

Contents

<i>FOREWORD</i>	1
Gergely Eberhardt, Zoltán Nagy, Ernő Jeges, Zoltán Hornák Copy protection through software watermarking and obfuscation	2
Zoltán Gál, Andrea Karsai, Péter Orosz Effect of WiFi systems on multimedia applications	8
Tamás Bóhm, Géza Németh Algorithm for formant tracking, modification and synthesis	15
István Pintér Estimation of instantaneous parameters of speech signals with Teager-operator and Hilbert-Huang-transform	21
János Zoltán Szabó, Tibor Csöndes TITAN, TTCN-3 test execution environment	27
Gergely Ács, Levente Buttyán A taxonomy of routing protocols for wireless sensor networks	32
Borbála Katalin Benkő, Tamás Katona, Róbert Schulcz CASCADAS – Autonomic communication and situated pervasive services	41
László T. Kóczy, János Botzheim, Richárd Sallai, Kornél Csányi, Tamás Kuti Applying fuzzy inference in the supervision system of mobile telecommunication networks	47
Róbert Fajta, Péter Domokos, István Majzik Adding high availability features to server applications using aspect oriented programming	56
<i>Call for Papers – 16th IST Mobile and Wireless Communications Summit (x)</i>	63

Protectors

GYULA SALLAI – president, Scientific Association for Infocommunications

ÁKOS DETREKŐI – president, National Council of Hungary for Information and Communications Technology

Editor-in-Chief: CSABA ATTILA SZABÓ

Editorial Board

Chairman: LÁSZLÓ ZOMBORY

BARTOLITS ISTVÁN
BÁRSONY ISTVÁN
BUTTYÁN LEVENTE
GYŐRI ERZSÉBET

IMRE SÁNDOR
KÁNTOR CSABA
LOIS LÁSZLÓ
NÉMETH GÉZA
PAKSY GÉZA

PRAZSÁK GERGŐ
TÉTÉNYI ISTVÁN
VESZELY GYULA
VONDERVISZT LAJOS

Editorial Office (Subscription and Advertisements):
Scientific Association for Infocommunications
H-1055 Budapest V., Kossuth Lajos tér 6-8.
Phone: +36 353 1027, Fax: +36 353 0451
e-mail: info@hte.hu • www.hte.hu

Articles can be sent also to the following address:
Budapest University of Technology and Economics
Department of Telecommunications
Tel.: +36 463 3261, Fax: +36 463 3263
e-mail: szabo@hit.bme.hu

Publisher: PÉTER NAGY • Manager: ANDRÁS DANKÓ

Foreword

szabo@hit.bme.hu

Our journal is continuing with the practice of publishing English issues regularly, at present twice a year, in July and in January. As before, most part of the present issue contains English versions of reviewed research papers, selected from the preceding five Hungarian issues. We included also one research paper from open call. The editors would like to encourage prospective authors to submit their results specifically for the English issues.

Being a selection, the papers' topics span a wide range of issues of current interest: content protection techniques, speech technology, testing methodology, pervasive communications, aspect oriented programming and Fuzzy-based network supervision methods.

Eberhart et al deal with copyright protection through software watermarking and obfuscation. They propose a scheme which combines obfuscating and software-watermarking techniques in order to provide a solution to overcome the problems concerning software copy protection.

Gál et al analyze the properties of multimedia applications (video, streaming, IP phone) operated over IEEE 802.11b/g/a WiFi systems thus helping to choose among 802.11b or 802.11g and/or 802.11a systems.

Precise formant tracking has been a challenge for researchers in speech processing for long. It is an efficient tool for analyzing and altering the spectral content of speech, furthermore it provides an opportunity to modify timbre and voice quality. *Böhm and Németh* present a method to track and modify formants in speech signals. The method proposed by the authors is based on the linear prediction model.

In order to analyse the fine structure of speech signals, methods for determining nonlinear and non-stationary characteristics of speech are necessary. *Pintér* in his paper presents the Teager-operator and the Hilbert-Huang Transform (HHT) as speech processing methods suitable for the estimation of instantaneous amplitude and instantaneous frequency.

Szabó and Csöndes deal with TTCN-3 (Testing and Test Control Notation), a language which was standardized by ETSI. The paper presents the TTCN-3 test execution environment of Ericsson called TITAN. As a result of the authors' development, TTCN-3 and TITAN became a widely used test solution within Ericsson and it became also possible to contribute to the TTCN-3 standardization work within ETSI.

The design of energy efficient routing protocols for wireless sensor networks is a challenging task, which has been in the focus of the sensor network research community in the recent past. This effort resulted in a huge number of sensor network routing protocols. *Ács and Buttyán* in their paper, propose a taxonomy of sensor network routing protocols, and classify the mainstream protocols proposed in the literature using this taxonomy.

Benkő et al summarize the main objectives of and innovations in CASCADAS, one of the four 6th Framework IST-FET projects, that aim at providing both theoretical and practical background for a new generation of complex, distributed, pervasive services.

In mobile telecommunication networks the transmission level is affected by objects located between transmitter and receiver stations in the so-called Fresnel-zone. *Kóczy and Botzheim* propose the use of intelligent decision making subsystems which can decide from the degree of attenuation and from its time dependent behaviour what the reason of the attenuation could be. The paper demonstrates the intelligent module of a network supervision system based on fuzzy logics.

The transformation of existing software to a fault tolerant one typically requires redesign and heavy modifications in the original source code. *Fajta et al* analyze how to use aspect-oriented programming (AOP), an emerging programming paradigm for this purpose.

László Zombory,
President of
the Editorial Board

Csaba A. Szabó,
Editor-in-Chief

Copy protection through software watermarking and obfuscation

GERGELY EBERHARDT, ZOLTÁN NAGY

SEARCH-LAB Ltd., {gergely.eberhardt, zoltan.nagy}@search-lab.hu

ERNŐ JEGES, ZOLTÁN HORNÁK

BME, Department of Measurement and Information Systems, SEARCH Laboratory
{jeges, hornak}@mit.bme.hu

Keywords: software copy protection, software watermarking, obfuscation, reverse engineering, trusted OS, mobile software

Enforcement of copyright laws in the field of software products is primarily managed in legal way by the software developer companies, as the available technological solutions are not strong enough to prevent illegal distribution and use of software. Almost all copy protection techniques were cracked within some weeks after their market launch. The lack of technical copyright enforcement solutions is responsible for the failure of some recently appeared business models, partially also for the recent dotcom crash. The situation is particularly dangerous in case of the growing market of mobile software products. In this paper we aim at proposing a scheme which combines obfuscating and software-watermarking techniques in order to provide a solution which is purely technical, but strong enough to overcome the problems concerning software copy protection. The solution we propose is focusing primarily on mobile software products, where we can rely on the hardware based integrity protection of the operating system.

(In: 2006/5, pp.51–57.)

1. Introduction

According to the statistics of the Business Software Alliance (BSA), the global financial losses due to software piracy were about 30 billion USD in 2004 [3]. Nearly half of the used software is illegal in the European Union. Technical and legal actions could not change substantially this situation. It is a common belief that there is not much to be done against the piracy of software in the PC world. However in the world of mobile phones, where the integrity of the operating system can be trusted, the emerging market of mobile software products has still the opportunity to evolve in such way that the losses due to illegal software distribution could be avoided, or at least moderated.

In our belief the availability of a strong copy protection scheme is the most important prerequisite for further expansion of the mobile phone software market. This is why our research targeted embedded systems used in mobile phones. This way slightly different assumptions can be made than on usually discussed PCs.

A *trusted OS* is an essential ground for achieving a strong copy protection, since it is obvious that any software-based protection can be circumvented if the OS can be tampered. Solutions, when developers do not rely on the OS at all, are also possible. However, with these schemes are based on *security by obscurity*: they simply hide the parts of the code that checks the integrity and validity of the software, assuming that the time needed to reveal and remove this checking is long enough not to remarkably affect the revenues. Although the time needed for reverse-engineering and circumventing the protection of the software can be lengthened with this approach, experience proved that sooner or later all protections based on security by obscurity are cracked.

As opposed to this we propose a scheme, which merges public key infrastructure (PKI) with obfuscation and software watermarking techniques, assuming a trusted and tamperproof OS resulting in a protection supporting freely distributable, but also copy protected software.

2. Theoretical background

Two main categories of software copy protection mechanisms exist: the autonomous systems and those, which use external collaboration [10].

The protections of autonomous systems are integrated into the software itself, so the security depends only on the used software techniques. These techniques include integrity protection, software obfuscation, checksumming, encryption, preventing the running in debug mode and other methods making the task of the cracker harder [8].

The most common solutions are based on the program checking itself. As these checks are part of the program, one can reveal them by reverse engineering and can bypass the protection by modifying the code [10]. These techniques are neither theoretically nor – based on our experience – practically secure enough [2].

The other category of protection mechanisms use external collaboration; more specifically the program uses a tamperproof processor, an operating system or other secure hardware or software solutions. This support can be either on-line or off-line [10]. In case of on-line collaboration some of the checking functions are executed on other computers than the attacker could access. As opposed to this, the off-line collaboration does not require an on-line connection, but only a se-

cure hardware or software item. The secure hardware is usually a smart card, while the secure software is the part of a trusted OS, like in our case targeting mobile phone software.

To link the authorized user to his or her instance of the software, usually the services of a public key infrastructure [9] are used. For a copy protected software a license containing the information about the user, about the product issuer or distributor, and about the product itself (e.g. a hash of the particular software instance) should be attached to the product, so that the OS could check the authorization. The integrity of this license is protected by a digital signature, and the OS should not run copy protected software without the appropriate license.

However, to support multiple use cases and business models, and to allow comfortable software development, the OS should be capable of running both copy protected and unprotected software, which is one of the biggest challenges in developing a successful copy protection scheme. This means that in a manipulated piece of software or when the license is removed the OS should also detect that the code was initially protected.

As opposed to usual protection models used in multimedia files using watermarking to trace a content in order to identify its origin, software use watermarking to make possible the indication on the code that it is copy-protected. As the instructions of the code can be obfuscated arbitrarily as long as the user-perceptible output remains the same, this enables us to implement more efficacious watermarks to software than to audio/video files.

We can differentiate between two types of *software watermarking* techniques: static and dynamic. In case of static watermarks the information is injected into the application's executable file itself. The watermark is typically inside the initialized *data, code or text* sections of the executable [1,7,8,11,13,14]. As opposed to this, the dynamic watermarks are stored in the program's execution state, and not in the program code. This means that the presence of a watermark is indicated by some run-time behavior of the executable [5,6,12].

To prevent the easy removal of watermarks we can use *software obfuscation*, which is a collection of several different code transformations with the common goal to make the reverse engineering more difficult both in case of automatic tools and for the human understanding of the code [4,15]. The most important aim of these transformations is to make code transformations meant to remove the watermark hard to accomplish.

3. Requirements and quality goals

In this section we define the requirements and assumptions towards our copy protection scheme.

3.1. Trusted operating system – integrity & confidentiality

For a trusted OS we assume that the *integrity* of the undergoing processes is ensured (e.g. protected memory areas cannot be changed). However the *confidentiality* of the information flowing through the OS does not have to be guaranteed. This implies that although the method of watermark detection is kept in secret, it is hard to remove the watermark from the protected application even if the watermark generation and detection algorithms are well known to the public, thus avoiding security by obscurity.

The OS should also be capable of checking the digital signature of the applications and support PKI with certificate chains and revocation.

3.2. Same observable behavior

In our case the *observer* is the customer, longing to use the protected program. From his or her point of view the transformed program should be functionally the same as the original one. For example it should have the same windows, the same files and the same connections to the outside world, and all these should behave in the same way. However the speed, the memory usage and the inner states of the program during the execution, including the program code could be modified slightly or even to a greater extent.

3.3. Harder to reverse engineer

On one hand the obfuscation should make the *automatic* decompilation of the code very difficult, while on the other hand the code should be made incomprehensible for *human understanding* to the greatest extent possible [8]. Our goal is to make the reverse engineering difficult. This means that the reverse engineering and thus removing the protection should be extremely time-consuming in order not to be worth the effort and time spent on doing it. To state it in other way, *difficult* and *to make harder* here means that the decompilation of the protected program should include the solving of such a complex problem, whose complexity can be deduced to a problem accepted to be hard in cryptography, so typically being equivalent with as if one should have to break a cryptographic algorithm.

3.4. Harder to remove the watermark

This quality indicator is strongly connected to the one described in the previous paragraph. A robust obfuscation method not only makes the reverse engineering harder, but also prevents the easy removal of the watermark.

As in case of the trusted OS we assume only the integrity, but not the confidentiality: we have to hypothesize that the watermark detection method cannot be kept in secret. Thus, the removal of the watermark has to be hard enough also when the detection method is public, or when everybody can detect the watermark. The watermark has to be embedded in the original code deeply enough so that the watermark should not be removed without the full understanding of the *whole*

code. The watermark generation on asymmetric encryption can be used for example where the private key is kept in secret, but the algorithm itself is considered to be public.

3.5. Scalability of the transformations costs

Because every transformation has a cost in terms of speed and memory usage, it is important that the protection is effective from this point of view as well. To fulfill the different requirements and limitations on transformation costs for the different areas of the code, the accomplished transformations should be *scalable* in terms of speed and memory usage of the resulting transformed applications.

4. The proposed copy protection scheme

To summarize, our scheme is constructed from the above mentioned building blocks, taking into account the listed requirements. The integrity of the software is ensured through a digitally signed license, and if the license or its digital signature is invalid, the OS prevents the application from running. However, if there is no license attached to the application, the OS can start it, but should continuously check for the presence of watermarks in it, which designates that it is a protected piece of software, so should have a license file attached. The removal of watermark is hard because of different obfuscation methods utilized, so the attacker cannot change or break the protection to make the application run without the license.

The license checking algorithm being the most important part of our scheme is shown on the *Figure 1*.

- (1) Check if a digitally signed license is attached to the application.
- (2) If there is a license, and if both the license (e.g. its hash signature) and the digital signature are valid,

then the application can be run without accomplishing any further checking for watermarks.

- (3) If the license or its digital signature are not valid, the application should stop immediately, and the OS should make the appropriate steps (for example by logging or even reporting the event), as in this case the copy protection is violated.
- (4) If there is no license attached to the application, it could either be freely distributable software but also a copy protected but manipulated program, so it should be allowed to start.
- (5) Parallel to this the OS should start the continuous search for watermarks.
- (6) If a watermark is found in the application, its running should be stopped immediately, and again the appropriate steps should be made, because the presence of the watermark unambiguously signals that a license should be attached to the application, without which it is an illegal copy.

So, in our scheme we use software watermarks to indicate that the program is copy protected, thus the inserted watermark should have the following properties:

- it has to be able to store only a one bit information, indicating that the program is copy protected,
- it should not use a secret method to hide the watermark, as the attacker can reveal and understand the watermark detection algorithm,
- all watermark removal should not be possible via automated or manual means even in cases when the attacker might know the detection algorithm.

To fulfill these requirements we have chosen to use dynamic watermarks, because in this way the binary form of the watermark is formed during the program execution.

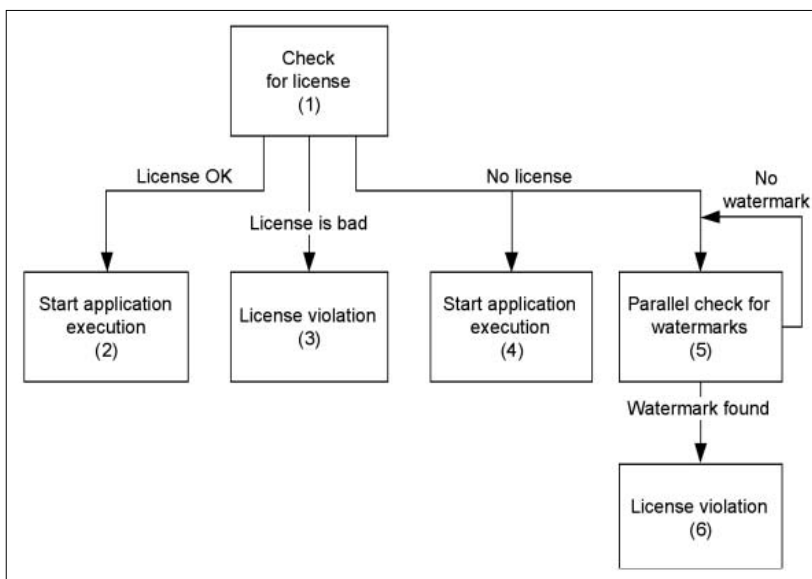
To detect dynamic watermarks the program should run for a while. This implies that the program state should be checked continuously during the execution. There is no time limit, after which it is assumed that

there is no watermark in the program. The watermark detection procedure could slow down the execution of the application, so in the proposed copy protection checking algorithm the watermark detection is done only in case the license file is not present. So it will be the developers' essential interest to provide licenses to their released products in order to exploit the capabilities of the device to the best extent possible.

The connection between the program and the OS during the watermark checking process is illustrated in the *Figure 2*.

Our dynamic watermark only stores one-bit information in form of a special number derived from a random value appended with its value transformed with a function *f*.

Figure 1. The license checking algorithm



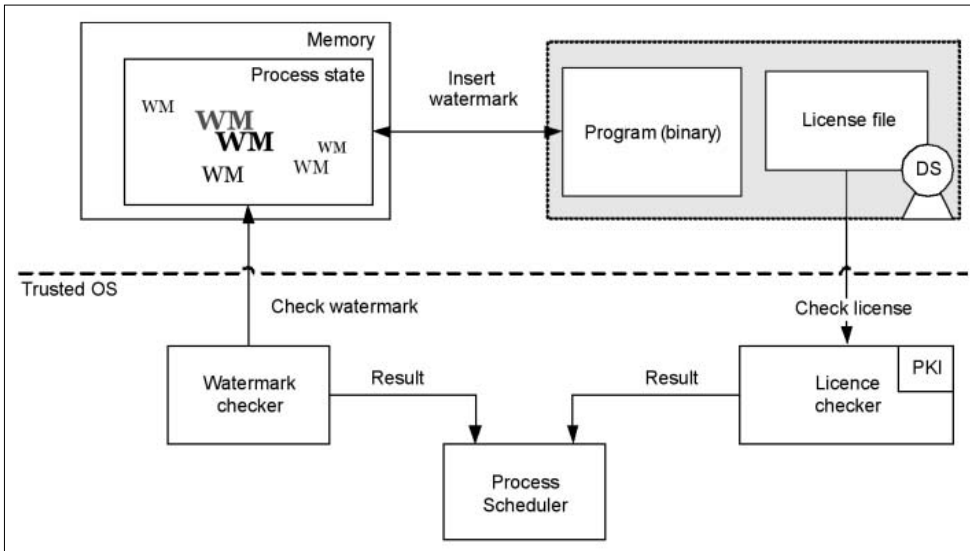


Figure 2. Connection between the program and the OS during the watermark checking process

The transformation accomplished with this function can be a digital signature, a hash value, CRC or others for example even a simple XOR operation with a constant value; its role is simply to keep the probability of appearing of such a value pair in a non-protected application low, so if such a value appears in an application frequently enough, it can designate the presence of the watermark.

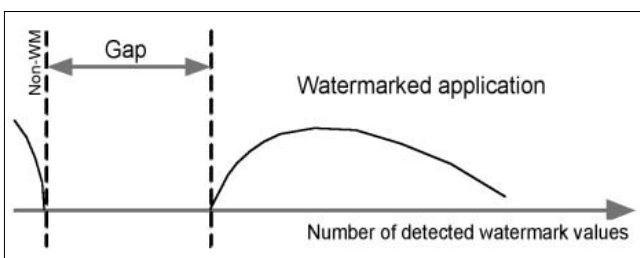
So the watermark can be defined as follows:

$$WM = (RND; f(RND))$$

These different WM values should be hidden in the application as frequently and for as long as possible, which means that these values should appear in the state of the program frequently enough to allow their detection and statistical evaluation of these detections. To achieve this goal we can for example pick the parameters of different data obfuscation transformations in a way that one or more original values of a variable (typically a loop control variable) are transformed into watermarks values, which are then stored in program state in the transformed domain.

The gap between the prevalence of such watermark values in non-watermarked and watermarked applications, as shown in the Figure 3, can allow easy detection of the watermarks by statistical means.

Figure 3. The gap between non-watermarked and watermarked applications regarding the prevalence of watermark values



As the attacker cannot know the exact value of the watermark due to its randomness, he or she can only detect, when it is formed in the program's state resulting from the appropriate inputs. Therefore the attacker has to execute all branches of the program with all possible input values to be sure that the watermark is fully removed.

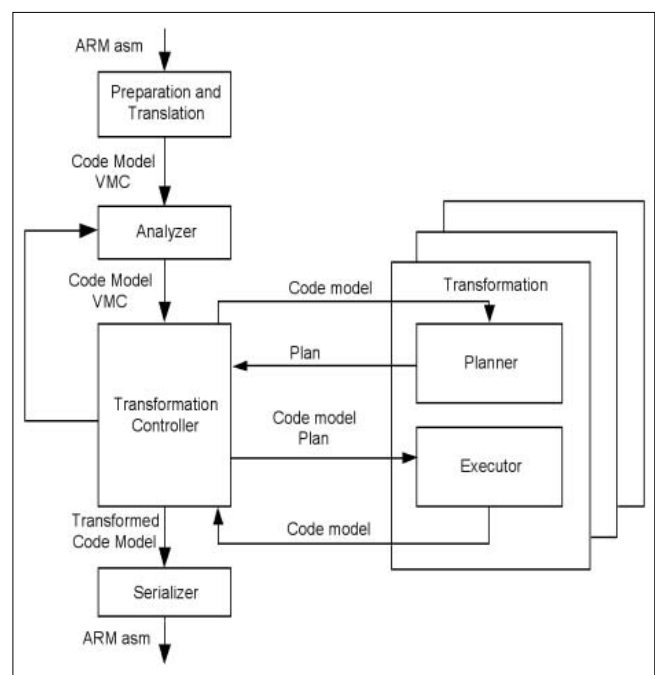
5. The architecture of the system

Figure 4 below illustrates the main parts of the system and their connections. The obfuscation and the software watermarking transformations take place integrated into the usual C/C++ compilation process, which usually starts with a preprocessing step.

The inputs of the system are the C/C++ source files of the application to be protected. Different directives can be placed in the source code to control the obfuscation and watermark insertion process. These directives are collected during this preparation process, and the original source is passed to the *Compiler*, which will produce LST file and the debug information file.

The latter two form the basic sources of information about the code, but other files originating from the *Disassembler* (disassembly LST), *Linker* (map file) and the *Profiler* (profile information) can be also used for this purpose.

Figure 4. The modules of the system and their connections



The collected information is merged and prepared to form a compiler independent representation called the *Code Model*, which is used to accomplish the needed transformations, while the code itself is translated into an abstract representation called *Virtual Machine Code (VMC)*, on which the transformations will be carried out.

After this preparation the system analyzes the gathered information, during which the *Analyzer* module performs control and data flow analysis and finalizes the *Code Model*, which then contains all necessary information to plan and accomplish any watermarking or obfuscating transformations.

The transformations are accomplished in several iterations. Upon every iteration the *Transformation Controller* creates a detailed plan about the transformation to be accomplished in the current step, along with their sequence. During the transformations the intermediate representation must remain in consistent state, which means that even after every iteration the code should be functionally equivalent with the original one.

