2004.

www.iariajournals.org

International Journal On Advances in Intelligent Systems
ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2679

International Journal On Advances in Internet Technology
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING
issn: 1942-2652

International Journal On Advances in Life Sciences
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO
issn: 1942-2660

International Journal On Advances in Networks and Services
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION
issn: 1942-2644

International Journal On Advances in Security
ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS
issn: 1942-2636

International Journal On Advances in Software
ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2628

International Journal On Advances in Systems and Measurements
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL
issn: 1942-261x

International Journal On Advances in Telecommunications
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA
issn: 1942-2601