To ensure efficiency and functional equivalency of the code, every transformation is done in two steps. The first step is responsible for making a transformation plan (*Planner*) and the other is responsible for the execution of this plan (*Executor*). The Planner is responsible for finding proper and optimal parameters for a specific transformation in the current context, while the Executor ensures the functional equivalency. This way it is enough to prove the correctness of the Executor formally.

The transformation steps batched in the iterations are applied to the abstract representation of the code until the expected goals are reached, namely the hiding of the adequate number of watermarks and reaching the desired level of obfuscation. After the transformations are accomplished, the abstract representation of the code is translated (serialized) back to an assembly source code, which is ready to be compiled by an assembler. The result of the process is the compiled object code, which is on one hand obfuscated and on the other hand it contains the watermark.

6. Results

To evaluate our copy protection scheme we have implemented the full framework system based on the above introduced architecture along with a number of transformation methods. At the end of this article introducing the proposed scheme and the framework system architecture we illustrate the capabilities of such architecture with a simple example, the implementation of the *Hide Library Calls* obfuscation technique.

Most programs heavily use calls to the standard libraries and to the operating system, and since the se-

mantics of the library functions are well known, such calls can provide useful clues to perform reverse engineering on an application. The *Hide Library Calls* technique can be used to dismiss this help from the code.

There are different methods to obfuscate the calls to such fixed functions. The basic idea behind all of these transformations is that the original API call is changed to an inner *wrapper* function of the application, which will call the real API function. The obfuscator can create an interface function to each API call, but the call of each API function can be integrated into a single function as well. In this latter case usually the value of an input parameter designates which API call should be called by the wrapper function.

In case of this obfuscation the *Planner* (see architecture in Figure 4 above) is simple, because it has to find the possible call references and has to choose a subset of them, which are to be hidden (even all can be chosen to be hidden). So the generated *Plan* contains the list of call places. The steps of the algorithm for hiding library calls once the plan is ready is as follows:

1. Creating a new function, which will be the wrapper function calling the actual API functions.

2. Assigning identifiers (values) to API calls to be hidden, and creating the instructions and the blocks of the wrapper function regarding to the calls and the identifiers.

3. Changing the original instructions calling API functions to point to the new wrapper function and passing the appropriate identifiers as additional arguments to it.

The following example shows an API call after the *Hide Library Calls* transformation is accomplished:

```
ldr lr, LI11          @ Save return address
mov ip, #1           @ Set ip to the called function ID
ldr r5, LI12         @ Load address of global variable
str ip, [r5, #0]     @ Save ID to the global variable
b HideCalls_2       @ Call function
LI11:
.align 0
.word .L12          @ Address of the next function
LI12:
.align 0
.word .LD110        @ Address of the global variable
.L12:
```

7. Summary

In the above article we have presented our scheme, which combines cryptography, software watermarking and obfuscation in order to achieve a strong technical solution for software copy protection, targeting primarily the mobile software developers. Based on this scheme we have designed the architecture of a protection tool that can be integrated in a development environment to provide copy protection services.

The architecture is robust and open in a sense that the module dealing with transformations – both water-

marking and obfuscation – is completely independent of the processor, the OS and the development environment, as it works on an abstract representation of the source code. This way, by replacing the preprocessing, translating and serializing modules, we can integrate our system into several environments.

As in the case of code transformations the formal proof of correctness of transformations is essential, all transformation are done in two steps: after planning the particular transformations in order to form a transformation sequence that fulfills our goals, the separate and much simpler transformation steps are executed so that their accomplished activity can be formally proven to be correct. This proof should be done for all transformations that can be executed within our framework.

Having the framework ready, the next step in our research is to broaden the set of such transformations to test different kinds of obfuscation, and to inject certain code fragments into our *Code Model* to implement dynamic watermarking. Our goal is on one hand to develop and test the efficiency of several control and data obfuscation methods, and on the other hand to hide dynamic watermark in the code and accomplish several measurements regarding the ability of an independent code (which will be the OS) to detect them.

Acknowledgements

The project is being realized by the financial support of the Economic Competitiveness Operative Programme (Gazdasági Versenyképesség Operatív Programja, GVOP 3.1.1/AKF) of the Hungarian Government.

References

- [1] G. Arboit,
“A Method for Watermarking Java Programs via Opaque Predicates”,
In: The 5th International Conference on Electronic Commerce Research (ICECR-5), 2002.
- [2] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, K. Yang,
“On the (im)possibility of obfuscating programs”,
In: Proc. CRYPTO’01.
Lecture notes in computer science, Vol. 2139.
Springer, Berlin-Heidelberg-New York, 2001.
pp.1–18.
- [3] First Annual BSA and IDC Global Software Privacy Study, Business Software Alliance and IDC Global Software, 2004.
- [4] C. Collberg, C. Thomborson, D. Low,
“A Taxonomy of Obfuscating Transformations”,
Technical Report 148, Dept. of Computer Science,
The University of Auckland, 1997.
- [5] C. Collberg, C. Thomborson,
“On the Limits of Software Watermarking”,
Technical Report 164, Dept. of Computer Science,
The University of Auckland, 1998.
- [6] C. Collberg, C. Thomborson, G. M. Townsend,
“Dynamic Graph-Based Software Watermarking”,
Technical Report TR04-08, 2004.
- [7] R. Davidson, N. Myhrvold,
“Method and system for generating and auditing a signature for a computer program”,
US Patent 5,559,884,
Microsoft Corporation, 1996.
- [8] G. Hachez,
“A Comparative Study of Software Protection Tools Suited for E-Commerce with Contributions to Software Watermarking and Smart Cards”,
Ph.D. thesis, Universite Catholique de Louvain, 2003.
- [9] International Telegraph and Telephone Consultative Committee (CCITT):
The Directory – Authentication Framework,
Recommendation X.509, 1988.
- [10] A. Mana, J. Lopez, J. J. Ortega, E. Pimentel, J. M. Troya,
“A framework for secure execution of software”,
International Journal of Information Security,
Volume 2, Issue 4, Springer, November 2004.
pp.99–112,
- [11] A. Monden, H. Iida, K. Matsumoto,
“A Practical Method for Watermarking Java Programs”,
The 24th Computer Software and Applications Conference (Compsac’00), Taipei, Taiwan, Oct. 2000.
- [12] J. Palsberg, S. Krishnaswamy, M. Kwon, D. Ma, Q. Shao, Y. Zhang,
“Experience with Software Watermarking”,
In: Proc. of the 16th Annual Computer Security Applications Conference, ACSAC 2000,
pp.308–316.
- [13] J. P. Stern, G. Hachez, F. Koeune, J.-J. Quisquater,
“Robust Object Watermarking: Application to Code”,
In: A. Pfitzmann, editor, Information Hiding ’99,
Vol. 1768 of Lectures Notes in Computer Science (LNCS), Dresden, Germany, 2000.
pp.368–378.
- [14] R. Venkatesan, V. Vazirani, S. Sinha,
“Graph Theoretic Approach to Software Watermarking”,
In: Proceedings of the 4th International Workshop on Information Hiding table of contents, 2001.
pp.157–168.
- [15] G. Wroblewski,
“General Method of Program Code Obfuscation”,
Ph.D. thesis, Wroclaw University of Technology,
Institute of Engineering Cybernetics, 2002.

Effect of WiFi systems on multimedia applications

ZOLTÁN GÁL, ANDREA KARSAI, PÉTER OROSZ

University of Debrecen, Center of Informatics
zgal@unideb.hu, kandrea@fox.unideb.hu, oroszp@unideb.hu

Keywords: IPv4, IPv6, IEEE 802.11b/g, IEEE 802.11a, WiFi, L2/L3 roaming, VoIP, codec, QoS, H.323.

At WiFi hot-spot deployment an essential question is whether 802.11b or 802.11g and/or 802.11a system should be installed. For this decision an efficiency analysis is needed beyond the economic and rational considerations. As IP phone systems are increasingly available in university environment, the analysis of practicability of WiFi phones during movement appears as an obvious object in indoor and outdoor environment as well. In this paper we focus on the analysis of the properties of multimedia applications (video, streaming, IP phone) operated over IEEE 802.11b/g/a WiFi systems. (In: 2006/6, pp.15–23.)

1. Introduction

Wireless data transmission mechanisms that belong to the IEEE 802.11 family are spreading widely in indoor and outdoor environments as well due to their mobility feature. When deploying hotspots several considerations are required to be taken about the applied technology. This is more than just an economical or rational issue, it requires efficiency analysis as well. The WiFi system is based on the ISM (Industrial-Science-Medical) frequency range that allows service providers to deploy and operate multiple hotspots in the same physical area independently from each other. In outdoor environment the different service providers use radio channels usually in uncoordinated way. Since ETSI standards are applied to the radiated microwave power the densely deployed systems may cause interference with each other.

In company or academic environment network users set up more and more requirements toward mobile WiFi devices (such as laptops, palmtops, intelligent mobile phones) to support multimedia services. Since IP phone systems are dynamically spreading in academic environment, we need to analyze the usability of WiFi phones during physical movement in indoor and outdoor environment. In the 2.4GHz ISM range the voice transmission feature of the WiFi IP phone greatly depends on the applied voice encoding mechanism. The 5GHz WiFi transmission has a special channel coding mechanism that is more effective than that of the 802.11g. Its transmission rate is very sensitive to the distance between the access point and the client. During physical movement the high compression ratio data transmission standards are more sensitive to the radio cell roaming than the algorithms with lower compression ratio.

We know from previous analyses that the quality of multimedia services on mobile terminals is heavily affected by the velocity of the terminal during physical movement [1]. On mobile terminals the quality of multimedia applications depends strongly on data-link layer events.

2. Overview of multimedia coding/decoding algorithms

Spectacular development of DSP (Digital Signal Processing) architectures in the last few years and researches on human speech recognition affected the improvement of voice encoding and decoding technologies [2]. The new codecs beyond the usual AD/DA conversion apply internal patterns to analyze the input audio signal and forward it as a minimal bandwidth data stream.

PCM

The simple PCM (Pulse Code Modulation) audio is encoded according to the ITU-T G.711 standard [3]. The 64 kbps PCM voice is compressed by μ -law or A-law procedures that convert the 12 or 13 bit sample to an 8 bit one using logarithmic scale. Benefits are simplicity, low complexity, low latency, good sound quality. Disadvantage is the high bandwidth requirement.

ADPCM

The ADPCM (Adaptive Differential Pulse Code Modulation) is also a common compression solution that is defined in the ITU-T G.726 standard. It uses 4 bit samples that are transmitted at 32 kbps. Unlike PCM these 4 bit words do not encode the amplitude of the voice rather the difference of amplitudes and the alteration rates. The algorithm applies a very simple linear estimate. Benefits are simplicity, low complexity, good sound quality, low latency, multiple coding rates. Disadvantages are relatively high bandwidth requirement, poor sound quality at lower bandwidths.

AMR-NB

The AMR-NB (Adaptive Multi Rate – Narrow Band) is mostly used on GSM and UMTS mobile networks. The algorithm supports eight compression ratios (4.75; 5.15; 5.90; 6.70; 7.40; 7.95; 10.20; 12.20 kbps). The algorithm can switch between these ratios at any time which

is profitable on IP networks. The sender can alter the outgoing bandwidth at any time based on the real time statistics provided by RTP: according to the RTP signaling the encoder compresses the forthcoming samples at the newly changed rate and the decoder can decompress it in the same way. The 20 ms frames are encoded by ACELP algorithm using 5 ms *lookahead* value. Benefits are simplicity, relatively low complexity, low bandwidth requirement, good sound quality, low latency, multiple coding rates. Disadvantages are few implementations, lack of open source code.

AMR-WB

The AMR-WB mechanism is used by the G722.2 encoder that is optimized to wide bandwidth. It uses ACELP algorithm and encodes 7 kHz audio sampled at 16 kHz. It adaptively alters encoding rates (23.85; 23.05; 19.85; 18.25; 15.85; 14.25; 12.65; 8.85; 6.6 kbps). The encoder uses 20 ms frames and 5 ms *lookahead* buffer. Benefits are very good sound quality, low latency, multiple encoding rates. Disadvantages are high bandwidth requirement for good sound quality, moderate computing complexity.

RTP protocol

The RTP (Real-Time Protocol) provides point-to-point application layer transport service for real-time (audio, video) traffic, therefore it uses services like PDU identification, sequencing, time stamping and transmission control. It is used commonly over UDP using its multiplexing and checksum generation services and it is sometimes used over TCP as well. RTP cannot guarantee neither packet arrival nor correct packet sequence. RTP is optimized to a variable and overloaded network condition typical for IP networks. RTP transports content data to one direction while it uses the duplex channels of RTCP to relay control information that includes quality parameters as well. RTP can perform time recovering, sender identification, content identification, sequencing and loss detection. QoS, resource allocation, packet loss recovery and on-time delivery are not related to RTP.

Classification of voice encoders/decoders

PCM and ADPCM belong to the family of waveform codecs that use the redundant characteristics of the waveform. Compression techniques developed in the last 10-15 years are focusing on the voice source characteristics. These compression codecs create the simplified parameters of the original voice source that results in smaller bandwidth. They are called source codecs including LPC (Linear Predictive Coding), CELP (Code Excited Linear Prediction) and MP-MLQ (Multi-purpose Multilevel Quantization) procedures. Advanced codecs substitute the human voice source with a mathematical model and they transmit the representation of the voice instead of the compressed voice. The most common telephone voice encoding and packet switched voice standards are the following:

- *G.711*: 64 kbps PCM voice encoding technique used in the conventional digital PBX centers and networks.
- *G.726*: Uses 40, 32, 24, 16 kbps ADPCM encoding. ADPCM voice transmission is recommended between packet switched and conventional PBX networks.
- *G.728*: Low latency fluctuation version of CELP that transmits voice at 16 kbps. CELP voice has to be trans-coded to public telephone format in order to set up communication with a public endpoint.
- *G.729*: By using CELP compression it converts the voice audio to an 8 bit data stream. Two subversions exist that significantly differ from each other in processing complexity both providing 32 kbps ADPCM quality voice.
- *G.731*: Compresses voice or multimedia audio consuming very low bandwidth. As part of the H.324 protocol family it operates at 5.3 kbps and 6.3 kbps. The former applies CELP, the latter uses MP-MLQ technology while both provide good voice quality and further flexibility for the system.
- *GSM*: The GSM (Global System for Mobile Communication) standard of ETSI I-30036 is widely used in European mobile networks for voice and low bandwidth data communication. Full rate GSM operates at 13 kbps by using RPE (Regular Pulse Excited) encoder at 8 kHz sampling rate. Half rate GSM requires 7 kbps bandwidth at 5 kHz sampling rate. The input voice is split up into 20 ms frames it makes eight short term approximations for each of them. Furthermore each frame is divided into 5 ms sub-frames where the encoder calculates latency and gain for the long term approximator. Finally it quantizes the rest of the signal in each sub-frame. The GSM encoder generates good quality voice however the G.728 (CELP) encoder outperforms it by its higher bandwidth. GSM encoder requires low processing time. Benefits: simplicity, relatively low complexity, low bandwidth, low latency, open source. Disadvantages: G.728 outperforms it in quality/bandwidth ratio.

Nullsoft Video protocol

The Nullsoft Video (NSV) format is a bit-stream that is able to provide joint packaging for audio and video data [4]. It complies with all major video and audio compression mechanisms. As it is a bit-stream format, it does not need to download the entire data file before playing. It can provide streaming service as reliable synchronization occurs at any point of the stream. Secondary data channels may provide multiple sound-subtitle- or data flows.

NSV file consists of two major parts: optional header and mandatory bit-stream. All multi-bytes integers are stored in LSB format where the least valuable byte is the leftmost byte. Therefore a 4 bit and a 20 bit number will be stored in 3 bytes long. The voice and video data packet are transmitted in one frame. The voice

may follow or precede the video data. The number of channels for additional information (e.g. title, 16:9/4:3 screen ratio, secondary audio channel, etc.) is limited to 15.

3. Characteristics of the VoIP network

After compressing the voice and converting it to data the stream will be transmitted over the IP network using RTP protocol. In VoIP networks we must consider the latency and the bandwidth as well. Bandwidth requirements are critical as they depend on not just the selected codec but the overhead of the layer protocols (IP, UDP) [5]. The latency is affected by the propagation speed of the signal the buffer handling mechanism of the sender and receiver as well as the encapsulation delay.

Bandwidth requirements on VoIP network

The operation of voice conversation over IP networks is affected by several parameters. Bandwidth requirement of the applied codec can be in the range of 3...64 kbps. The voice PDU usually is not longer than 20 bytes while the L2 (Ethernet) and L3 (IP) layers add significant overhead. Therefore the effective bandwidth requirement is affected by these overheads [6]. In order to simplify the problem various solutions had been introduced. By using Voice Activity Detection (VOD) the sender can interrupt the stream if the signal of the local analogue source decreases to a certain threshold level. The bandwidth requirement is decreased to its half as in human conversations a person listens to the other in half of the times. This solution claims more attention in determining the appropriate moments for switching on and off otherwise it can cause content loss. However the total silence can be disturbing as well. Comfort noise is usually applied to eliminate this problem that is perceptible at the pair of the non-speaking person as locally generated white noise.

Advanced systems reproduce remote background noise in the silence period of the remote person. Another solution is the compression of the RTP PDU header. As the RTP PDU may contain duplicated or redundant information routers alongside the route compress the header therefore the bandwidth requirement of voice can be significantly decreased. In most common LAN/MAN technological environment the required physical bandwidth is shown in *Table 1*. IP/UDP/RTP generates 40 bytes and Ethernet makes 14 bytes overhead.

Table 1. Comparison of the VoIP/channel bandwidth

Algorithm	Voice bandwidth [kbps]	Codec latency [ms]	Voice PDU size [B]	Voice rate [PDU/sec]	L2 PDU size [B]	Physical bandwidth [kbps]
G.729	8.0	15.0	20	50	74	29.60
G.711	64.0	1.5	160	50	214	85.60
G.723.1	6.3	37.5	30	26	84	17.47
G.723.2	5.3	37.5	30	22	84	14.78

Each voice connection means two call streams while video connection creates four or six call streams simultaneously.

Latency on VoIP network

In VoIP design a generally accepted rule is to keep the endpoint to endpoint latency under 150 ms. Transmission latency of today's media is not perceptible for human ear however together with handling latency it may cause perceptible distortion. On user side the latency tolerance is about 250 ms. Voice stream with higher latency cause interference with the natural voice stream therefore they may quench each other. Handling latency has effect on conventional line-switched telephone networks also but its importance is higher at packet switched transmission due to buffering. It should be kept under 150-200 ms at latency design. Latency of the G.729 standard is about 20 ms what was designed in consideration of future demands as well. A VoIP product generates a frame in every 10 ms on average then it orders them in pairs and puts into a packet therefore the value of latency will be 20 ms. In packet switched networks latency is generated by putting the current packet into the outgoing queue and the latency of the queue. These values are device-dependent and typically not exceed 30 ms.

VoIP applications are sensitive not only to latency but to its alteration as well. Unlike the conventional telephone networks in packet switched transmission the value of latency can fluctuate by the network traffic load. Jitter is a short term alteration of the latency that is the fluctuation between the expected and real packet arrival time. Devices compensate it by using *playout* buffer therefore gaps in the voice stream can be avoided. It increases the total latency of the system. The buffer can be fix-sized or adaptive at different devices. In case of VoIP jitter is the most significant quality obstructive parameter. Packet switched voice transmission usually passes through systems with different latencies and transfer parameters what results in poor quality. General feature of these applications is a large size receiver buffer with at least 1 second voice buffering.

QoS on VoIP network

In packet switched networks voice quality is largely determined by the latency and the jitter of the network therefore QoS parameters have high priority in network design. Further important issues are the segregation of the voice traffic from other network data and the protection of critical data against the possible high bandwidth of the voice traffic. Elements of the effective QoS design are: required bandwidth, packet loss, latency

and jitter. These factors are ensured by the following techniques:

- *Control strategy*: traffic limitation that means packet drops when the traffic between two networking devices exceeds a given threshold. This parameter can be configured at input and output side as well. Typical example of such techniques is RED (Random Early Detection) and WRED (Weighted RED). These techniques identify packets that can be dropped when necessary.

- *Traffic design*: provides equal input and output packet rate for buffering. Unlike control strategy it tries to avoid packet drop whereas it increases the latency and jitter caused by buffering.

- *Call set-up control*: controls rejection of the bandwidth requirement of the applications. In VoIP networks RSVP (Resource Reservation Protocol) can be applied to reserve bandwidth for a call. The H.323 gatekeeper can limit this reserved bandwidth.

- *Queues/scheduling*: used at buffering by discovering the priority of the packets. One buffer can be configured for latency-sensitive packets and another for data packets. IP/RTP priority queue is common mechanism in VoIP.

- *Tagging*: There are different techniques to tag packets that require special treatment. In VoIP networks packets can be tagged by IP preference bits (IP header ToS field) for example. The packet tagging mechanisms are important to preserve the internetwork QoS parameters of the packets.

- *Fragmentation*: Further fragmentation of large packets can be enabled on some devices before transmit-

ting on a low bandwidth link. This feature protects voice packets against high latency required for the transmission of large packets. Therefore the voice packet can be put among the fragments of a large data packet.

4. Measuring environment and measured values

For our measurements we applied wireless devices (access points and mobile terminal) that support IEEE 802.11b/g and IEEE 802.11a transmission mechanisms as well. We analyzed the effects of L2 roaming that was occurred during on foot physical movement of the mobile terminal in our indoor test network as shown on *Figure 1*. Mobile terminal passed at 5-6 km/h (1.4-1.7 m/s) velocity parallel with the straight line that links the two access points. Within one measurement period (TS_i) the mobile terminal gets from the micro cell of AP₁ to the micro cell of AP₂ then backward it reaches to the cell of AP₁ again. As a multimedia service video streaming and IP phone were run on the mobile terminal. MT was a laptop and WinAmp multimedia application was run for streaming (TCP) and SoftPhone application for voice conversation (UDP) analyses.

For TCP traffic the wired node was a streaming server where we downloaded Nullsoft (NSV) format multimedia contents from with different bandwidth. In order to measure UDP traffic SoftPhone application was run on the mobile terminal and the wired node as well and voice conversation was generated between them. Voice en-

Figure 1. The measurement environment

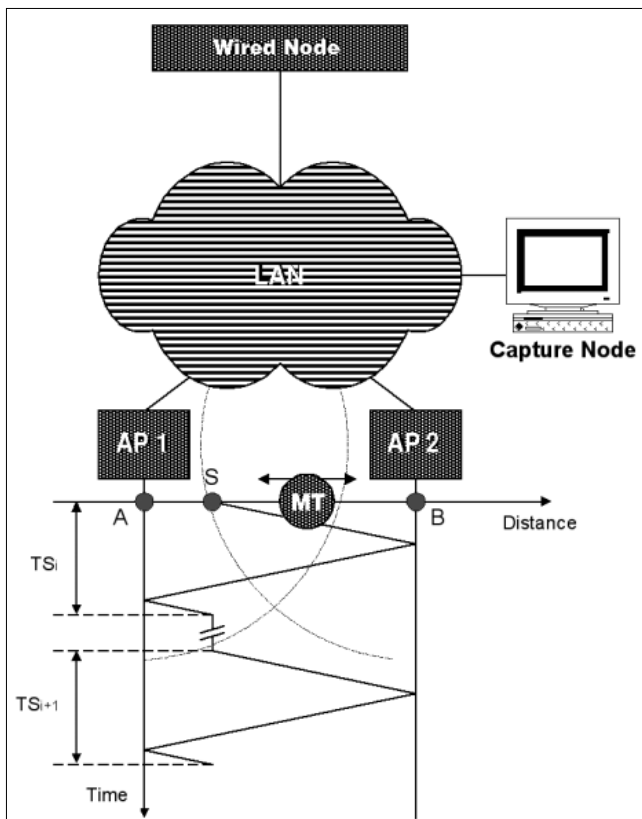
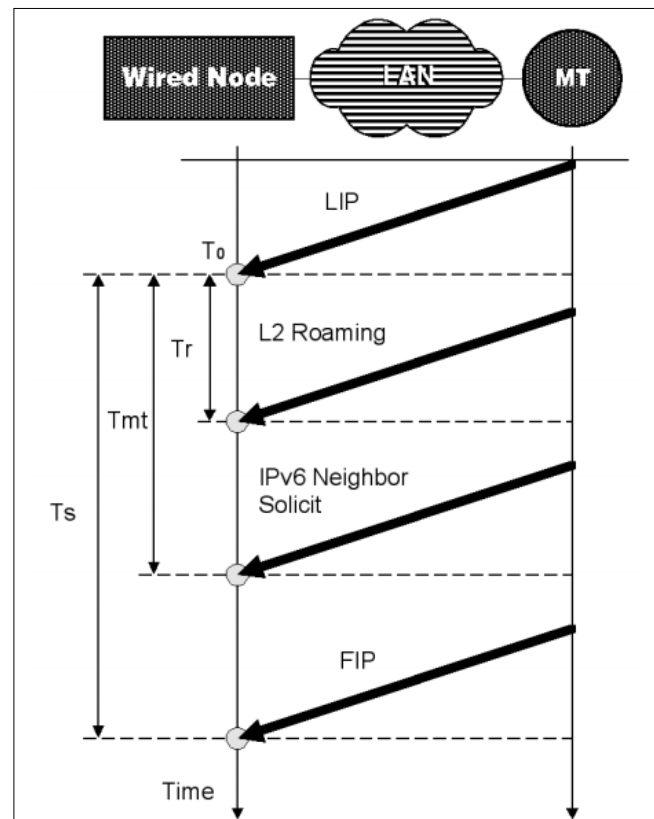


Figure 2. The measured time intervals



coding mechanism was selected on the Phone Center located inside the wired LAN. Bandwidth values of the streaming contents were the following: 80, 150, 300, 500 kbps. The applied voice encoding mechanisms were: G.728 (16 kbps), GSM (29 kbps), G.711 (80 kbps), Wideband (272 kbps). We set the *data retry* parameter of the access points to fixed 32 for TCP and UDP while we adjusted the *beacon period* between 20, 50 and 100 ms. Receiver buffer of the WinAmp application was fixed to 1000 ms. We performed twelve measurements for TCP, the same amount for UDP with each IEEE 802.11 standard, creating in this way seventy-two captured data files.

The two access points (AP₁, AP₂) were connected to the same VLAN on a L2 switch. Ethernet frames from the VLAN appear on the wired network were passed to a traffic capture node by mirroring to a dedicated physical port that used tcpdump to capture and store the data in libcap format file. Afterwards we analyzed the measured values with Ethereal v0.10.4 protocol analyzer therefore we can determine the time intervals that affect the quality of the applications. Radiated microwave power of the access points were set to 5 mW for IEEE 802.11b/g and 11dB for IEEE 802.11a. Physical distance between the two APs was 50 meters, traffic of the MT was unencrypted and association type was open. In each TS_i measurement (i = 1,2,...72) the MT started from point S then passed on B, S, A points and finally reached S again.

In order to determine the times that affect the quality of applications we identified the T₀ moment (Figure 2) in every capture data file. This is the arrival moment of the LIP (Last Important Packet) received by the wired node before L2 roaming. This is the last content data of the MT before roaming. Descriptions of LIP and FIP packets are shown in Table 2.

L2 roaming event occurs within the Tr time as we described in our previous paper [1]. Tr time is identified by the arrival of the roaming frames to the new AP. On the mobile terminal IPv6 was also used that can perceive the recovery of the data link layer of the protocol stack and it immediately initiates the discovery of the neighbor nodes. Tmt time denotes the relative moment when the MT can restart LLC transmission. We used this feature of IPv6 to identify the frames sent during accurate roaming event as we can experience multiple cell changes during the physical movement within indoor points S->B->S->A->S due to the Rayley fading effect. The Ts time is the operational latency of the multimedia connection. This is perceptible directly by the user and its high value may produce gaps in service and loss of the connection. FIP packets are used to identify the Ts value.

Table 2. Meaning of the significant packets

Transport layer	Important packet	Description
TCP	LIP	Last ACK packet (60 bytes) from MT sent to the server before L2 roaming
TCP	FIP	First ACK packet (60 bytes) sent to the server after L2 roaming
UDP	LIP	Last UDP packet (60 bytes) from MT sent to the wired node before L2 roaming
UDP	FIP	First UDP packet sent to the wired node after L2 roaming

5. Analysis and explanation of measurement results

Comparison and analysis of the measurement results gave us the possibility to draw important conclusions. Different IEEE 802.11 standards show different behaviors at roaming events in indoor environment [7]. The roaming process is greatly depends on the beacon period (Tb) value which is a configuration parameter of the access point [8]. MT learns the beacon period of the AP [9] from a signal in the beacon. As soon as the MT does not receive eight consecutive beacons roaming event will be initiated [1]. By continuously monitoring the incoming beacon frames the MT perceives the loss of radio signal quality and initiates a roaming process.

Measurement results for TCP traffic are shown in Figures 3, 4, 5 while UDP traffic values are displayed in Figures 6, 7, 8.

- When the beacon period was decreased step by step from 100 ms down to 50 ms and 20 ms the MT perceived more rapidly the alteration of S/N ratio, therefore it became more sensitive to the change of environmental conditions during the physical movement. In this way the roaming times rise up from 0.5-2.8 s to 0.1-14.9 s and then fall down to 1.5-2.9 s at each streaming technology. At first TCP dropout rises up from 2.5-17 s to 2.4-19.8 s than falls down to 1.8-7.9 s. Consequently the Tb=20 ms beacon period is better then the 100 ms value. It is a useful establishment about the beacon configuration. However very low Tb values may cause multiple roaming events in indoor environment due to the multipath signal propagation that induces repeated TCP dropout.

- In the case of IP phone communication, for a given voice coding technique the adjustment of beacon period from 100 ms to 50 and 20 ms involves the decrease of the cell change time from 0.1-44.5 sec to 0.1-12.5 sec while UDP dropout decreases from 0.2-49.8 sec to 0.2-19.9 sec that indicates the advantage of Tb=20 ms value.

- In indoor environment the IEEE 802.11 technologies show different behaviors regarding the beacon period. During streaming the IEEE 802.11a resumes the connection within a longer period of time. Regarding roaming performance it is followed by the IEEE 802.11b and the 802.11b that has the most advantageous features in case of indoor roaming.

- IEEE 802.11a produces very high latency therefore connection is dropped even in wide bandwidth voice communication while IEEE 802.11g shows the best reaction time where the connection dropout can be kept under 4 sec. This dropout is affordable be-

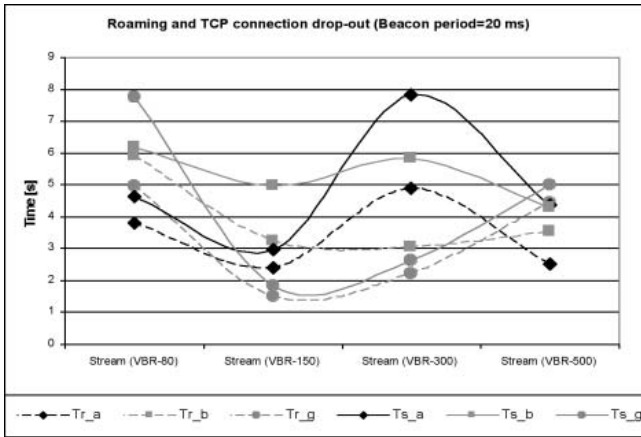


Figure 3. The streaming behavior ($T_b=20$ msec)

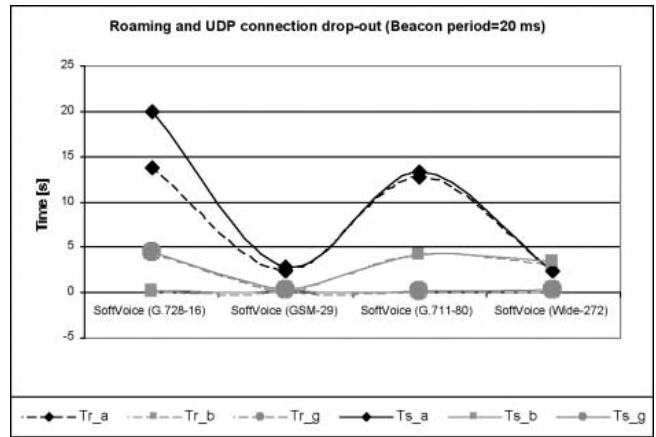


Figure 6. The IP Phone behavior ($T_b=20$ msec)

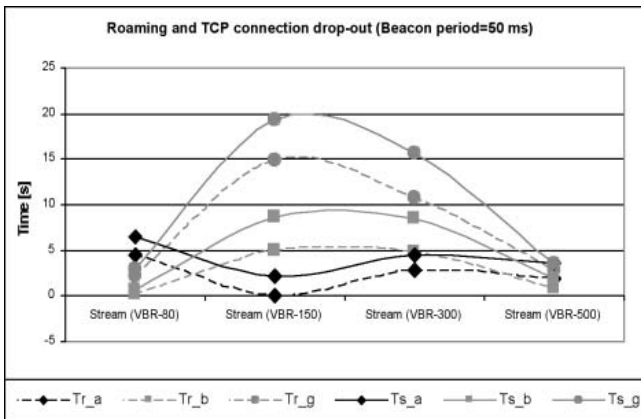


Figure 4. The streaming behavior ($T_b=50$ msec)

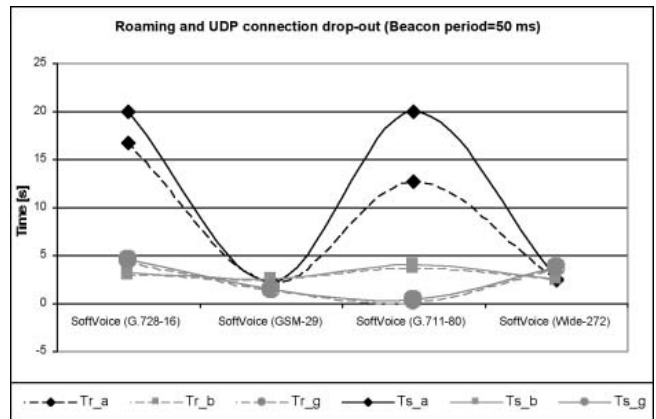


Figure 7. The IP Phone behavior ($T_b=50$ msec)

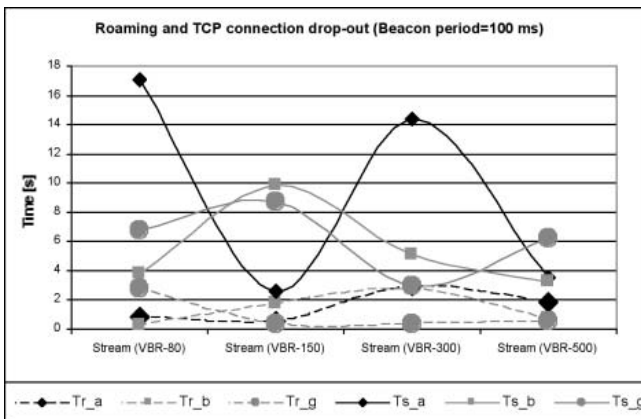


Figure 5. The streaming behavior ($T_b=100$ msec)

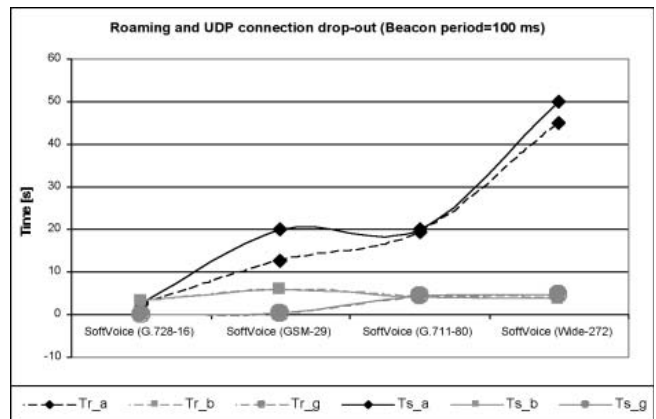


Figure 8. The IP Phone behavior ($T_b=100$ msec)

cause it may be a relatively rare event in mobile IP phone systems if users are notified previously.

- Application dropout times have different characteristics depending on the bandwidth of the stream. NSV application at 150 kbps depends least on the radio technology at $T_b=20$ msec while it depends the most on the technology at $T_b=50$ ms. With $T_b \leq 50$ ms NSV programmes at 80 and 500 kbps are least affected by the radio technology.

- IP phone communication with $T_b \leq 50$ ms provides the least dropout by using GSM encoding that is optimized to mobile environment. GSM provides weaker sound quality than G.728 however it adapts more ef-

fectively to roaming situations. G.711 greatly depends on the roaming mechanism of the radio technology that comes from the features of the PCM designed for conventional wired environment.

- After roaming with high beacon period streaming resumes within a 4-11 sec period while with $T_b \leq 50$ the latency of the TCP connection is 3 sec that can be observed at the time difference $T_s - T_r$.

- At IEEE 802.11b/g latency of UDP traffic (IP phone) is under 0.5 sec for each voice encoding technique. At IEEE 802.11a the resuming of UDP traffic is significantly delayed (3-7 sec) after the MT arrives to the new radio cell.

- IEEE 802.11a technology performs its best at $T_b = 50$ ms. IEEE 802.11g provides the best data-link service for streaming at $T_b = 20$ ms.
- For IP telephony IEEE 802.11g gives the best overall performance independently from the voice encoding mechanism and the beacon period. It is followed by the 802.11b and the 802.11a that shows the disadvantageous behavior.
- Flexibility order of the voice encoding mechanisms for data-link layer dropout (descending order): GSM, Wideband, G.711, G.728.

6. Conclusions

In this paper we analyzed the characteristic features of multimedia applications (streaming, IP phone) during indoor roaming events operated on IEEE 802.11a, 802.11b and 802.11g transmission technologies. For TCP measurements NSV format streams were sent towards a laptop moving on foot in indoor environment. For UDP measurements IP SoftPhone were run on the laptop while it had a voice conversation with a wired IP phone. Measurements were performed for the common encoding mechanisms.

Based on these measurements we can establish that the beacon period of the access point has significant impact on the roaming process. Different IEEE 802.11 technologies show different behaviors for both TCP and UDP traffics. Beacon period also has effects on them.

For mobile indoor wireless streaming terminal IEEE 802.11g with ≤ 50 ms beacon period provides the best performance. For mobile WiFi IP phones in slightly loaded 802.11g indoor environment good quality of service can be experienced. Roaming may cause 2-3 sec of dropout in best case that is an acceptable value for users if they are previously notified. GSM voice encoding flexibly adapts to dropouts in the data link layer, it is followed by the wideband encoding despite its higher bandwidth requirement compared to G.711 and G.728.

In the future transport layer services with low dropout (< 250 ms) and fast roaming convergence are required to design in order to provide continuously acceptable operational quality for multimedia services in indoor mobile WiFi systems. Performance of the H.323/SIP mobile IP video conference and web collaboration services in mobile environment have to be evaluated as well.

References

- [1] Zoltán Gál, Andrea Karsai, Peter Orosz, "Evaluation of IPv6 Services in Mobile WiFi Environment", Selected Papers of Info-Communication-Technology, Volume LX., 2005, pp.47–54.
- [2] Dodd, Annabel Z., The Essential Guide to Telecommunications.
- [3] Newton, Harry, Newton's Telecom Dictionary, <http://www.harrynewton.com/>
- [4] Justin Frankel, "Nullsoft Video (NSV) Format 2.1 Specification", <http://ultravox.aol.com/NSVFormat.rtf>
- [5] Jonathan Davidson, "Voice over IP Fundamentals", <http://www.ciscopress.com/>
- [6] Cisco Documentation DVD Home Page: "Voice/Data Integration Technologies", http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/voicdata.htm
- [7] WiFi Alliance: "Wi-Fi Certified for WMM- Support for Multimedia Applications with Quality of Service in Wi-Fi Networks", http://www.wi-fi.org/files/uploaded_files/wp_1_WMM%20QoS%20In%20Wi-Fi_9-1-04.pdf
- [8] SpectraLink, Co.: "White paper: Deploying netlink wireless telephones, best practices", http://www.spectralink.com/files/literature/NetLink_Best_Practices_122105_01.pdf
- [9] 3Com, Co.: "White paper: Deploying 802.11 Wireless LANs", http://www.3com.com/other/pdfs/products/en_US/wireless_lans_wp.pdf

Algorithm for formant tracking, modification and synthesis

TAMÁS BŐHM, GÉZA NÉMETH

BME Department of Telecommunications and Media Informatics
{bohm, nemeth}@tmit.bme.hu

Keywords: formant tracking, formant synthesis, linear prediction, speech character modification

Precise formant tracking has been a challenge for researchers in speech processing for long. In this paper, the authors present a method to track and modify formants in speech signals. It is an efficient tool for analyzing and altering the spectral content of speech, furthermore it provides an opportunity to modify timbre and voice quality. The method is based on the linear prediction model. (In: 2006/8, pp.11–16.)

1. Introduction

During the production of voiced speech sounds the vocal folds vibrate in a quasi-periodic manner. The excitation signal produced by this oscillation is modulated by the resonator system of the vocal tract (pharyngeal, nasal and oral cavity): while harmonics near the resonant frequency are boosted, other harmonics are attenuated. These vocal tract resonances are referred to as *formants*.

The resonant frequencies are manifested as local maxima in the vocal tract transfer function. Besides their center frequency, formants can be characterized by their bandwidth and amplitude. The former refers to the width of the boosted frequency region in the transfer function – where the function value does not fall more than 3 dB below the local maximum. The latter is the value of the function at the peak [1].

Although formants and their time course (the *formant tracks* or *formant trajectories*) are clearly visible on the spectrum and on the spectrogram for the human eye, automatic formant measurement and tracking is far from trivial.

There is a definite need for exact formant tracking together with the ability to modify formant trajectories both for research purposes and for specific applications. The latter include the modification of the voice character (such as dialect transformation, speech correction or timbre modification) and smoothing the formant trajectories of waveforms generated by concatenative text-to-speech systems. Such algorithms may also be applied to alter the speaker-specific characteristics of speech so that the listener recognizes another speaker uttering the same sentence.

The prospective applications require a method that can re-synthesize speech after altering the formant structure. This can be achieved only by employing a precise formant extraction algorithm. The problem of re-synthesis for our targeted applications has not been extensively studied, we could not find an appropriate solution in the literature.

Formant extraction has been intensively studied during the past decades. Traditional methods employ peak finding on some kind of non-linearly smoothed spectra [2][3]. One way of doing this is cepstral spectrum smoothing when the maxima due to periodicity (corresponding to glottal excitation) are removed from the cepstrum and it is then Fourier transformed [1]. Rabiner and Schafer's method combines this procedure with "analysis by synthesis" [4]. Another approach is the use of various filter banks [5].

Some techniques borrowed from speech recognition can also be applied to formant tracking. For example, methods based on Hidden Markov Models (HMMs) [6] and Line Spectrum Pairs (LSP) are common. The latter is an implementation of the linear prediction (LP) analysis that is performed in the frequency domain instead of the time domain and the calculated coefficients follow the time course of the high-amplitude regions in the spectrum (that roughly correspond to the formants).

In this paper we report a highly accurate method of formant analysis, tracking and modification. In order to demonstrate the algorithm, we created a graphical computer program that is capable of producing various displays of the formant tracks. Section 2 outlines our approach and Section 3 gives the details of the algorithm. Section 4 is a description of the graphical demo program while Section 5 summarizes our findings.

2. Basic concepts

2.1. Linear prediction-based spectrum

As it was discussed earlier, formants are local maxima in the speech spectrum. The spectrum can be calculated by means of Fast Fourier Transform (FFT) but this gives a function with numerous local maxima and minima. It is not straightforward to reliably find the peaks of such a spectrum. Linear prediction-based spectrum calculation [2] is more common for the purpose of formant extraction because it has several advantages over FFT:

- The "resolution" of the spectrum can be set by the order of prediction (that corresponds to the number of poles in the transfer function).
- Linear prediction gives a good estimate of the spectrum primarily at the peaks (that are of interest here).
- It is capable of producing acceptable spectrum estimates even for short speech segments.

The transfer function estimated from the linear prediction coefficients:

$$H(z) = \frac{1}{1 - \sum_{k=1}^p \alpha_k z^{-k}} \quad z = r \cdot e^{j2\pi \frac{f}{f_s}}$$

This results in a spectrum estimate much smoother than the FFT while formants are not distorted.

2.2. Formant extraction

Two ways of formant extraction have been most frequently discussed in publications: spectrum-based and pole-based. While the former uses amplitude or phase spectrum to find the local maxima corresponding to formants, the latter calculate with the poles in the z-domain.

One spectrum-based method is the McCandless algorithm that detects peaks in the logarithm of the absolute spectrum [7]. Christensen, Strong and Palmer developed a similar procedure but they apply peak finding on the negative second derivative of the log spectrum [8]. Yegnanarayana proved that the first derivative of the complex spectrum phase is showing noteworthy similarity to the amplitude spectrum [9]. Employing the second derivative of this function instead of the log spectrum allows more accurate formant extraction. The method of Reddy and Swamy is calculating simultaneously in the f- and the z-domain so it can distinguish between formants near to each other [10]. Although the above-mentioned methods have been implemented and thoroughly studied, none of them fits our needs.

Our formant tracker is pole-based, similar to the one developed by Slifka and Anderson for speaker transformation [11]. The H(z) transfer function gives an all-pole (i.e. with no zeros) model for the vocal tract. The poles of this function correspond to the resonances of the system. The poles are the roots of the polynomial in the denominator of the transfer function:

$$H(z) = \frac{1}{1 - \sum_{k=1}^p \alpha_k z^{-k}} = \frac{1}{\prod_{i=1}^p (1 - p_i z^{-1})}$$

where α_k are the linear prediction coefficients. Formant frequencies and bandwidths can be obtained from the $p_i = r_i \cdot e^{j\omega_i}$ form of the poles:

$$F_i = \frac{f_s}{2\pi} \varphi_i \quad B_i = \frac{f_s}{\pi} \ln\left(\frac{1}{r_i}\right)$$

The results are highly accurate but not all poles correspond to formants (e.g. real poles cannot be formants). Such poles are due to lip radiation or background noise.

3. Algorithm

The input data is pitch synchronously segmented speech with sound boundary labels. The phonetic transcript of the utterance can improve the accuracy of the results. We can distinguish two separate stages of the processing: analysis and synthesis. The former one refers to the tracking of formant trajectories and its output is the formant data throughout the utterance and some side information (such as the linear prediction residuals and gains). During synthesis, the modification of the formants and the re-synthesis of speech is implemented, the output is a new waveform.

3.1. Analysis

3.1.1. LP analysis and calculating the poles

In order to apply LP analysis locally in time that is essential for formant tracking, we need to calculate the LP coefficients (LPCs) for every pitch period separately. In order to reduce spectral distortion, our algorithm is determining LPCs for two adjacent pitch periods (a time frame) instead of one and employs Hanning windowing. The window is shifted from pitch period to pitch period. For unvoiced sounds we create "virtual pitch marks" with a constant time step.

First we obtain the PARCOR coefficients by the method of Burg, then we convert these to linear prediction coefficients in order to calculate the transfer function [2].

LP analysis and synthesis do not guarantee that the energy of the output signal is the same as the energy of the input. To avoid this kind of distortion, it is reasonable to store the energy for each time frame that can be used to restore the original level on the output. If we normalize the energy for the length of the frame, we can also use this value for silence detection. The LP residual signal should also be stored for lossless coding.

As a next step, we need to calculate the poles of the system: the roots of the transfer function denominator. Although the polynomial in the denominator has only real coefficients, the roots can also be complex. This prohibits using the Newton-Raphson and Brent methods. We chose to employ the Laguerre root finding procedure [12] instead.

We can obtain all the roots by applying the Laguerre algorithm iteratively: one run results in either one real root or a complex conjugated pair of roots. After dividing the polynomial with this/these, we run the root finding again.

We can calculate the formant data (frequency, bandwidth and amplitude) by using both the formulae in Section 2 and the spectrum. Note that in synthesis time we should avoid using these data: the calculation is more accurate if we implement formant frequency changes directly on the corresponding pole.

Poles without a corresponding formant also need to be stored for re-synthesis.

3.1.2. Mapping of poles to formants

Not all the poles have a corresponding formant.

In order to discard these poles from further analysis, several criteria should be examined:

- The formant frequency needs to be higher than the fundamental frequency.
- The absolute value of the pole needs to reach a threshold so that we limit the bandwidth of the formant.
- The energy of the time frame needs to be higher than an energy threshold (silence detection).
- Poles with a real part that is zero or near zero can be excluded (these may be due to low frequency narrowband noise).

Only poles meeting all four criteria are considered formants. Formant data (frequency, bandwidth and amplitude) can be calculated from the selected complex conjugated pole pairs.

The above criteria can only exclude the evidently incorrect results and give a first estimate of the formant-pole mapping. The final mapping is obtained by applying continuity constraints on the formant trajectories.

3.1.3. Formant trajectories

Thus the next step is to arrange the formant candidates into continuous formant tracks by using the initial formant-pole mapping and sound boundary information. We need to map the formants of a time frame to the formants of the next time frame in a way that a continuous formant track should emerge. We map a formant candidate to the formant track whose last assigned formant is nearest in frequency. Formant tracks that have no or minimal collision with each other can be merged. Trajectories that are running parallel near to each other are also merged into one track. Extremely short trajectories are discarded.

There is no point in applying continuity constraints before and after obstruents (stops, fricatives and affricates) because the production of these sounds implies such articulatory movements that can cause a quick change in the formants. If the phonetic transcript of the utterance is available then our method does not try to connect formants through these boundaries. If it is not available, every sound boundary is treated as a break point. This approach may lead to less accurate information on several sound transitions but it improves the general efficiency of the mapping.

3.2. Synthesis

As the first step, the input of the re-synthesis procedure is created from the output of the analysis, so we alter the formant tracks. The way of mapping is always determined by the actual application.

For example, we might use some kind of interpolation in order to spectrally smooth the output of a concatenative TTS system. Note that by modifying the trajectories we move the poles on the z plane.

The second step is re-synthesis. The modified formant frequencies give the new pole locations (by using the same formulae as in the analysis). We construct the polynomial for the denominator of the transfer function

from the poles. The linear predictive coefficients are obtained as the coefficients of the polynomial. Using these and also the stored residual signal, linear prediction synthesis can be performed. As a last step we restore the energy of the time frames. The result is a new waveform with modified formant tracks.

4. Results

Evaluation was done for test utterances in Hungarian and separately for the analysis and synthesis stage.

4.1. Analysis

The accuracy of formant analysis and tracking was tested in three ways. First, reference spectrograms for testing was generated by a Kay Elemetrics CSL 4300B digital signal analyzer at the Linguistics Institute of the Hungarian Academy of Sciences. These spectrograms were compared manually with the formant tracks obtained by our algorithm. Second, measured formant frequency values were compared with published data [13].

Finally, we measured a mapping error rate as defined in [14]. For this purpose the formant tracks produced by the demo application (that is to be described later) were compared to spectrograms. The test set consisted of 29 two-word Hungarian recordings by a male speaker. A mapping error in the first three formant tracks was found in only two cases (6.90%) – in one case the third while in the other case all the three formants were mapped incorrectly.

According to [14] an algorithm using nominal formant frequency values achieved a mapping error of 3.62–3.99%. Although this is lower than our error rate, the algorithm presented in this paper does not rely on predefined, typical formant frequencies so it is independent of the physiological attributes, gender and language of the speaker. The paper cited above reports an error rate of 13.04% for a method with similar properties.

An example of the output is given in *Figure 1*. The utterance was “jaj hajít”. According to the figure, formants for voiced sounds were generally well detected, even after the /h/-vowel transition in the second word. This case was highlighted as problematic in [14] for formant analyzers that do not use nominal formant frequencies.

4.2. Synthesis

The synthesis capabilities of the method are evaluated in the context of prospective applications since the modification method is designed to fit these. Evaluation and fine tuning of the system is underway. We report the results of two simple initial experiments here.

By changing the formant structure, we can modify a recorded vowel to another sound. As an example, Hungarian “fésű” (fE:SU) was transformed to “fásű” (fA:SU) by raising the first and lowering the second formant track along the frequency axis. Such a formant modified re-

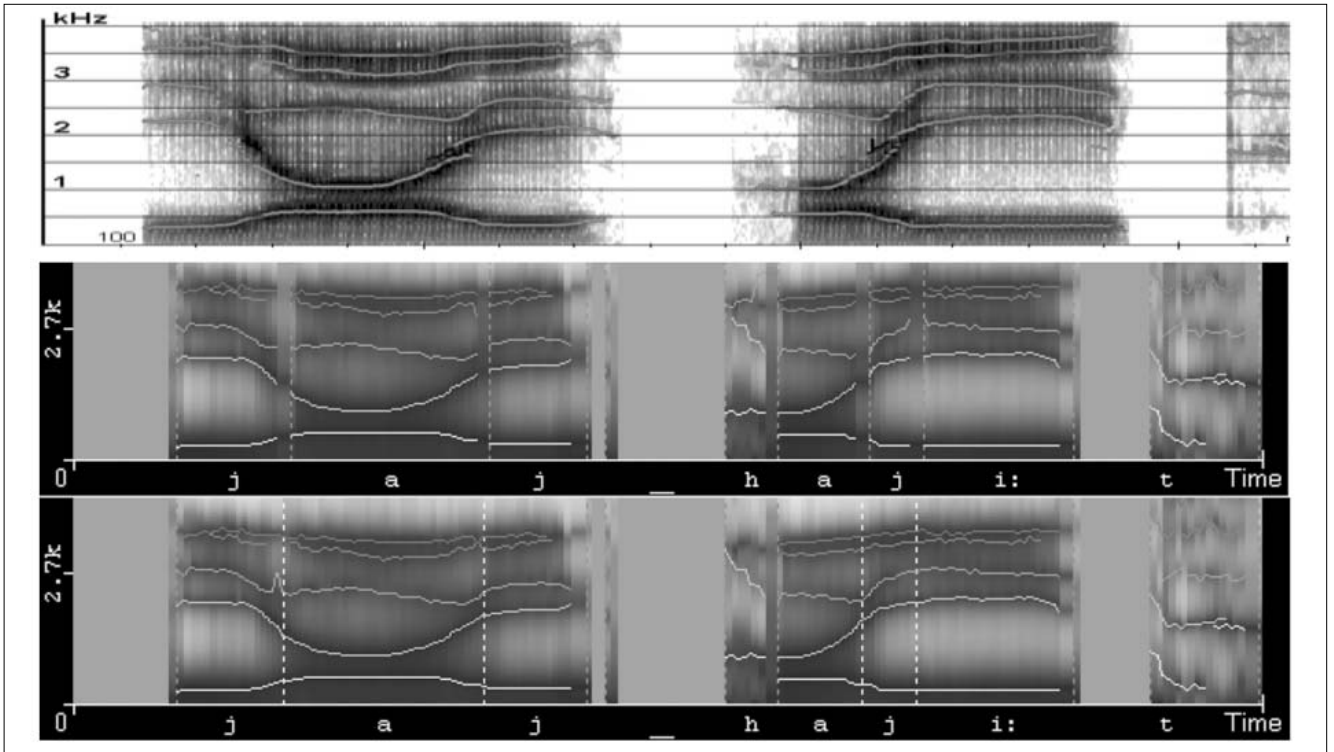
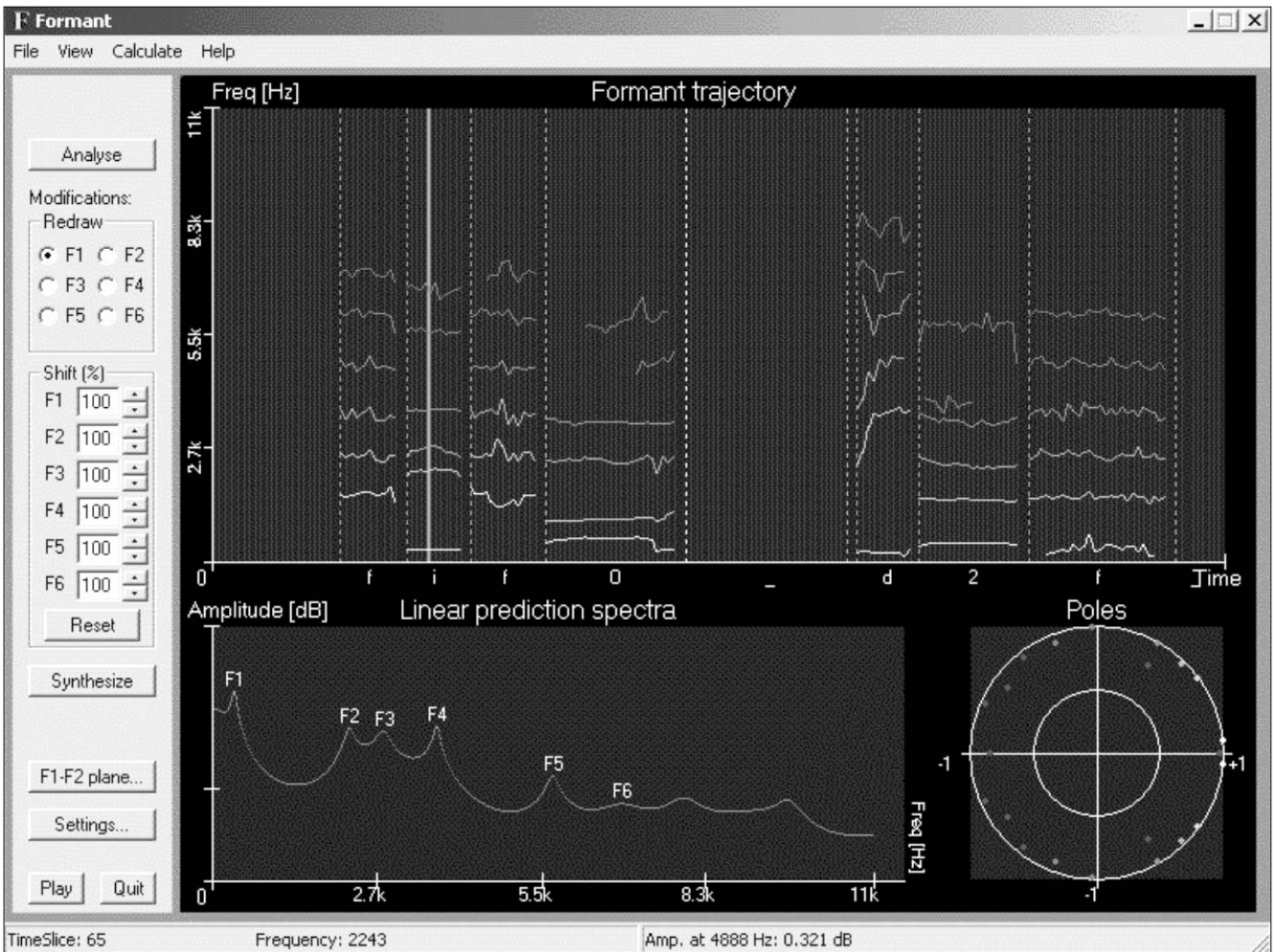


Figure 1. Analysis of a short recording by Kay Elemetrics CSL 4300B (top panel) and by the proposed algorithm – without (middle panel) and with (bottom panel) phonetic transcription

Figure 2. Graphical computer program for formant tracking and modification



ording was listened to by four native Hungarian subjects. All of the listeners were confident that they heard the meaningless word "fásü". This technique was effectively used in initial experiments to extend speech databases of concatenative synthesizers with vowels that were not recorded (e.g. vowels that do not exist in Hungarian).

It is generally accepted among speech researchers that the higher formant frequencies bear characteristics specific to the speaker. We conducted several initial experiments towards voice transformation where the aim is to modify the personal features so that the original speaker's identity disappears. By altering some carefully chosen formants, we could confuse the recognition of speaker identity in several listeners.

While manipulating formant tracks we noted that in case of drastic changes audible artifacts may appear in the speech signal. In case of minor (at most about 20%) changes the re-synthesized speech is of good quality and natural-sounding.

4.3. A graphical computer program for formant tracking and modification

A graphical computer program was developed for demonstration and evaluation (Figure 2). This program

is publicly available for educational and research purposes [15].

After analyzing the waveform, the program displays all the formant tracks (at most six) in different colors on the time-frequency plane. In order to check the results, the program is also capable of displaying the spectrogram or to show both plots overlaid. After selecting a time frame, the corresponding short-time linear prediction spectrum is drawn (with the formant peaks marked) and the poles of the linear prediction filter appear on the z-plane (in the bottom right corner of the window). The latter is a novel way of visualization that bears the same information as the linear prediction spectrum.

Formant tracks can also be displayed on the F_1 - F_2 plane (Figure 3). Instead of the commonly used scatterplots, the program draws the movement of the first two formants in time as a continuous curve.

The figure shows the F_1 - F_2 tracks for the vowels in the utterance. The horizontal position of the points of the curve corresponds to the value of the first formant while the vertical position refers to the second formant. The shade of the line represents the time: data from the first pitch period of the vowel is drawn with the darkest color and then it becomes lighter and lighter with each pitch period, the last pitch period being the light-

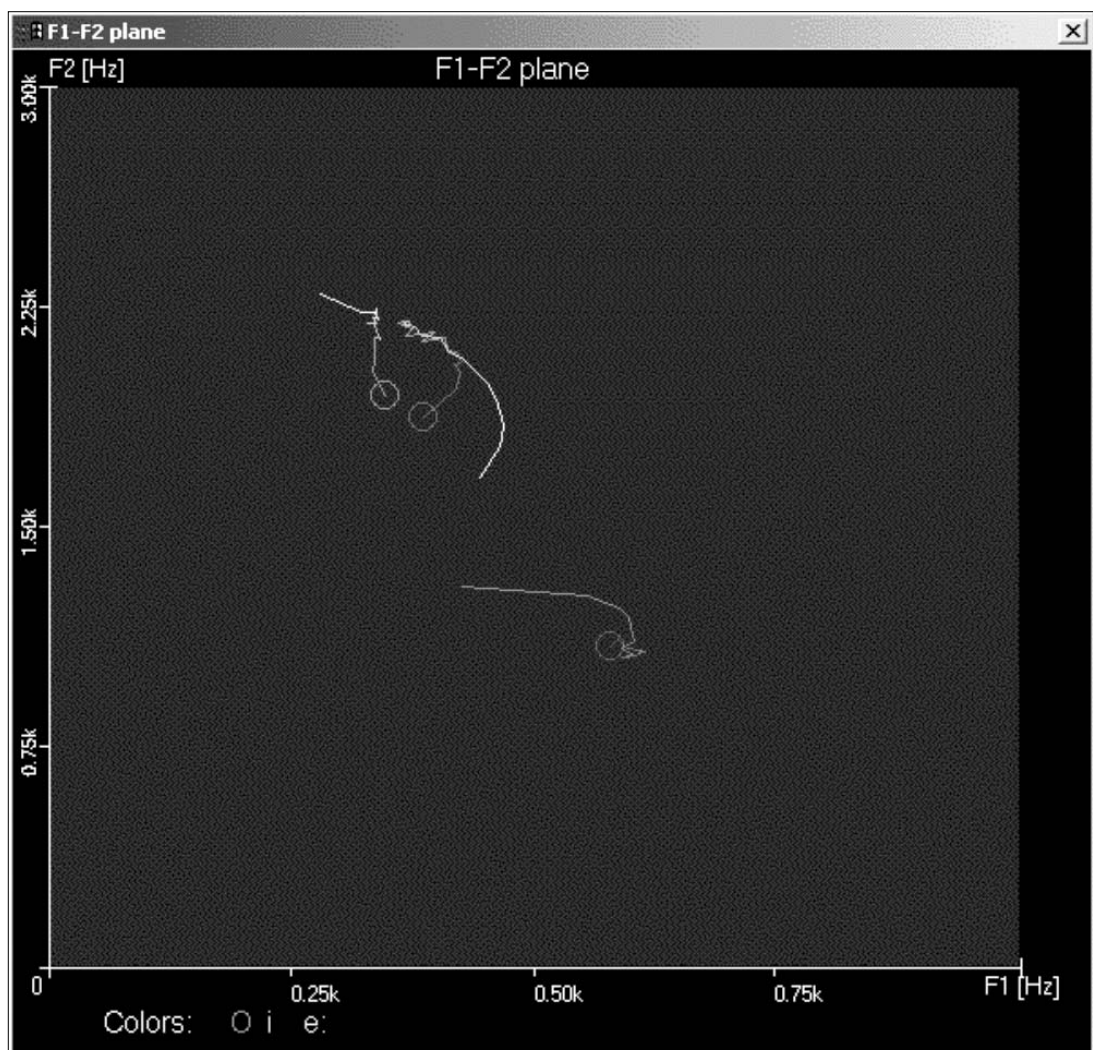


Figure 3.
Formant tracks of
vowels on
the F_1 - F_2 plane

est. In order to improve visibility, circles mark the starting points.

Formant tracks can be altered by redrawing them by hand or by using a factor for each trajectory. The waveform can be re-synthesized with the modified formant tracks and saved to a file.

One can imagine a wide range of applications for this program in the education of phonetics. For example, it can be used as a tool to demonstrate the formant structure of speech sounds or the distinctive features of vowels. It is also capable of visualizing the similarities and differences among different realizations of the same phoneme and to show the effects of co-articulation. It can also be used for phonetics research: for producing the stimulus set of perceptual experiments, for the study of dialects with “analysis by synthesis” and for producing plots of formant tracks.

5. Summary

A general purpose formant tracker and modifier is needed for a wide range of applications. In this paper such an algorithm was described and evaluated. Our method gives acceptable results even with scarce information and by using further input data (phonetic transcript), the results become very accurate. The algorithm was implemented and built into a graphical computer program.

This publicly available program can serve as a tool for education and research – besides in the courses taught by the authors, it is also used at the ELTE Faculty of Humanities. Furthermore, it was successfully applied to extensively examine the formant structure of Hungarian vowels (the data of this study is being analyzed).

References

- [1] Gordos, G., Takács, Gy.:
Digital Speech Processing
(Digitális beszédfeldolgozás),
Műszaki Könyvkiadó, Budapest 1983.
pp.52–59; 241–247.
- [2] Markel, J. D., Gray, A. H.:
Linear Prediction of Speech,
Springer-Verlag, Berlin 1976., pp.154–158.
- [3] Lobanov, B., Levkovskaya, T., Kheidorov, I.:
Speaker and channel – Normalized set of formant
parameters for telephone speech recognition,
Proc. of Eurospeech 1999, Vol. 1., pp.331–334.
- [4] Rabiner, L. R., Schafer, R. W.:
Digital Processing of Speech Signals,
Prentice-Hall, Englewood Cliffs, 1978.
- [5] Ouni, K., Lachiri, Z., Ellouze, N.:
Formant estimation using Gammachirp filterbank,
Proc. of Eurospeech 2001, Vol. 4., pp.2471–2474.
- [6] Weber, K., Bengio, S., Bourlard, H.:
HMM2 – Extraction of formant structures and
their use for robust ASR,
Proc. of Eurospeech 2001, Vol. 1, pp.607–610.
- [7] McCandless, S. S.:
An algorithm for automatic formant extraction using
linear prediction spectra, IEEE Trans. on Acoustics,
Speech and Signal Processing, Vol. 22, no.2., 1974.
- [8] Christensen, R. L., Strong, W. J., Palmer, E. P.:
A comparison of three methods of
extracting resonance information from predictor
coefficient coded speech, IEEE Trans. on Acoustics,
Speech and Signal Processing, Vol. 24, no.1., 1976.
- [9] Yegnanarayana, B.:
Formant extraction from linear prediction phase
spectra, Journal of the Acoustical Society of America,
1978, Vol. 63, pp.1638.
- [10] Reddy, N. S., Swamy, M. N. S.:
High-resolution formant extraction from linear
prediction phase spectra, IEEE Trans. on Acoustics,
Speech and Signal Processing, Vol. 32, no.6., 1984.
- [11] Slifka, J., Anderson, T. R.:
Speaker modification with LPC pole analysis,
Proc. of ICASSP 1995, pp.644–647.
- [12] Orchard, M. T.:
The Laguerre method for
finding the zeros of polynomials,
IEEE Transactions on Circuits and Systems, 1989.
Vol. 36, no.11, pp.1377–1381.
- [13] Olaszy, G.:
Electronic Speech Production
(Elektronikus beszédelőállítás),
Műszaki Könyvkiadó, Budapest, 1989.
- [14] Lee, M., van Santen, J., Möbius, B., Olive, J.:
Formant tracking using segmental
phonemic information, Proc. of Eurospeech 1999,
Vol. 6, pp.2789–2792.
- [15] <http://fonetika.nytud.hu>

Estimation of instantaneous parameters of speech signals with Teager-operator and Hilbert-Huang-transform

ISTVÁN PINTÉR

College of Kecskemét, GAMF Faculty, Department of Automation and Applied Informatics
pinter.istvan@gamf.kefo.hu

Keywords: Teager-operator, HHT, instantaneous amplitude and frequency, reconstruction from instantaneous parameters

In order to analyze the fine structure of speech signals, methods for determining nonlinear and non-stationary characteristics of speech are necessary. In this paper the Teager-operator and the Hilbert-Huang Transform (HHT) are presented as speech processing methods suitable for the estimation of instantaneous amplitude and instantaneous frequency. (In: 2006/8, pp.28–37.)

1. Introduction

In digital speech processing the so-called quasi-stationary signal model is often used in solutions to a lot of practical problems. It means that it is possible to process the speech signal with the sequence of overlapped speech segments in order to complete the computations necessary for the algorithm in question. It is presumed that the parameters of the applied speech model do not vary during the segment under processing. According to the relevant literature, the suitable segment duration is 2...5 times longer than the fundamental period, and the sufficient overlap is 1...3 times of pitch period [1].

During the progress of digital speech processing a demand has arisen for analyzing methods suitable for investigating speech signal changes of lower duration than the pitch period itself. These types of speech signal changes constitute the fine structure of speech. The phenomenon of the small fluctuation of the fundamental period during nonlinear vocal chord vibration – among many others – is an example which calls for methods necessary to represent the fine structure of speech. These methods ought to derive physically meaningful parameters from few speech samples only. Consequently, for these purposes the speech processing methods based on the quasi-stationary assumption are not appropriate [2].

Summing up succinctly the essence of the problem, it could be said that it is not possible to increase the time-resolution of the analysis, while keeping the detailed frequency-domain representation because of the uncertainty-relation of Dennis Gabor. Nowadays, the wavelet-transform is widely used in applications requiring increased time-resolution. However, the time-resolution of the wavelet-method is also limited by the time-scale uncertainty relation, which replaces the time-frequency uncertainty mentioned above [2,3].

A possible method, suitable for analyzing the fine structure of speech, is the Teager-operator-based Energy Separation (ES) algorithm. The wavelet-based analysis and the Teager-operator have recently led to suc-

cessful applications [4]. Another possibility in determining the instantaneous parameters is the application of the Hilbert-Huang-transform [5]. Because we have not found published results in the relevant literature available to us on the comparison of the Teager-operator-based methods and HHT-based ones, their comparison has been chosen as the subject of this article.

2. The Teager-operator and the ES-algorithm

2.1. The continuous-time Teager-operator and the estimation of instantaneous parameters

The definition of the Teager-operator has become possible after detailed investigation of the nonlinear physical phenomena of human speech production. In order to describe the fast changes in the speech-signal's energy during the fundamental period, it is useful to determine the overall energy of the system producing the speech. This overall energy can be estimated by applying a suitable operator to the speech signal – the operator is termed as Teager-operator [2]:

$$\Psi\{x(t)\} = \left(\frac{dx(t)}{dt}\right)^2 - x(t) \cdot \frac{d^2x(t)}{dt^2}, \quad (1)$$

where $\Psi\{\cdot\}$ denotes the Teager-operator. In the case of the signal $x(t)=a \cdot \cos(\omega \cdot t + \varphi)$ we get:

$$\frac{dx(t)}{dt} = -a \cdot \omega \cdot \sin(\omega \cdot t + \varphi), \quad (2)$$

$$\frac{d^2x(t)}{dt^2} = -a \cdot \omega^2 \cdot \cos(\omega \cdot t + \varphi),$$

which leads to $\Psi\{x(t)\} = a^2 \cdot \omega^2$ (3)

It can be checked that the result will be the same when the operator is applied to the signal $x(t)=a \cdot \sin(\omega \cdot t + \varphi)$, as it is expected. It is interesting to note that the next equation also holds:

$$\Psi\{a \cdot e^{j(\omega t + \varphi)}\} = 0. \quad (4)$$

A possible generalization of the signal $x(t)=a \cdot \cos(\omega \cdot t + \varphi)$ is the case when both the amplitude and the

phase are time-dependent, that is the form of the resulting AM-FM signal is the following:

$$x(t) = a(t) \cdot \cos(\varphi(t)). \quad (5)$$

By direct algebraic manipulation it is easy to check that in the case of arbitrary amplitude- and phase-function the application of the operator in (1) results in a formula which is not easy to manipulate further. However, in the case of slowly varying amplitude- and phase-functions by using the approximations below:

$$\frac{da(t)}{dt} \approx 0, \quad \frac{d\varphi(t)}{dt} \approx \text{const}, \quad \frac{d^2\varphi(t)}{dt^2} \approx 0, \quad (6)$$

and by applying the Teager-operator to the signal in (5) we get:

$$\begin{aligned} \frac{dx(t)}{dt} &\approx -a \cdot \frac{d\varphi(t)}{dt} \cdot \sin \varphi(t), \\ \frac{d^2x(t)}{dt^2} &\approx -a \cdot \left(\frac{d\varphi(t)}{dt}\right)^2 \cdot \cos \varphi(t), \\ \Psi\{x(t)\} &\approx a^2(t) \cdot \left(\frac{d\varphi(t)}{dt}\right)^2. \end{aligned} \quad (7)$$

The operator can also be applied to the derivative of the signal:

$$\Psi\left\{\frac{dx(t)}{dt}\right\} = \left(\frac{d^2x(t)}{dt^2}\right)^2 - \frac{dx(t)}{dt} \cdot \frac{d^3x(t)}{dt^3}. \quad (8)$$

By using the approximations in (6), after detailed calculations we finally get the relation below for the AM-FM signal in (5):

$$\Psi\left\{\frac{dx(t)}{dt}\right\} \approx a^2(t) \cdot \left(\frac{d\varphi(t)}{dt}\right)^4. \quad (9)$$

Therefore, both the value of the magnitude and the magnitude of the derivative of the phase (which is by definition the absolute value of the angular frequency) can be estimated by the formulae below:

$$\frac{\Psi\{x(t)\}}{\sqrt{\Psi\left\{\frac{dx(t)}{dt}\right\}}} = |a(t)|, \quad (10)$$

$$\sqrt{\frac{\Psi\left\{\frac{dx(t)}{dt}\right\}}{\Psi\{x(t)\}}} = \left|\frac{d\varphi(t)}{dt}\right|. \quad (11)$$

So, based on equations (1), (10) and (11) the slowly time-varying envelope and the slowly time-varying instantaneous angular frequency can be estimated from the signal itself. It is easy to check that for the signal $x(t) = a \cdot \cos(\omega \cdot t + \varphi)$ these estimations give exactly the same values of the (constant) amplitude and (constant) angular frequency.

2.2. Discrete-time Teager-operator and the ES-algorithm

After proper sampling and suitably approximating the derivation with differences equations (1), (10) and (11) can be considered as the basis of the computations. According to our numerical experiments the five-

point Savitzky-Golay smoothing derivative algorithm [6] gives acceptable results. This type of computation is further called as direct computation. The discrete-time version of the Teager-operator can also be derived from the continuous form in (1) by approximating the differentiation with the $d(n) = x(n) - x(n-1)$ difference. This leads to the next definition of the discrete-time Teager-operator:

$$\Psi_D\{x(n)\} = x^2(n) - x(n-1) \cdot x(n+1). \quad (12)$$

After some algebraic manipulation it calls forth that by applying the Teager-operator to the signal $x(n) = a \cdot \cos(\omega \cdot n + \varphi)$ discrete-time sequence the result is below:

$$\Psi_D\{x(n)\} = a^2 \cdot \sin^2 \omega, \quad (13)$$

where ω denotes digital angular frequency. In the case of the discrete-time Teager operator it can be shown that, starting from the $x(n) = a(n) \cdot \cos(\varphi(n))$ sequence the estimation formulae of the slowly varying instantaneous parameters are given below [2]:

$$a(n) \approx \frac{2 \cdot \Psi_D\{x(n)\}}{\sqrt{\Psi_D\{x(n+1) - x(n-1)\}}}. \quad (14)$$

$$\omega(n) \approx \arcsin\left(\sqrt{\frac{\Psi_D\{x(n+1) - x(n-1)\}}{4 \cdot \Psi_D\{x(n)\}}}\right). \quad (15)$$

The computation procedure defined by equations (12), (14) and (15) is called ES-algorithm in the literature. The benefit of the ES-algorithm is that it needs only three samples for the estimation, while the direct method above needs five samples, however, in the latest case the evaluation of the arcsin(.) function is not needed for determination of instantaneous digital angular frequency.

3. The Hilbert-Huang-transform and the computation of the instantaneous parameters

In the previous part it has been shown when several well-defined conditions are fulfilled, the computation of the instantaneous parameters is possible. These conditions can be guaranteed e.g. with a suitable band-pass filtering before estimation.

A natural question could arise: is there a much more general method for estimating the physically meaningful instantaneous parameters? The question was answered positively in 1988 in a paper by Norden E. Huang et al. [5]. The authors elaborated a signal decomposition algorithm which results in signal components having positive instantaneous frequencies, so the instantaneous parameters can be estimated using these components.

They proposed the so-called EMD-algorithm (Empirical Mode Decomposition), they termed the component signals as IMFs (Intrinsic Mode Function), and the instantaneous parameters of IMFs can be estimated by using the so-called canonical representation of analytic signals.

3.1. The empirical mode decomposition algorithm and the intrinsic mode functions

The intrinsic mode functions ought to have two basic properties [5]:

- The number of extrema and the number of the zero-crossings are the same or their difference equals 1,
- The mean value between the envelopes of local maxima and local minima is zero.

Details of the algorithm for determination of the intrinsic mode functions can be found in [5]. When determining an IMF only local speech sample values are used, that is the IMFs are computed with a locally adaptive manner. Moreover, the original signal can be reconstructed by summing up the IMFs, that is:

$$x(n) = r(n) + \sum_{k=0}^{K-1} m_k(n), \tag{16}$$

where $r(n)$ denotes the residual signal, $m_k(n)$ denotes the k -th IMF, and K denotes the number of IMFs. There is no estimation for the number of IMFs in [5], so it has to be determined experimentally.

3.2. The canonical representation of the signal and the instantaneous parameters

As it is well known from the work of Dennis Gabor [7], the $x(t)=a(t)\cdot\cos(\varphi(t))$ signal model is not unambiguous, but the so-called canonical representation – which can be derived from the complex analytic signal – is unambiguous. This latter signal is composed from the signal itself and also from its Hilbert-transform as given below:

$$\hat{x}(t) = H\{x(t)\} = \frac{1}{\pi} \cdot P \int_{-\infty}^{+\infty} \frac{x(\tau)}{t-\tau} d\tau \tag{17}$$

$$Z(t) = x(t) + j \cdot \hat{x}(t) = A(t) \cdot e^{j\Phi(t)} \tag{18}$$

and the canonical representation is defined as:

$$x(t) = A(t) \cdot \cos(\Phi(t)) \tag{19}$$

The instantaneous parameters in (19) are defined as:

$$A(t) = \sqrt{x^2(t) + \hat{x}^2(t)} \tag{20}$$

$$\omega(t) = \frac{d\Phi(t)}{dt} = \frac{d}{dt} \left(\arctan \left(\frac{\hat{x}(t)}{x(t)} \right) \right) \tag{21}$$

Although equation (21) defines the instantaneous angular frequency as a derivative of the phase of the analytic signal it can also be computed as the following partial derivative:

$$\omega(t) = \text{Im} \left\{ \frac{\partial}{\partial t} \ln(Z(t)) \right\} \tag{22}$$

which leads to the next relation:

$$\omega(t) = \frac{x(t) \cdot \frac{d\hat{x}(t)}{dt} - \frac{dx(t)}{dt} \cdot \hat{x}(t)}{x^2(t) + \hat{x}^2(t)}, \tag{23}$$

which can also be reached after the completion of the derivation in (21). Both (21) and (23) can be used for deriving an algorithm for the estimation of instanta-

neous angular frequency. From the point of view of realization we have to note there is an important relation between the Hilbert-transform and Fourier-transform of the signal, which is the following:

$$\hat{X}(j\omega) = -j \cdot \text{sgn}(\omega) \cdot X(j\omega), \tag{24}$$

moreover, the relation below also fulfils:

$$F\{x(t) + j \cdot \hat{x}(t)\} = \begin{cases} 2 \cdot X(j\omega) & \omega > 0 \\ 0 & 0 \leq \omega \end{cases}, \tag{25}$$

where $F\{\cdot\}$ denotes the Fourier-transformation.

3.3. The computation of the discrete-time Hilbert-transform and the estimation of the instantaneous parameters

The discrete-time Hilbert-transform of a signal can be computed either starting from (24) and by applying a suitable digital filtering operation [8] or by using an FFT-based algorithm (25). After determining the Hilbert-transform of the speech, the estimation of the instantaneous amplitude can be given as follows:

$$A(n) = \sqrt{x^2(n) + \hat{x}^2(n)} \tag{26}$$

For the computation of the instantaneous frequency two algorithms can be derived depending on the use of (21) or on the use of (23). By using (21) the phase-sequence can be given with the equation below:

$$\Phi(n) = \text{arctg} \left(\frac{\hat{x}(n)}{x(n)} \right), \tag{27}$$

During the evolution of the signal the phase-change can be given as:

$$\Phi_u(n) = \Phi(n) + r(n) \cdot 2\pi, \quad \Phi(n) \in [-\pi, \pi], \tag{28}$$

where $r(n)$ is a positive integer. After computing the instantaneous phase by applying a suitable phase-unwrapping algorithm, the instantaneous digital angular frequency can be approximated by the difference below:

$$\omega(n) = \Phi_u(n) - \Phi_u(n-1). \tag{29}$$

Another procedure can be derived using equation (23) and by approximating the derivation with a suitable manner. As in the previous part, the five-point Savitzky-Golay smoothing derivative algorithm can also be applied in this case.

4. Comparison of the instantaneous parameters computed with the Teager-operator and the HHT

4.1. The reconstruction of the signal from its instantaneous parameters

As it was presented in the second part of this paper, the absolute value of the amplitude and the frequency of the slowly varying signal can be estimated by using two algorithm-pairs. In the third part the basis of the estimation of the instantaneous amplitude was the analytic signal computed from the IMFs and the estimation of the instantaneous frequency was determined either di-

rectly or from the instantaneous phase sequence. For these estimations two algorithm-pairs have also been given. Because these algorithms have been derived using very different signal models, it is necessary to compare the similarity or dissimilarity of the estimations of the instantaneous parameters. There are four corresponding algorithm-pairs to be compared. In order to compare these algorithm-pairs, it is necessary to estimate the instantaneous parameters and from these to re-estimate the original speech signal by using the same reconstruction algorithm. In the case of the original signal $x(n)$ and the estimated signal $\tilde{x}(n)$, the performance of the algorithm-pairs can be characterized by the noise-to-signal ratio below:

$$NSR = 10 \cdot \log \left(\frac{\sum_{n=4}^{N-5} [x(n) - \tilde{x}(n)]^2}{\sum_{n=4}^{N-5} x^2(n)} \right) \quad (30)$$

Because only one algorithm estimates the phase directly and all the others estimate the instantaneous frequency, the basis sequence in reconstruction was in all cases the estimated instantaneous frequency and the determination of the phase sequence was the following:

$$\tilde{\Phi}(k) = \tilde{\Phi}(-1) + \sum_{n=0}^k \tilde{\omega}(n) \quad k = 0, 1, \dots, N-1 \quad (31)$$

According to our numerical experiments there is a phase-jitter between the original and the reconstructed signal, so the best initial phase value $\Phi(-1)$ has been determined by searching the best NSR using $\pi/180$ (1°) phase-step.

4.2. The comparison of the method using a test signal

For the test signal the following AM-FM signal, which can be found in the relevant literature, has been used [2]:

$$s(n) = (0,998)^n \cdot [1 + 0,8 \cdot \cos(2 \cdot \pi \cdot f_3 \cdot n)] \cdot \cos \left[2 \cdot \pi \cdot \left(f_1 \cdot n + \frac{1}{2 \cdot \pi} \cdot \sin(2 \cdot \pi \cdot f_2 \cdot n) \right) \right] \quad (32)$$

$$f_s = 10000\text{Hz} \quad f_1 = \frac{1000\text{Hz}}{f_s} \quad f_2 = \frac{100\text{Hz}}{f_s} \quad f_3 = \frac{50\text{Hz}}{f_s}$$

By examining the time-domain figure of this signal, it is clearly an IMF, so it is expected from the EMD-algorithm to give back only one IMF. This is also the case, as it can be seen in *Figure 1*. The difference between the reconstructed and the original signal can be characterized numerically, as it can be found in *Table 1*.

Table 1.
Characterization of the algorithm-pairs in the case of the test signal

Method	Original signal NSR (dB)	IMF1 NSR (dB)
Direct computation	-8	-8
ES-algorithm	-18	-19
HHT (phase-difference)	-24	-27
HHT (smoothing derivative)	-7	-7

Table 2.
Characterization of the algorithm-pairs in the case of the band-limited speech signal

Method	Original signal NSR (dB)	IMF1 NSR (dB)
Direct computation	-5	-5
ES-algorithm	-2	-2
HHT (phase-difference)	-29	-30
HHT (smoothing derivative)	-14	-14

4.3. The comparison of the methods in the case of band-limited speech

In the case of the speech signal it is necessary to ensure the slow changes of the parameters to be estimated, which can be solved by suitable band-pass filtering. Although – to our best knowledge – there is no accepted method for designing the suitable filter, it can be deduced from the relevant literature that a member of some 1 CB filter bank is suitable for the application of the Teager-operator [4] which is the basis for the above mentioned two methods. In this part the estimation of the instantaneous parameters of band-limited speech signal is illustrated.

The speech samples stem from the utterance of the Hungarian word ‘igen’, uttered by a native male speaker, using 8 kHz sampling frequency and 16 bit linear quantization. The original utterance was band-limited to 300 Hz...3400 Hz by using a linear-phase FIR filter. After examining the spectrogram, because of the presence of a strong formant near 500 Hz, a member of a perceptual wavelet filter-bank has been used for further FIR-filtering [9]. By examining the band-limited signal, it can also be clearly seen that it is an IMF too, so it is expected from the EMD-algorithm to give back only one IMF. This is also the case, as it can be seen in *Figure 2*. The difference between the reconstructed and the original signal can be characterized numerically, as it can be found in *Table 2*. The best result has been given by the HHT (phase-difference) method.

In *Figure 2* and in the case of the Teager-operator-based methods the deep valley at 177 ms is caused by the realized program, because it gives back 0 value for the instantaneous frequency in order to avoid the square root from the negative value (see equations (10),(11), (14) and (15)).

4.4. The comparison of the methods in the case of speech signals

The results presented in the previous part show that the Teager-operator-based estimations are very similar to those computed by the HHT-based methods. In the following, our results in analyzing one uttered word are presented. However, there was no band-pass filtering in the investigations below. The EMD process itself serves as an adaptive band-pass filter-bank. The adaptive nature of the algorithm stems from the iterative computing of the upper and lower envelopes. That is, in the case of the first IMF these envelopes are fitted to the

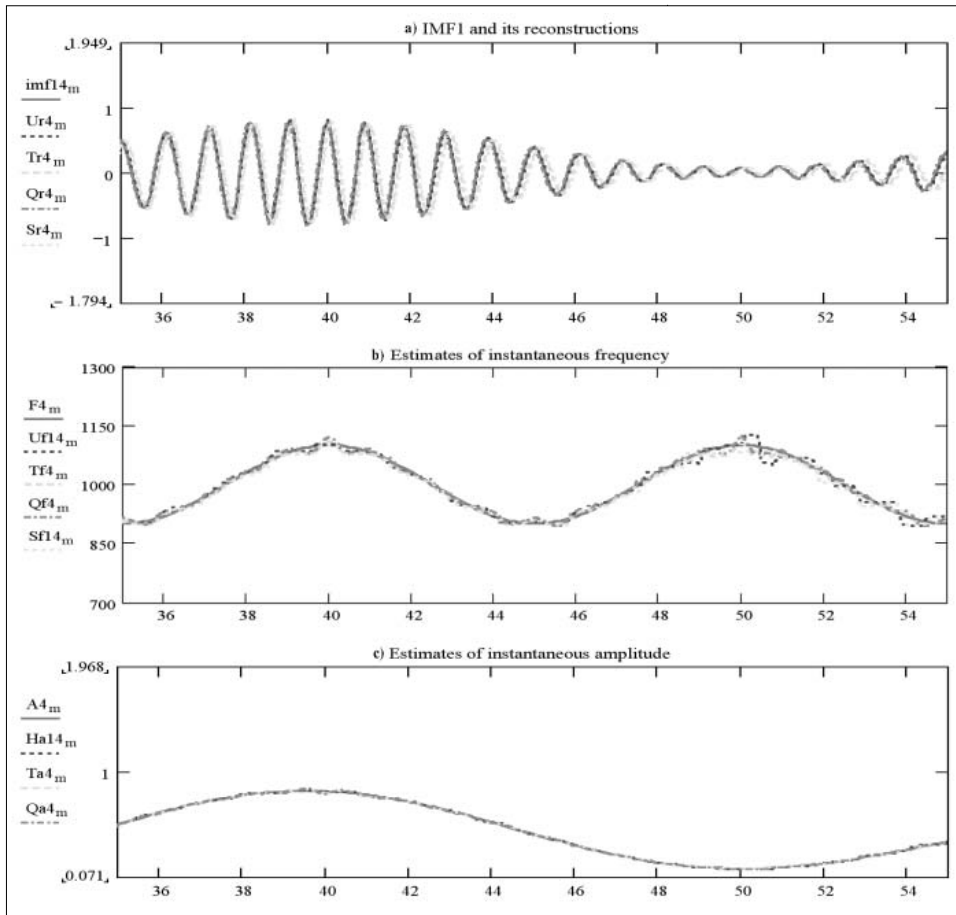


Figure 1.

Results of the application of each algorithm-pair on the test signal.

- a) the IMF1 and its estimations
- b) the theoretical instantaneous frequency and its estimations
- c) theoretical instantaneous amplitude and its estimations

rapid changes in the signal structure, which means the extraction of the higher frequency signal component. After subtracting the first IMF from the original signal, the procedure above is repeated in the lower frequency parts of the signal several times. It is not obvious however, whether this type of filtering is enough for the application of the Teager-operator or not.

This question has also been examined with the utterance of the Hungarian word analyzed in the previous part. It has been mentioned in the third part that there is no basis for the number of the IMFs. However, according to our numerical experiments, by using the first three IMFs the original speech can be reconstructed with NSR of -22 dB, so the instantaneous parameters have been estimated in the case of the first three IMF using the four methods mentioned in the previous part of the paper. The reconstruction itself has been accomplished for these three IMFs and the reconstructed speech has been computed by summation of the reconstructed IMFs.

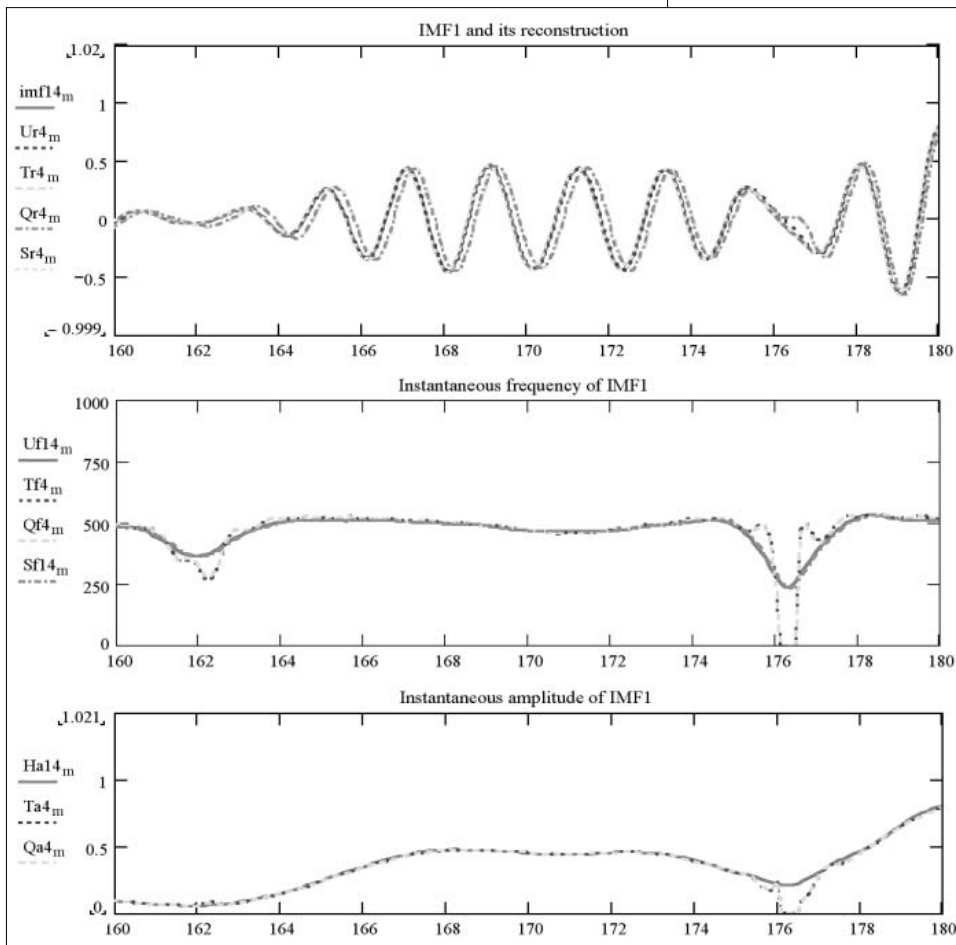


Figure 2.

- a) the IMF1 and its estimations
- b) the estimations of the instantaneous frequency
- c) the estimations of the instantaneous amplitude

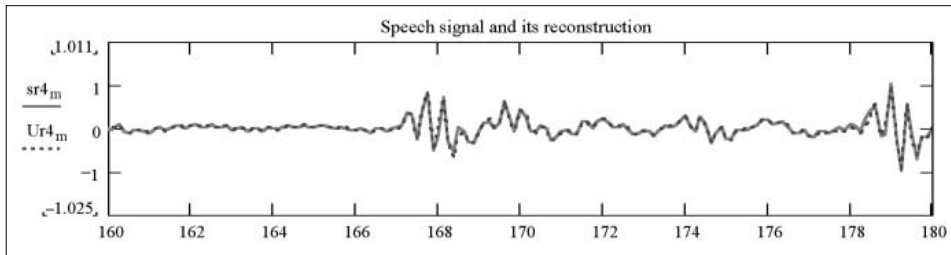


Figure 3.
Reconstruction of the speech signal from the first three reconstructed IMFs.

Method	Original signal NSR (dB)	IMF1 NSR (dB)	IMF2 NSR (dB)	IMF3 NSR (dB)
HHT (phase-difference)	-12	-10	-19	-24

Table 3.
Data in the case of best reconstruction

For the ease of survey Figure 3 presents the parts of the speech computed in the best case, and in the Table 3 the numerical values can be seen.

5. Conclusions

In this paper four methods have been proposed for the estimation of the instantaneous amplitude and instantaneous frequency of the speech signal. Two of these methods are based on the Teager-operator, and the others are based on the HHT. The methods have been illustrated with figures computed using a test signal and a speech signal, moreover, a reconstruction method has also been proposed in order to re-compute the speech from the instantaneous parameters. The reconstruction method was also the basis for the comparison of the methods mentioned above.

Our most important result is, that the Teager-operator based methods and the HHT-based methods give similar estimates for the instantaneous parameters of the IMFs of speech. It is planned to continue the work to discover the application areas of the algorithms presented in this paper.

Acknowledgement

The author would like to thank for their kind help and encouragement in his speech processing algorithm development work to Géza Gordos, Géza Németh and Péter Tatai (BME VIK TMIT).

References

- [1] Gordos G., Takács Gy.: Digitális beszédfeldolgozás (in Hungarian). Műszaki Könyvkiadó, 1983.
- [2] Quatieri, T.F.: Discrete-time Speech Signal Processing: Principles and Practice. Prentice-Hall, 2001.
- [3] Abbate, A., DeCusatis, M.C., Das, K.P.: Wavelets and Subbands: Fundamentals and Applications. Birkhäuser, 2002.
- [4] Chen, S-H., Wang, J-F.: Speech Enhancement Using Perceptual Wavelet Packet Decomposition and Teager Energy Operator. Journal of VLSI Signal Processing 36, pp.125–139., Kluwer Academic Publishers, 2004.
- [5] Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N-C., Tung, C.C., Liu, H.H.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proc. R. Soc. Lond. A (1998) 454, pp.903–995.
- [6] Valkó P. Vajda S.: Műszaki-tudományos feladatok megoldása személyi számítógéppel (in Hungarian). Műszaki Könyvkiadó, 1987.
- [7] Gábor, D.: Theory of communication. Journal Inst. Electr. Eng. Vol. 93., pp.429–457., 1946.
- [8] Símonyi E.: Digitális szűrők – a digitális jelfeldolgozás alapjai. Műszaki Könyvkiadó, 1984. (in Hungarian)
- [9] Pintér, I.: Perceptual wavelet-representation of speech signals and its application to speech enhancement. Computer, Speech and Language, Vol. 10. No.1, pp.1–22., Academic Press, 1996.

TITAN, TTCN-3 test execution environment

JÁNOS ZOLTÁN SZABÓ, TIBOR CSÖNDES

*Test Competence Center, Ericsson Hungary Ltd.
{janos.zoltan.szabo, tibor.csondes}@ericsson.com*

Keywords: *testing, TTCN-3, test system implementation*

This paper presents the TTCN-3 test execution environment of Ericsson called TITAN. We show the internal details and the operation of the toolset. Unique TITAN features and differences from other commercial TTCN-3 tools are discussed. As a result of our development TTCN-3 and TITAN became a widely used test solution within Ericsson and we contributed to the TTCN-3 standardization work within ETSI. (In: 2006/9, pp.29–33.)

1. Introduction

During the 80's and 90's the 2nd version of TTCN (Tree and Tabular Combined Notation) was used for testing telecommunication systems based on the OSI Basic Reference Model [1]. The TTCN-2 language had two properties that prevented it from wide acceptance: its difficult tabular format and the limited application area. Therefore, at the end of 90's, ETSI started the redesign of the language and standardization of TTCN version 3.

In the planning phase it was considered that besides the recent conformance testing methods the new language should be applicable for new types of tests, like: interoperability testing, performance testing, robustness testing and system testing.

ETSI gave up the close relationship to OSI BRM and made it possible to test internet based protocols and Application Programming Interfaces (API).

1.1. TTCN-3 Language

The first version of TTCN-3 (Testing and Test Control Notation) was published as a set of ETSI standards in 2000. Since then several minor enhancements and corrections have been made on the language. The latest, third edition of the standard documents were issued in 2005 [2]. The creators of TTCN-3 tried to get rid of the clumsy structures and thus prevent the bad reputation of the TTCN language, which was widespread among telecom experts since TTCN-2. This was worth the efforts since the result has been a language that is easy to understand and helps testing considerably. The textual representation and basic control statements of TTCN-3 is quite similar to programming languages like C/C++ so many potential users can understand the basic language constructions without deeper TTCN-3 knowledge.

A detailed TTCN-3 introduction can be found in [3].

1.2. History of TITAN

The development of TITAN was started as an M.Sc. thesis work in the beginning of 2000. The main goal of

the project was to create an efficient, protocol and application independent test environment, which is also capable of running performance tests. TTCN-3, which was still under development in ETSI at that time, was a perfect choice as the input language of the new test tool.

The first prototype of the system was ready in less than one year. This version supported only a subset of TTCN-3 language features, but its architecture was very similar to the current state [4]. Since the first release TITAN has been under continuous development: it follows the changes of the language specification and has more and more convenience functions. TITAN supports almost all constructs of TTCN-3 and numerous non-standard language extensions.

The main milestones of the development were the following:

- 2000: first prototype
- 2001: parallel and distributed test execution
- 2002–2003: support of ASN.1 [5]
and built-in encoders/decoders
- 2004: graphical user interface
- 2005: full TTCN-3 and ASN.1 semantic analysis

2. Structure of TITAN

TITAN uses C++ as intermediate language for realization of the TTCN-3 test system. The block diagram of TITAN test execution environment can be seen in *Figure 1* (next page).

TITAN consists of the following parts:

2.1. TTCN-3 and ASN.1 Compiler

The TTCN-3 and ASN.1 compiler is the largest and most complex part of TITAN. Its tasks include the parsing and analysis of the test suite and reporting the syntax and semantic errors that are present in the input. If all modules of the test suite are found to be correct the compiler generates C++ program modules that will be parts of the Executable Test Suite (ETS).

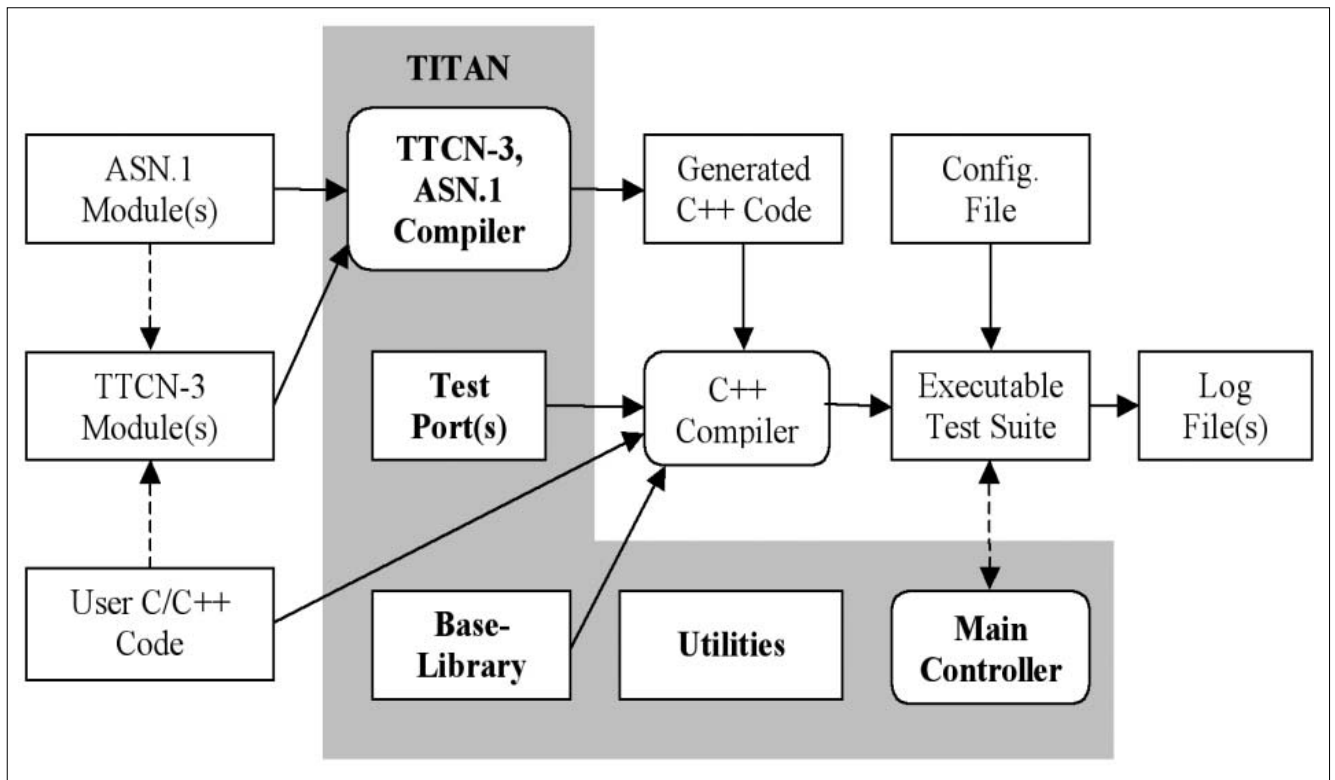


Figure 1. TITAN block diagram

2.2. Base Library

The base library contains those common and static parts of the ETS that are independent of the actual test suite. It consists of manually written C++ code, which is compiled into binary form in advance. The base library comprises the C++ classes representing the basic TTCN-3 data types and the functions implementing the built-in operations of the language (such as timer, port, test component and verdict handling). Other auxiliary functions that are necessary to run the ETS (e.g. the logging and configuration file processing routines) are also parts of the base library.

2.3. Test Ports

TTCN-3 models all external communication between the test system and the outside world using abstract messages sent and received through communication ports. The purpose of test ports is to bridge the gap between the Executable Test Suite and the System Under Test (SUT). In fact, the test ports are the C++ realizations of TTCN-3 communication ports in the ETS.

TITAN provides a well-defined programming interface, the so-called Test Port API for handling incoming and outgoing messages. Typical test port implementations communicate with the SUT using IP-based protocols accessed through the socket API of the operating system. The responsibility of test ports includes the encoding and decoding of structured, abstract TTCN-3 and ASN.1 messages, that is, to transform them to and from the transfer coding (i.e. binary octet streams used on the underlying communication channels).

2.4. Utility Programs

TITAN contains several small utility programs that make test suite development, compilation, test execution and result analysis easier. There are scripts for automated test suite launching, tools for test log post-processing (utilities for merging, formatting and filtering log files) and so on.

2.5. Main Controller

When a test suite requires more than one TTCN-3 test components to be run in parallel, their operation must be coordinated. This task is done by a dedicated application called Main Controller (MC), which belongs to TITAN and is independent of the actual test suite. The MC has direct connections for supervising all other components of the TTCN-3 test system. It contributes to the creation and termination of test components as well as the establishment and break-down of port connections, among others. When running performance tests the test system can be distributed over several networked computers to generate high traffic load against the SUT. In this case the load balancing between the participating tester computers is also done by the MC. To avoid bottlenecks in the test system the MC only has central coordinating tasks; it does not take part in elementary test operations. Further details about TITAN's distributed test architecture can be read in [6].

Besides, the MC has the user interface for interactive test execution, which can be either command line or graphical. The current state of the test system can be continuously monitored and the user has the possibility for intervention in test run by stopping the current test case or starting a new one.

3. The Operation of TITAN

3.1. Syntax and Semantic Analysis

The lexical analysis and syntax checking of input modules are done by parsers generated with conventional tools GNU *flex* and *bison*. During parsing the compiler builds up special memory structures called abstract syntax tree (AST). The integrated TTCN-3 and ASN.1 compiler has the advantage that the different front-ends for the two languages transform the definitions into the unified structures of one common AST. This allows the direct usage of data types and values of protocols with ASN.1 descriptions from TTCN-3 test suites.

The purpose of semantic analysis is to detect errors like invalid references, forbidden operations or type clashes and perform some transformations on the AST for code generation (e.g. calculating and reducing constant arithmetic expressions). TITAN compiler has a one-pass semantic analyzer, which means the algorithm walks through the AST only once. However, the order in which the AST nodes are visited is influenced by the references between the definitions. The most challenging task of semantic analyzer development was to properly identify and handle the ambiguous language constructs of TTCN-3 and ASN.1 that cannot be classified during the syntax check (e.g. the *start* operation of TTCN-3 can refer to a port, timer or test component with the same syntax).

When the compiler detects a syntax or semantic error it does not stop after printing the first error message, but keeps on analyzing to discover more errors in the test suite. Special error recovery techniques are employed in order to prohibit error messages whenever references point to existing, but erroneous definitions. Otherwise one simple fault could launch an avalanche of error messages.

3.2. Code Generation

C++ code generation is based on the AST, on which the semantic checker has made some simplifications. The generated code of TITAN uses static typing, which means that every TTCN-3 and ASN.1 data type is mapped to distinct C++ classes. The main benefits of the statically typed run-time environment are the high execution speed and modest memory usage, since the C++ compiler can arrange the optimal memory structures for TTCN-3 data values and their fields. A further advantage of static typing is that the type correctness of TTCN-3 operations is checked by the C++ compiler as well. This is an extra verification step on the entire ETS without executing it.

The latter property of the generated code was exploited in earlier versions of TITAN where the compiler lacked semantic analyzer. Instead of building AST the equivalent pieces of C++ code were created immediately while parsing the TTCN-3 input. The internal interfaces of the run-time environment were designed in such way that TTCN-3 semantic errors were mapped to simi-

lar kinds of faults in the output, which were caught and reported by the C++ compiler.

The significant drawback of static typing is the large size of the generated C++ class hierarchy. The output generated from the data types of complex protocols has long compilation time. The large code of data types is compensated by mapping other TTCN-3 definitions, such as values, data templates and behavior descriptions (functions, test cases, etc.) to very compact C++ code.

In case of dynamic typing, which is used by most of commercial TTCN-3 tools on the market, all TTCN-3 data values are constructed from the same generic structure that can carry the values of any type. Because of this every run-time operation must examine whether the given arguments have the correct types. This applies to, for example, the built-in elementary operation of integer addition, which has to first ensure that the generic structures of arguments contain numbers rather than strings or something else. The extra tasks of dynamic typing can result in 10 or 100 times slower execution speed compared to TITAN.

3.3. Executable Test Suite Derivation

The entire compilation process including the translation of the test suite to C++, the compilation of generated code and test ports to binary form and the final linking of the ETS is managed by UNIX *make*. A special file called *Makefile*, which is interpreted by *make*, describes the different steps of compilation and the dependencies between them. TITAN can create a *Makefile* based on the list of TTCN-3 and ASN.1 modules and test ports needed by the test suite.

Typical TTCN-3 test suites contain a lot of modules. It can be observed that the majority of modules change very seldom during the test development process. For example, the modules that define the message types of protocols will never change in normal cases. The changes between two compilations and test runs are usually limited to a few modules and a few lines of TTCN-3 code within them (mostly the behavior statements that describe the test cases). Since a full re-compilation can take several minutes or hours in case of complex test suites this should be avoided after minor changes.

TITAN compiler together with the *make* utility supports incremental compilation. This means that the results of previous compilation are reused as much as possible and only the updated modules are translated to C++ and subsequently to binary code. Identification of the modules that require re-compilation is not an easy task in some cases.

If the definitions of a module are imported by another than any change of the imported definitions will affect the importing definitions, which can be imported into a third module, and so on. So a single change in one module can cause several modules to be re-compiled in order to maintain the consistency of the ETS.

Despite the above difficulties practical examples show that the incremental build system of TITAN can significantly decrease the compilation times and improve the efficiency of TTCN-3 test suite development.

3.4. Encoding and Decoding

During test run the abstract messages to be sent to or received from SUT must be encoded or decoded. To make this task easier TITAN contains several built-in codecs, which are accessible through a special C++ API. Using the built-in codec the encoding or decoding of a message can be done in a few lines of C++ code regardless the complexity of the data type. The encoding and decoding functionality can be placed in test ports or external functions, which are written entirely in C++, but can be invoked from TTCN-3.

ASN.1 data types can be encoded according to the standardized Basic Encoding Rules (BER). TITAN supports two different encoding schemes for TTCN-3 data types: a table-based bit-oriented (RAW) and a text-based (TEXT) one. The exact coding rules of TTCN-3 types, which can be quite complex in case of some protocols,

are specified using the attributes of the respective type definitions.

3.5. Graphical User Interface

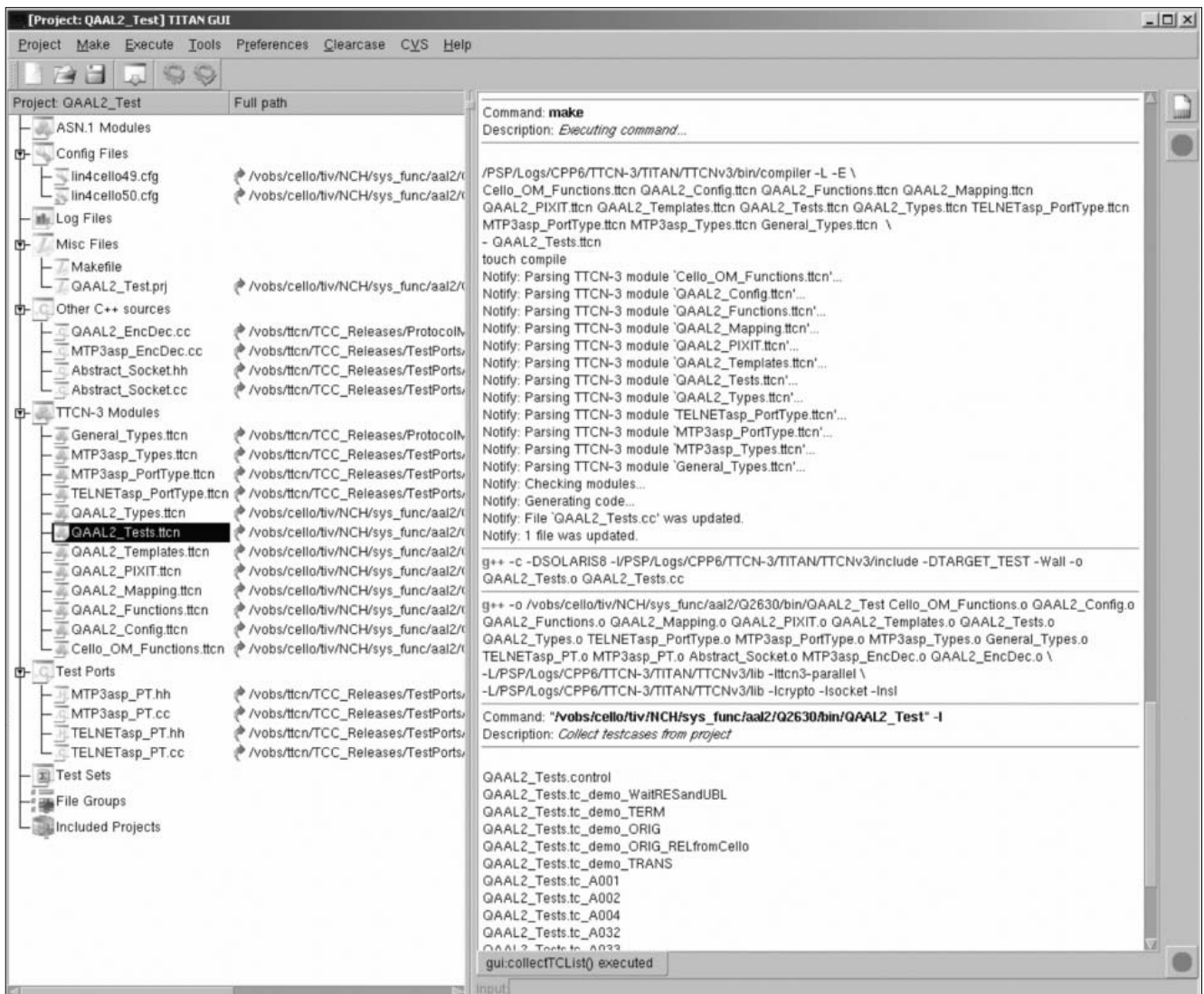
TITAN contains a Graphical User Interface (GUI) built together with the Main Controller, which provides a user-friendly environment both for test development and test execution.

Figure 2 shows a screenshot of the main GUI window. The left part shows the overview of the current project (lists of TTCN-3 modules, test ports, configuration files and other source files) while the right window lets the user follow the compilation process. The example presents the results of an incremental compilation.

4. TTCN-3 Interfaces

A TTCN-3 Executable Test Suite can communicate with the outside world via interfaces. ETSI has defined 6 interfaces in 2 standards (TTCN-3 Runtime Interface, TRI

Figure 2. Graphical User Interface



[7] and TTCN-3 Control Interface, TCI [8]). The two TRI interfaces (SUT Adapter and Platform Adapter) describe the connections between the ETS and the system under test as well as the operating system (e.g. timer handling). TCI contains four interfaces: Test Management (test case execution, test suite parameterization), Component Handling, Coding/Decoding and Logging.

These standards describe the interfaces with programming language independent notation and give canonical C, Java and XML mappings for the data types and procedures. The aim of the interface standardization was to allow users to switch from one TTCN-3 runtime environment to a tool of another vendor without changing the application specific software modules.

At the moment TITAN does not support any of the standard interfaces above. There are several reasons. On one hand, in 2002-2003, when the interface standards were published by ETSI, TITAN was already a mature and complete TTCN-3 test system with its full featured proprietary interfaces. On the other hand, during the development we preferred other technical aspects than ETSI.

Our goal was to provide an effective test system in such a way that the users need to develop the smallest and simplest external program modules possible. Therefore TITAN provides only one programming interface towards SUT, which is the test port interface. TITAN supports the functionalities of other parts of TRI and TCI as efficient built-in modules without public programming interfaces.

TITAN test port interface has more advantages than the SUT Adapter interface of TRI. A test port instance always handles single TTCN-3 communication port and consequently one protocol. Therefore the distribution of messages of different protocols is solved by the interface itself. In contrast, with TRI all the messages toward SUT are processed by the same function of the single adapter.

The separation of different protocols has to be implemented in the adapter by the user. Whenever a new protocol or system interface is introduced in the test system, a new test port, like a building block, can be simply added with TITAN. However, with TRI it will be necessary to redesign the entire adapter. In addition, test ports are more suitable for distributed performance testing since the test ports connect the TTCN-3 parallel test components directly to the SUT eliminating traffic bottlenecks.

Later implementation of the standardized interfaces in TITAN could cause difficulties, because they assume dynamic typing and multi threaded operation. Generally the interfaces of TRI and TCI were not designed for efficient operation so the practical advantages are questionable in TITAN. We think it would not be possible to achieve better performance, simpler structure or more comfortable usage compared to the existing built in functionalities of TITAN.

5. Summary

Thanks to the development and usage of TITAN, Ericsson has joined the TTCN-3 standardization work within ETSI. Ericsson has submitted 196 out of the total 340 Change Requests (CRs), which represents our activity in the field of TTCN-3 standardization. Most of our CRs resolve ambiguous structures and conflicts in the core language; but several extensions have also been proposed, which made the language simpler and more usable.

TITAN is the official TTCN-3 Test Tool within Ericsson since 2003. Since then a department with almost 40 people, the Test Competence Center is working on the development, deployment and support of TITAN and TITAN-based test solutions. Our product portfolio includes Test Ports, TTCN-3 Protocol Modules and complete Test Suites. Test Competence Center has customers from Ericsson units all over the world. Although TITAN has not been sold outside Ericsson the number of its users has been continuously growing during the last years. Almost 50 Test Ports and 100 Protocol modules have been developed for TITAN, which gives a great opportunity for testing a wide spectrum of telecommunication systems.

References

- [1] ITU-T, X.200, Information Technology – Open Systems Interconnection – Basic Ref. Model: The Basic Model, 1994.
- [2] ETSI ES 201 873-1, v3.1.1 (06/2005) The Testing and Test Control Notation, version 3. Part1: Core Language
- [3] Jens Grabowski, Dieter Hogrefe, György Réthy, Ina Schieferdecker, Anthony Wiles, Colin Willcock, “An introduction to the testing and test control notation (TTCN-3)”, *Computer Networks*, Vol. 42, Issue 3, pp.375–403., Elsevier North-Holland, Inc. 2003.
- [4] János Zoltán Szabó, “Experiences of TTCN-3 Test Executor Development”, *Testing of Communicating Systems XIV., Application to Internet Technologies and Services*, Edited by I. Schieferdecker, H. König and A. Wolisz, Kluwer Academic Publishers, 2002.
- [5] ITU-T X.680 (07-2002) Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.
- [6] János Zoltán Szabó, “Performance Testing Architecture for Communication Protocols”, *Periodica Polytechnica, Electrical Engineering, Budapest University of Technology and Economics*, 2003. 47/1-2.
- [7] ETSI ES 201 873-5, v3.1.1 (06/2005) The Testing and Test Control Notation, version 3. Part5: TTCN-3 Runtime Interface (TRI)
- [8] ETSI ES 201 873-6, v3.1.1 (06/2005) The Testing and Test Control Notation, version 3. Part6: TTCN-3 Control Interface (TCI)

A taxonomy of routing protocols for wireless sensor networks

GERGELY ÁCS, LEVENTE BUTTYÁN

*Budapest University of Technology and Economics, Department of Telecommunications,
Laboratory of Cryptography and Systems Security (CrySyS)
{acs, buttyan}@crysys.hu*

Keywords: *sensor networks, routing protocols, network and operational models*

Wireless sensor networks are large scale networks consisting of a large number of tiny sensor nodes and a few base stations, which communicate using multi-hop wireless communications. The design of energy efficient routing protocols for such networks is a challenging task, which has been in the focus of the sensor network research community in the recent past. This effort resulted in a huge number of sensor network routing protocols. The proposed protocols show a high variety, which stems from the diverse requirements of the various envisioned application scenarios. In this work, we propose a taxonomy of sensor network routing protocols, and classify the mainstream protocols proposed in the literature using this taxonomy. We distinguish five families of protocols based on the way the next hop is selected on the route of a message, and briefly describe the operation of a representative member from each group. (In: 2006/12, pp.3–11.)

1. Introduction

Sensor networks are composed of resource constrained sensor nodes and more resourced base stations. All nodes in a network communicate with each other via wireless links, where the communication cost is much higher than the computational cost. Moreover, the energy needed to transmit a message is about twice as great as the energy needed to receive the same message. Consequently, the route of each message destined to the base station is really crucial in terms network lifetime: e.g., using short routes to the base station that contains nodes with depleted batteries may yield decreased network lifetime. On the other hand, using a long route composed of many sensor nodes can significantly increase the network delay.

Unfortunately, some requirements for the routing protocols are conflicting. Always selecting the shortest route towards the base station causes the intermediate nodes to deplete faster, which results in a decreased network lifetime (if we measure the network lifetime by the time that lasts until the first node dies in the entire network). At the same time, always choosing the shortest path may result the lowest energy consumption and lowest network delay globally. Ultimately, the routing objectives are tailored by the application; e.g., real-time applications require minimal network delay, while applications performing statistical computations may require maximized network lifetime. Hence, different routing mechanisms have been proposed for different applications [1]. These routing mechanisms primarily differ in terms of routing objectives and routing techniques, where the techniques are mainly influenced by the network characteristics.

In this paper, we propose a taxonomy of sensor network routing protocols, and classify the mainstream protocols proposed in the literature using this taxonomy.

We distinguish five families of protocols based on the way the next hop is selected on the route of a message, and briefly describe the operation of a representative member from each group.

2. Taxonomy of routing protocols

In order to select the most suitable routing mechanism for a sensor application, we have to classify all routing protocols according to a well-defined taxonomy. Using this classification, all protocols become comparable for an application designer. As a part of this taxonomy, we define a system model that describes the network and operational characteristics of the routing protocols, and an objective model that describes the routing objectives of the protocols. Furthermore, the system model encompasses the definition of the network model as well as the operational model of routing protocols.

2.1. Network model

This model describes the characteristics of a network that can be divided into two groups: the characteristics of base stations, and the characteristics of sensor nodes.

2.1.1. Base station

It is commonly agreed that the base station is a powerful device with unconstrained energy supply and computational capacity. However, the following characteristics of a base station may severely influence the operation of a routing protocol:

Number:

The number of the base stations can be one or more than one. In most practical applications, the increased number of base stations provides more robust data ga-

thering, and may also decrease the network delay. However, the typical number of the base stations is one. If only one base station is presented (and there is no need for explicit communication between sensor nodes), the destination node for all messages is the same, while in case of multiple base stations, the destination node can differ for some messages.

Mobility:

During the routing process, the base station can be fixed (stationary) or mobile. In some applications, where the number of base stations is too small to ensure acceptable network delay and robustness, the base station supports mobility during data gathering. This property of the base station severely affects the routing protocol, since all nodes in the network field cannot be continuously aware of the current position of the base station, and the routing mechanism needs to find the mobile base station in the field. Moreover, the routing topology created by a routing protocol may heavily vary in time that causes extra overhead in the network layer. Some routing protocols cannot be employed with mobile base stations, while others tolerate limited mobility.

Presence:

The base station can be either continuously or partially presented during the routing process. In the latter case, the routing protocol must support the temporary lack of a base station (e.g., the base station is switched off for a certain amount of time due to maintenance reasons), since a missing base station cannot definitely mean a failure. Thus, the messages should not be dropped or rerouted rather their delivery should be delayed.

Coverage:

Many routing protocols assume that base station can cover the whole network field by its power range. In these networks, the base station can reach every other node, if there are no obstacles in the field. Therefore, there is no need for routing between the sensor nodes and the base station in such cases. However, we note that it is not a reasonable assumption in most practical applications; a more realistic assumption would be that the base station can communicate with only some nodes in its close vicinity.

2.1.2. Sensor nodes

In most sensor networks, sensor nodes are homogeneous tiny devices with constrained energy supply and computational capabilities. In addition, we assume that all sensor nodes are stationary. The following characteristics of sensor nodes may differ for some networks. Hence, they can influence the protocol operation.

Deployment:

Sensor nodes can be deployed in either a deterministic or a random fashion. When the nodes are deployed along a road-side, or in a metro-station, the deployment is rather deterministic than random. In these cases, the routing protocol should adapt to the fixed

network topology. However, numerous routing protocols proposed so far assume that the deployment is random (e.g., the nodes are dropped out from a helicopter).

Transmission power:

The transmission power can be either dynamically adjustable or fixed. In the latter case, each sensor node transmits each message using the same energy level. In the former case, every node can calculate what energy level should be used to transmit a message to a neighboring node. This energy level may be inversely proportional to the cost assigned to the neighboring node.

Coverage:

It is commonly assumed that a sensor node cannot reach all nodes in the network field. A routing protocol can require large transmission power per node in order to get a fully connected network. However, it can only be beneficial in small-sized networks due to the large energy consumption and interference range.

Addressing:

The task of routing in sensor networks is to deliver the queries coming from the base station to the sensor nodes which have the requested data (in case of query-driven routing protocols, see later), and to return the requested data to the base station. Accordingly, we can distinguish the addressing method of queries and responses:

- Query-addressing: All routing protocols which use query dissemination in the networks employ data-based (What is the average temperature?), or location-based addressing (What is the average temperature in location (x,y) ?).
- Response-addressing: The response is either returned on the reversed path which the query traversed, or it is routed back purely based on location information. In the former case, neighboring nodes use locally (or globally) unique identifiers to identify the neighbor from which they received the query, and which is further used to forward the reply towards the destination.

MAC interface:

The data-link layer can be responsible for neighbor discovery (where the neighbor definition is protocol-dependent). In addition, it also need to perform the calculation of cost values (where the cost definition is also protocol-dependent). Some routing protocols are integrated with the data-link layer in order to achieve better performance in terms network delay and energy consumption (cross layer design). However, the data-link layer is generally not responsible for these tasks. Thus, the routing protocol itself must calculate its own neighbor list and the costs of neighbors.

2.2. Operational model

The operational model describes the main orthogonal operational characteristics of a routing protocol.

Communication pattern:

A routing protocol can support the communication from sensor nodes to sensor nodes, from base stations to sensor nodes, as well as from sensor nodes to base stations.

- **Node-to-Node:** This communication pattern is not typical for sensor networks, only those protocols support that inherently which were primarily proposed for ad hoc networks but can also be used in sensor networks. Generally, there is no need for this kind of communication in sensor networks.
- **Node-to-Base station:** This pattern need to be supported in order to route responses back to the base station. This communication pattern is typically reverse-multicast (many-to-one), a.k.a. convergecast, which means that every sensor node is able to send (directly or indirectly) a message to any base station. If there are multiple base stations or only one node is responsible for gathering and transmitting the sensed data to the base station, this pattern can also be unicast.
- **Base station-to-Node:** This pattern needs to be supported in order to route requests originated from the base station to sensor nodes. This communication pattern is typically anycast (one-to-many), which means that any sensor node which has the requested data can respond to the query. If some nodes are uniquely identified in the network (by their ids, locations, etc.), then multicast (one-to-many) and unicast (one-to-one) patterns can also be supported. All these patterns mean that the base station(s) are able to send (directly or indirectly) a message to any sensor nodes.

Hierarchy:

Employing hierarchical routing protocols, a hierarchy level is assigned to each node, and a node only forwards those messages that are originated from a lower-level node. Optionally, a node aggregates incoming data and forwards this aggregated data to upper-layer nodes. The base station can be found on the top of the hierarchy. The hierarchy construction can be dynamic or static. Using dynamic construction, the role of the aggregator is rotated, and all nodes that are selected an aggregator will forward all data to their aggregator.

The aim of forming this hierarchy is to prolong the network lifetime. Using non-hierarchical protocols, each sensor node can accept all messages coming from any other sensor nodes for further aggregation and forwarding. Thus, any sensor node can behave as an aggregator node in non-hierarchical architectures.

Delivery method:

In most routing protocols, a node selects only a single path towards the base station, and the only instance

of a message (single/single) is forwarded along this single path. However, a node may also select multiple paths, and the node forwards either the single instance of a message on a deterministically or randomly chosen single path (multiple/single) or one copy of a message per path (multiple/multiple).

Computation:

Each sensor node selects the next-hop towards the base station either by itself in a decentralized manner, or every node sends its neighbor list to the base station and the base station computes the next-hop for all nodes in the network in a centralized manner. Although the centralized computation gives optimal solution, it may yield heavy network communication, which is only tolerable in small-sized networks with fixed network topology. Using decentralized or centralized computations, all nodes only store the identification of their neighbors, where the neighbor definition is protocol-dependent. In the simplest case, a node *A* considers another node *B* as a neighbor, if *A* receives a routing message sent by *B*. In other cases, nodes discover their neighbors by broadcasting simple *HELLO* messages on a certain energy level. A node *A* considers another node *B* as a neighbor, if *A* receives a *HELLO* message sent by *B*.

Next-hop:

A common characteristic of all protocols is that each node selects its next-hop (for the query and/or the response) towards the destination based on locally stored information, which may include the routing costs, next-hop identifiers, etc. The next-hop can be selected by

- randomly among all neighbors (probabilistic)
- inferring routing information from the sensed data that is carried by the message (content-based)
- using the stored routing control information (control-based)
- using a hierarchical-based scheme (hierarchical)
- using geographic positions (location-based)
- broadcasting the message and the neighbors decide whether to re-broadcast the message (broadcast-based)

If both queries and responses are routed in a location-based or broadcast-based manner, a node is typically required to store only negligible amount of routing information like the positions of neighbors or its own routing cost. Routing protocols belonging to this group are *stateless* protocols. On the other hand, if the queries or responses are routed in a probabilistic, hierarchical, content-based or control-based manner, a node may need more extensive processing or storage resources. These routing protocols are also referred as *stateful* protocols.

Reporting model:

The reporting model describes *what* initiates the data reporting process. In this sense, we distinguish time-driven, query-driven, and event-driven protocols.

* Here, we assume that the sensors may be responsible for different sensing tasks.

Time-driven:

Employing a time-driven routing protocol, a sensor node is triggered in specific moments, when it should perform its measurement task and forwards the measurement to its next-hop neighbor. These activations can be periodic or one-shot in time. Short periods may cause more traffic in the network, and the quality of routing in terms of energy efficiency becomes a crucial concern. Time-driven sensors may be pre-programmed, or the reporting schedule may come with explicit queries. Furthermore, a time-driven routing protocol can support the reporting of

- complex (the reported data has several atomic components, e.g., temperature and humidity) or simple (atomic) data (e.g., only temperature is reported)* ,
- aggregated or non-aggregated data,
- replicated (more than one sensor can provide the requested information) or unique data (only one sensor can provide the requested information).

Query-driven:

In most sensor applications, the base station disseminates its query in the network, while the sensor nodes try to resolve this query, and they may send a response back to the base station. Hence, the task of a query-driven protocol is to route the queries to the measurement area, and to route back the response to this query. Furthermore, a query-driven routing protocol can support the reporting of

- complex or simple (atomic),
- aggregated or non-aggregated,
- replicated or unique data.

Event-driven:

A sensor node sends a measurement towards the base station only if a given event occurs (e.g., the temperature falls below a certain threshold). Furthermore, an event-driven routing protocol can support the reporting of

- complex or simple (atomic),
- aggregated or non-aggregated,
- replicated or unique data.

Most routing protocols belong to multiple reporting models.

2.3. Routing objectives

Some sensor applications only require the successful delivery of messages between a source and a destination. However, there are applications that need even more assurance. These are the real-time requirements of the message delivery, and in parallel, the maximization of network lifetime.

Non-real time delivery:

The assurance of message delivery is indispensable for all routing protocols. It means that the protocol should always find the route between the communicating nodes, if it really exists. This correctness property can be proven in a formal way, while the average-case

performance can be evaluated by measuring the message delivery ratio.

Real-time delivery:

Some applications require that a message must be delivered within a specified time, otherwise the message becomes useless or its information content is decreasing after the time bound. Therefore, the main objective of these protocols is to completely control the network delay. The average-case performance of these protocols can be evaluated by measuring the message delivery ratio with time constraints.

Network lifetime:

This protocol objective is crucial for those networks, where the application must run on sensor nodes as long as possible. The protocols aiming this concern try to balance the energy consumption equally among nodes considering their residual energy levels. However, the metric used to determine the network lifetime is also application dependent. Most protocols assume that every node is equally important and they use the time until the first node dies as a metric, or the average energy consumption of the nodes as another metric. If nodes are not equally important, then the time until the last or high-priority nodes die can be a reasonable metric.

3. Protocols

Table 1 overviews the network model and routing objectives of the most significant routing protocols, while *Table 2* describes the operational model of the same protocols. We merged some protocols which have identical system model and routing objectives into a single row (we put a star after their common name). The content of each cell is explained in Section 2, where a single star in a cell means an arbitrary value (i.e., the protocol supports all values of the cell). In the followings, we distinguish routing protocol families based on how a protocol selects a next-hop on the route of the forwarded message. We also briefly describe the operation of a representative protocol from each family.

3.1. Content-based routing protocols

These protocols determine the next-hop on the route purely based on the query content. This type of routing protocols fits the most to the architecture of sensor networks, since the base station do not query specific nodes rather it requests only for data regardless of its origin. From this family of protocols and paradigms, we briefly describe Directed Diffusion [2], which is considered to be the basis of other protocols like GBR, and Energy Aware Routing.

Using Directed Diffusion, the base station initially floods the network with an interest, which contains attribute-value pairs describing the requested data. Upon reception of an interests, each sensor node sets a gradient towards the sender node. If a node receives the same interest from different neighbors, then the node

can set multiple gradients, which correspond to the same interest, pointing to different neighbors. The neighbors are differentiated by locally unique identifiers. The sources forward the data along their gradients that is followed by the intermediate nodes up to the base station along the route. If there are more gradients at a node for the same interest, then the node forwards one copy of the message for each neighbor along each gradient. A gradient defines the requested data at each sensor node in conjunction with the next-hop towards the base station for which a message, that contains the requested data, should be forwarded. Each gradient is weighted proportionally to the amount of data that is allowed to traverse the gradient.

After a while, the base station selects the route with the best quality and increases the weight of the gradients along the route (positive re-inforcement), whereas it decreases on the others (negative re-inforcement). Intermediate nodes may aggregate the received data, and forward this aggregated data along the corresponding gradients with a rate that is proportional to the weight of the gradient. The base station periodically re-sends the interests along the used routes in order to keep alive the gradients of intermediate nodes. In this way, the base station keeps the empirically best routes and eliminates the routes that have worse quality. Optionally, all nodes can use cache techniques in order to achieve shorter response time and increase robustness.

The paradigm fits well for tracking applications, and it only requires the usage of some local addressing method to distinguish the one-hop neighbors of a node. One main drawback of Directed Diffusion is that it consumes significant network resources until the selection of the empirically best route.

3.2. Probabilistic routing protocols

In order to aid load-balancing and increased robustness, the next-hop on the route can be selected in a random fashion among all neighbors. These protocols assume that all sensor nodes are homogeneous and randomly deployed. We overview the operation of Energy Aware Routing [5] protocol as follows.

The main objective of Energy Aware Routing is to prolong the network lifetime by aiding load-balancing. This on-demand protocol is destination initiated, which means that the destination initiates the construction of the routing topology. Using this routing protocol, sensor nodes randomly select the next-hop neighbor for each message to be forwarded. The probability of selecting a certain neighbor is inversely proportional to its cost. This cost of a neighbor depends on the residual energy of the node, and the energy needed to transmit a message to this node.

The neighbor list and their corresponding cost values are provided by the MAC protocol. The protocol saves 21.5% more energy and prolongs the network

Table 1. Network and objective model

Protocol	Network model										Objectives
	Base station				Sensor nodes						
	Num.	Mobility	Presence	Coverage	Deploy.	T. power	Coverage	Addressing		MAC	
								Query	Response		
Rumor Routing	One	Fixed	*	Partially	Random	*	Partially	Data	ID	*	Non real-time
MCFA	One	Fixed	*	Partially	*	*	Partially	X	X	*	Non real-time
Energy Aware Routing	More	Fixed	*	Partially	*	Adjustable	Partially	Data	ID	Needed	Non real-time, lifetime
Directed Diffusion	More	Limited	Continuous	Partially	*	*	Partially	Data	ID	*	Non real-time
GBR	More	Limited	Continuous	Partially	*	*	Partially	Data	ID	*	Non real-time, lifetime
LEACH	One	Fixed	Continuous	Full	Random	*	Partially	Data	ID	*	Non real-time, lifetime
TEEN	One	Fixed	Continuous	Partially	Random	*	Partially	Data	ID	*	Real-time, lifetime
APTEEN	One	Fixed	Continuous	Partially	Random	*	Partially	Data	ID	*	Real-time, lifetime
PEGASIS	One	Fixed	Continuous	Partially	Random	*	Full	Data	ID, Location	*	Non real-time, lifetime
ACQUIRE	More	Limited	Continuous	Partially	*	*	Partially	Data	ID	*	Non real-time
IDSQ/CADR	More	Fixed	*	Partially	*	*	Partially	Data	ID, Location	*	Non real-time, lifetime
Geographic Routing (guaranteed delivery)	More	Mobile	*	Partially	*	*	Partially	Location	Location	*	Non real-time
GEAR	More	Limited	*	Partially	*	*	Partially	Location	Location	*	Non real-time, lifetime
MECN	One	Fixed	*	Partially	*	Adjustable	Partially	X	Location	*	Non real-time, lifetime
TTDD	More	Mobile	*	Partially	*	*	Partially	Location	Location	*	Non real-time, lifetime
SAR (DAM)	More	Limited	*	Partially	*	*	Partially	X	ID	*	Non real-time, lifetime
HPAR	One	Fixed	*	Partially	*	Adjustable	Partially	*	*	*	Non real-time, lifetime
SPEED	More	Fixed	Continuous	Partially	*	*	Partially	Location	Location	Needed	Real-time, lifetime

lifetime by 44% compared to Directed Diffusion (if we measure the network lifetime by the time until the first node runs out of its energy supply).

The routing protocol consists of three phases:

- 1) Setup-phase
- 2) Data communication phase
- 3) Route maintenance

Setup-phase

Initially, the destination disseminates a request message in the network using a controlled flooding technique. By this message, each node determines all routes with their costs towards the destination.

1. The destination floods the network in the direction of the source node with a request message containing a cost value initially set to 0: $Cost=0$

2. Every intermediate node forwards the requests for those neighbors, which are closer to the source, but farther from the destination than the sender of the request. Formally, node N_i sends a request to N_j , where N_j is a neighbor of N_i , only if the following equations hold: $d(N_i, N_S) \geq d(N_j, N_S)$, $d(N_i, N_D) \leq d(N_j, N_D)$, where $d(N_i, N_j)$ denotes the distance of N_i and N_j , and N_S, N_D , are the identifiers of the source and the destination, resp.

3. Upon the reception of the request, N_j calculates the cost of the route from N_j to the destination in the following way: $C_{N_j, N_i} = Cost + Metric(N_j, N_i)$, where $Metric(N_j, N_i)$ denotes the metric between nodes N_j and N_i (see later).

4. The requests with too high costs are silently dropped by N_j . Only the requests with low costs are considered, and the corresponding neighbor is added to the routing table of N_j : $FT_j = \{i | C_{N_j, N_i} \leq \alpha(\min_k C_{N_j, N_k})\}$

5. N_j assigns a probability to each neighbor in its table FT_j :

$$P_{N_j, N_i} = \frac{1/C_{N_j, N_i}}{\sum_{k \in FT_j} 1/C_{N_j, N_k}}$$

6. N_j calculates the average cost of all routes, that are represented in FT_j , towards the destination:

$$Cost(N_j) = \sum_{i \in FT_j} P_{N_j, N_i} \cdot C_{N_j, N_i}$$

7. The cost value of the request to be re-broadcast is set to this average cost: $Cost = Cost(N_j)$, and the request is re-broadcast according to Step 2.

The metric used in Step 3 is calculated as follows: $C_{i,j} = e_{i,j} \alpha R_i^\beta$, where $C_{i,j}$ denotes the cost metric between node N_i and node N_j , $e_{i,j}$ denotes the energy consumed by transmitting a message from N_i to N_j , and R_i is the residual energy level of N_i normalized to the initial energy level. α and β are tunable parameters.

Data communication phase

1. The source sends the message to one of its neighbors, where the neighbor is randomly selected with a probability that is equal to the probability assigned to the corresponding neighbor in the routing table of the source.

Table 2. Classification of routing protocols with respect to operational model

Protocol	Operational model										
	Next-hop		Communication pattern			Hierarchy	Delivery	Computation	Reporting model		
	Query	Response	N2N	N2BS	BS2N				Time-driven	Query-driven	Event-driven
Rumor Routing	Random	Random	X	Rev. M.	Anycast	No	Single / Single	Decent.	X	Non-agg., Simple	X
MCFA	X	Broadcast	X	Rev. M.	X	No	Single / Single	Decent.	*	X	*
Energy Aware Routing	Broadcast	Random	X	Rev. M.	Anycast	No	Multiple / Single	Decent.	*	*	X
Directed Diffusion	Broadcast	Content	X	Rev. M.	Anycast	No	Multiple / Multiple	Decent.	*	*	X
GBR	Broadcast	Content	X	Rev. M.	Anycast	No	Multiple / Multiple	Decent.	*	*	X
LEACH	*	Hierarchical	X	Rev. M.	Anycast	Yes	Single / Single	*	*	*	X
TEEN	*	Hierarchical	X	Rev. M.	Anycast	Yes	Single / Single	*	X	*	*
APTEEN	Hierarchical	Hierarchical	X	Rev. M.	Anycast	Yes	Single / Single	*	*	*	*
PEGASIS	*	Hierarchical	X	Unicast	Anycast	Yes	Single / Single	*	*	*	X
ACQUIRE	Random	Controlled	X	Rev. M.	Anycast	No	Single / Single	Decent.	X	Non-agg.	X
IDSQ/CADR	Content	Controlled, Location	X	Rev. M.	Anycast	*	Single / Single	Decent.	X	*	X
Geographic Routing (guaranteed delivery)	Location	Location	Unicast	Unicast	Unicast	No	Single / Single	Decent.	Non-agg., Unique, Simple	Non-agg., Unique, Simple	Non-agg., Unique, Simple
GEAR	Location	Location	Unicast	Rev. M., Unicast	Anycast	No	Single / Single	Decent.	Non-agg.	Non-agg.	Non-agg.
MECN	X	Location	X	Rev. M.	Anycast	No	Single / Single	Decent.	*	X	*
TTDD	Hierarchical	Hierarchical	X	Rev. M.	Anycast	Yes	Single / Single	Decent.	Non-agg., Unique	Non-agg., Unique	X
SAR (DAM)	X	Hierarchical	X	Rev. M.	Anycast	Yes	Single / Single	Decent.	X	X	Simple
HPAR	Hierarchical	Hierarchical	X	Rev. M.	*	Yes	Single / Single	Cent.	*	*	*
SPEED	Location	Location	Uni-, Any-, Multicast	Uni-, Anycast, Rev. M.	Uni-, Any-, Multicast	No	Single / Single	Decent.	Non-agg.	Non-agg.	Non-agg.

2. Every intermediate node selects a next-hop for the message in the same way as the source. Namely, it selects a neighbor from its routing table with a probability that is equal to the probability assigned to the neighbor, and forwards the message to this selected neighbor.

3. This process repeats until the message reaches the destination (base station).

Route maintenance phase

The destination (base station) infrequently updates the routing table of all nodes by flooding the network with new requests.

The drawbacks of the protocol are its complex addressing method, and the increased communication overhead during the setup phase compared to Directed Diffusion.

3.3. Location-based routing protocols

These protocols select the next-hop towards the destination based on the known position of the neighbors and the destination. The position of the destination may denote the centroid of a region or the exact position of a specific node. Location-based routing protocols can avoid the communication overhead caused by flooding, but the calculation of the positions of neighbors may result extra overhead. The local minimum problem is common for all decentralized location-based routing protocols: it might happen that all neighbors of an intermediate node are farther from the destination than the node itself. In order to circumvent this problem, every protocol uses different routing techniques. Here, we describe the operation of GEAR (Geographical and Energy Aware Routing) [6], which is a sensor-specific location-based routing protocol.

Employing GEAR, a sensor node sends the request to only one neighbor towards the destination. Thus, the initial flood of the query is avoided and the protocol saves more energy than Directed Diffusion. The response can be routed in the same way as the query is routed, or some different mechanism may be used for response routing similar to Directed Diffusion. Each node maintains an estimated and a learned cost value for each destination. The learned cost value is used to circumvent holes in the network, and it is considered as a refinement of the estimated cost value. If there are no holes along a route (every intermediate node has a neighbor that is closer to the destination than the node itself), then the learned cost value equals to the estimated cost value for the destination at a node.

The protocol consists of two phases:

- 1) Forwarding the messages towards the target region
- 2) Disseminating the message within the target region

Forwarding the messages towards the target region

An intermediate node N selects the next-hop from those neighbors that are closer to the destination than

N . The next-hop neighbor must have the minimal learned cost value among the closer neighbors. If such neighbor does not exist, then N selects the neighbor which has the minimal learned cost value among all neighbors. Initially, the learned cost value equals to the estimated cost value for all nodes, where the latter one can be computed using the following formula:

$$c(N_i, R) = \alpha d(N_i, R) + (1 - \alpha)e(N_i),$$

where $d(N_i, R)$ is the distance between neighbor N_i and the centroid of the target region, $e(N_i)$ is the normalized residual energy of N_i , and α is a tunable parameter. Every intermediate node calculates its own estimated cost, and broadcasts its cost value. Thus, every node will be aware of the estimated cost of its neighbors. Initially, the learned cost value for destination R is $h(N_i, R) = c(N_i, R)$.

After N sends the message to its selected neighbor N_{min} (which has the minimal learned cost value for R), N updates its own learned cost:

$$h(N, R) = h(N_{min}, R) + C(N, N_{min}),$$

where $C(N, N_{min})$ denotes the cost of transmitting a message from N to N_{min} in terms of energy, distance, normalized residual energy, or a combination of these metrics. Therefore, if the path from N to R is composed of n nodes, the learned cost value of the path converges to the real cost of the path within n steps. Additionally, all nodes broadcast their own learned costs infrequently. Hence, nodes can circumvent any holes in the network using learned cost values with appropriate update techniques.

Apart from prolonging the network lifetime, GEAR successfully delivers even 80% more messages than other location-based routing protocols like GPSR.

Disseminating the message within the target region

When a message reaches the target region, the nodes inside this region employ either controlled or recursive flooding technique in order to disseminate the message inside the region. Controlled flooding is suggested to be used if nodes are not densely deployed. In high-density networks, recursive geographic flooding is more energy efficient than restricted flooding. In that case, the region is divided into four subregions and four copies of the message are created. This splitting and forwarding process continues until the regions with only one node are left.

3.4. Hierarchical routing protocols

In case of hierarchical protocols, all nodes forward a message for a node (also called aggregator) that is in a higher hierarchy level than the sender. Each node aggregates the incoming data by which they reduce the communication overload and conserve more energy. Therefore, these protocols increase the network lifetime and they are also well-scalable.

The set of nodes which forward to the same aggregator is called cluster, while the aggregator is also referred as clusterhead. Clusterheads are more resourced nodes, where resource is generally means that their residual energy level is higher than the average. The reason is that they are traversed by high traffic and they perform more computation (aggregation) than other nodes in the cluster. Hierarchical routing is mainly two-layer routing where one layer is used to select clusterheads and the other layer is used for routing. In the followings, we overview the operation of LEACH (Low Energy Adaptive Clustering Hierarchy) protocol [3], which has been served as a basis for several other routing protocols like TEEN, APTEEN, etc.

In LEACH, nodes dynamically form a cluster in a distributed manner. Clusterheads are elected randomly, and this role is dynamically rotated in order to aid load-balancing. Each clusterhead aggregates all data coming from its cluster, and the aggregated data is forwarded directly to the base station. Therefore, LEACH assumes that all nodes in the network are able to reach the base station directly. LEACH uses a TDMA/CDMA MAC to reduce inter-cluster and intra-cluster collisions.

However, data collection is centralized and is performed periodically. Therefore, this protocol is most appropriate when there is a need for constant monitoring by the sensor network. Simulations showed that only 5% of nodes need to act as clusterheads in order to minimize energy consumption. The operation of LEACH consists of two phases; setup phase and steady state phase. These phases periodically repeats after each other, a consecutive setup and steady state phase is also called a round during the protocol run.

Setup phase

In the setup phase, clusters are created and clusterheads are selected. A given fraction of nodes, denoted by p , declare themselves as clusterheads at the beginning of each round in the following way (from simulation results, p typically equals to 0.05). A sensor node n chooses a random number between 0 and 1. If this number is greater than a threshold denoted by $T(n)$, then node n declares itself as a clusterhead, where

$$T(n) = \frac{p}{1 - p \cdot (r \bmod 1 / p)}$$

if $n \in G$. In the formula of $T(n)$, r denotes the round number, and G is the set of nodes which have not been selected as a clusterhead in the last $1/p$ rounds. Afterwards, the newly selected clusterheads advertise their clusterhead status in a message broadcast in a certain energy level.

Every node except the clusterheads decide on the cluster to which they want to belong to. This decision is based on the signal strength of the advertisement. The non-clusterhead nodes inform their selected clusterheads that they will be a member of the cluster. Afterwards, a clusterhead creates a TDMA schedule and assigns a time slot to each node when it can transmit. This schedule is broadcast to all members in the cluster.

Steady state phase

In the steady state phase, the cluster-members forward all measured data to their clusterheads. After receiving all data, a clusterhead forwards the aggregated data to the base station. At the end of the phase (after a certain time determined a priori), the network is switched to setup phase again in order to create new clusters. The duration of the steady state phase is longer than the duration of the setup phase in order to minimize overhead.

A disadvantage of the protocol is that it is not applicable to networks deployed in large areas, since all nodes must be able to reach the base station directly. Another drawback is that the protocol assumes that all cluster-members continuously report data to their clusterhead. Furthermore, it might happen that the elected clusterheads will be concentrated in one part of the network. Hence, some nodes will not have any clusterheads in their vicinity. Moreover, the protocol assumes that all nodes have the same initial energy level. All despite, the main problem might be the extra overhead caused by dynamic clustering.

3.5. Broadcast-based routing protocols

The operation of these protocols is very straightforward. Each node in the network decides individually whether to forward a message or not. If a node decides to forward, it simply re-broadcasts the message. If it declines to forward, the message will be dropped. To the best of our knowledge, the only representative of this protocol family is called MCFA (Minimal Cost Forwarding Algorithm) [4].

The main advantage of MCFA is that nodes do not store any information about their neighbors, only their own cost. The protocol consists of two phases. In the first phase, each node calculates its own cost which is eventually the cost of the minimal costed route from the node to the base station. In order to perform this cost calculation, the base station floods the whole network with a request message with a cost field C initialized to 0. Initially, all node costs are set to infinity.

A node N_i , which receives the request message from node N_j , delays the re-broadcasting with a time proportional to $\alpha \cdot C_{N_i, N_j}$ (in order to choose the correct α value for the link between N_i and N_j the protocol should take into account the link delay, and link reliability), where C_{N_i, N_j} denotes the cost of the link between N_i and N_j (energy consumption, delay, etc.). Afterwards, N_i updates the cost field in the request message: $C = C + C_{N_i, N_j}$, sets its own node cost to C , and finally re-broadcasts the updated request message. After re-broadcasting, N_i declines to re-broadcast any further request messages. In [4], the authors proved that in ideal cases (when α is sufficiently large) each node re-broadcasts only one request message, which contains the minimal cost of the node.

In the second phase, all nodes are able to forward any messages towards the base station in the follow-

ing way. The source node N places its own cost C_N into the message to be sent, and finally broadcasts the message. A node M receiving this message checks whether $C_N - C_{N,M} = C_M$. If it holds, then M is on the minimal costed route between the base station and node N , so it re-broadcasts the message after placing $C_{N,M}$ into the message. Otherwise, M drops the message. Every subsequent node can check whether it lies on the minimal costed route or not based on the cost values inferred from the message.

A drawback of the protocol is that *every* node receiving a message must perform extra computation in order to determine whether it is on the minimal route or not.

4. Summary

In this work, we classified routing protocols proposed for wireless sensor networks. In Section 2, we presented a novel taxonomy of sensor routing protocols. As a part of this taxonomy, we defined a network model, an operational model, and various routing objectives.

In Table 1 and 2, we listed the most significant sensor routing protocols according to this taxonomy, where *Table 1* contains the classification of routing protocols based on the network model and routing objectives, while *Table 2* describes the operational characteristics of the same protocols. In Section 4, we divided all protocols into five groups according to the applied next-hop selection mechanism. Finally, we briefly described the operation of a representative sensor routing protocol from each of these groups.

Acknowledgements

The work described in this paper is based on results of IST FP6 STREP UbiSec&Sens (www.ist-ubisecsens.org). UbiSec&Sens receives research funding from the European Community's Sixth Framework Programme. Apart from this, the European Commission has no responsibility for the content of this paper. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The work presented in this paper has also been partially supported by the Hungarian Scientific Research Fund (contract number OTKA T046664). The first author has been further supported by the HSN Lab.

References

- [1] J. N. Al-Karaki, A. E. Kamal: Routing techniques in wireless sensor networks: a survey. In *IEEE Wireless Communications*, 2004. Vol. 11, pp.6–28.
- [2] C. Intanagonwiwat, R. Govindan, D. Estrin: Directed Diffusion: a scalable and robust communication paradigm for sensor networks. In *Proc. of ACM MobiCom '00*, Boston, MA, 2000, pp.56–67.
- [3] W. Heinzelman, A. Chandrakasan, H. Balakrishnan: Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proc. of the 33rd Hawaii International Conference on System Sciences (HICSS '00)*, January 2000.
- [4] F. Ye, A. Chen, S. Liu, L. Zhang: A scalable solution to minimum cost forwarding in large sensor networks. In *Proc. of the 10th Int. Conference on Computer Communications and Networks (ICCCN '01)*, 2001. pp.304–309.
- [5] C. Rahul, J. Rabaey: Energy Aware Routing for Low Energy Ad Hoc Sensor Networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, Orlando, FL, March 17-21, 2002., Vol. 1, pp.350–355.
- [6] Y. Yu, D. Estrin, R. Govindan: Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks. UCLA Computer Science Dept. Technical Report, UCLA-CSD TR-01-0023, May 2001.

CASCADAS – Autonomic communication and situated pervasive services

BORBÁLA KATALIN BENKŐ, TAMÁS KATONA, RÓBERT SCHULCZ

Budapest University of Technology and Economics, Department of Telecommunications
{bbenko,tkatona,schulcz}@hit.bme.hu

Keywords: *autonomic communication, ACE, pervasive services, self-organization, knowledge network, pervasive supervision*

CASCADAS (*Component-ware for Autonomic, Situation-aware Communications, And Dynamically Adaptable Services*) is one of the four 6th Framework IST-FET projects that aim at providing both theoretical and practical background for a new generation of complex, distributed, pervasive services. The basic building block of the situation-aware, self-organizing, autonomic communication based network is a common abstraction called ACE. ACEs provide and use services, adapt to the situation, create and manage plans, build up knowledge networks, and move and self-organize based on their own autonomic decisions. In CASCADAS, we plan not to categorically rely on the available Layer3 facilities so that to enable the possibility of emerging new-generation protocols/technologies such as utilizing the physical proximity or CPNs. (In: 2006/12, pp.23–28.)

1. Introduction

CASCADAS [1] is one of the four EU 6th Framework IST-FET projects (ANA, HAGGLE, Bionets, CASCADAS) that aim at researching situated, autonomic technologies from different viewpoints [2]. The goal of CASCADAS is to reduce the costs (management, communication, development, configuration etc.) of future emerging complex, highly distributed, pervasive services through a self-organizing network managing knowledge and adapting to the situation.

CASCADAS goals can be formulated on different levels. On the theoretical level, we aim to develop a common abstraction (ACE, meaning Autonomic Communication Element), and based on it, identify/provide models, algorithms and general principles in the fields of self-organization, knowledge network, pervasive supervision and security. In practice, we prove the feasibility/operability of the theoretical models employing demonstrational applications. A secondary goal is to elaborate towards new results in the field of communication. One idea is to work out a new communication protocol that also makes use of the physical proximity in the network, running directly above Layer2.¹ Another idea is that ACEs are running over CPN (Conceptual Packet Network) [3], making use of the self-organization and optimization possibilities provided by the CPN.

2. The CASCADAS vision

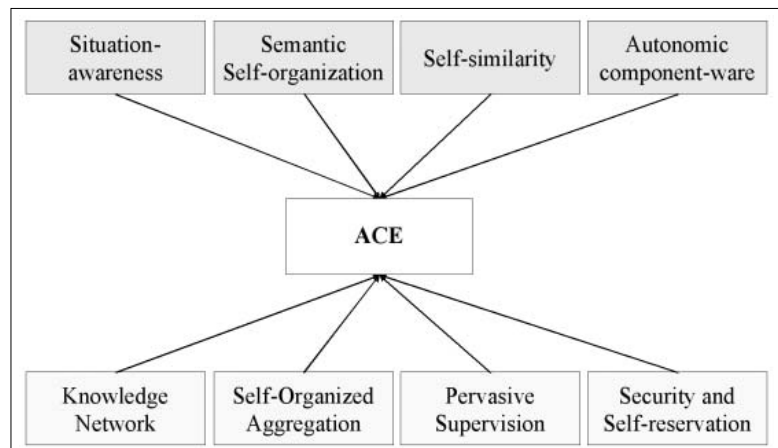
This section elaborates about the motivations and goals of the projects, giving detailed introduction/summary about background principles and the meaning of the keywords.

2.1. Vision

In the CASCADAS vision, the world is proceeding towards pervasive, situation-aware services; the project's goal is to explore emerging problems, elaborate models and provide solutions [4]. The basis of the abstraction is a common lightweight model, the ACE. Future services are envisioned to be available through ACEs (*Figure 1*). ACEs solve communication and management problems in an automatic and autonomic way, seeking for a kind of optimality.

- ACEs perceive and organize knowledge about the situation in order to understand it, including physical, technological, social, user-specific and problem-specific aspects.
- ACEs are capable of self-configuration and self-adaptation. They modify themselves and re-parameterize the services provided, in order to adapt to the situation.

Figure 1. Principles and tools in the CASCADAS vision



¹ Running over Layer2 – besides being an interesting research field – is also a practical consideration, since many concerning operating systems (e.g. TinyOS) support only Layer2.)

- ACEs make up into self-organized structures (e.g. in order to optimize the service or to create a new composed service).
- ACEs have self-* properties (self-healing, self-similarity, self-configuration, etc.)

The project examines autonomic self-organizing communication elements from several viewpoints:

- (1) *The ACE model*. Defining a common abstraction, modelling, creating a framework.
- (2) *Semantic self-organization*. Composition of ACEs, mobility.
- (3) *Knowledge networks*. Creation of knowledge network and knowledge management.
- (4) *Pervasive supervision*. Observing the procedures on the distributed system, and interventions when needed (e.g. self-healing).
- (5) *Security*. Besides classical tasks (authorization, right management, cryptography), reputation based trust models.

2.2. Keywords and the meaning behind

CASCADAS – just like the 3 sister projects – uses several keywords and principles that may not be commonly known in the telecom sector. Many of them have similar, partially overlapping meanings, and also there are some which are just new titles for long-known concepts.

Autonomic

The word “autonomic” gained wide attention first in 2001, as IBM started the “autonomic computing” initiative, aiming to create a self-managing artificial system, similar to the human autonomic nervous system. IBM had identified four (plus one) functional elements that are vital for autonomic operation:

- Self-Configuration: automatic configuration of components.
- Self-Healing: automatic discovery, and correction of faults.
- Self-Optimization: automatic monitoring and control of resources to ensure the optimal functioning with respect to the defined requirements.
- Self-Protection: proactive identification and protection from arbitrary attacks.

There is a difference between autonomic and automatic systems. “Automatic” means that the process is executed by itself, without external intervention. Autonomic means more: besides being of course automatic, it also shows self-* aspects (a kind of intelligence).

CASCADAS ACE is an autonomic component, having all four necessary properties.

Autonomic communication

Autonomic communication means that even the communication has self-* properties. First, the communication technology should be fault-tolerant and self-

optimizing; on the other hand, communication parties themselves are also making decisions in an autonomic way (message sending and receiving, interpreting the message, and the reaction to it).

Having a more abstract look at the problem, it can be said, that the goal of the IBM autonomic initiative was to adapt the system to a more or less known environment. In case of autonomic communication, the environment is highly variant, but through talking to each other, it is possible to get known with it, moreover, also to affect it on a certain level. Autonomic communication also means that the intelligence of the system is not centrally located, but is spread over the components.

Autonomic communication is the basic principle of CASCADAS.

Pervasive services

Pervasive (ubiquitous, everywhere) service means that the source of the service is rather the environment than a distinct computer. This can be interpreted in several ways. It may mean semantic labelling, a locality concept (that the service is only available for nearby clients), intensive logical mobility (the service is moving freely over the network to find the best environment), or that the environment re-organizes itself according to the actual needs (e.g. creates new instances of popular services). Another interpretation is that services simply surround the user (even the toaster is regarded as a computer).

Typically, pervasiveness is supplemented with intelligent aspects such as self-organization or autonomy.

CASCADAS focuses on pervasive services; it's an important element of the vision.

Situation-aware, situated

Situation-awareness (context-awareness, environment-awareness) is a kind of re-consideration of principles that are present in numerous fields (e.g. agent models, control theory). The meaning is that the element observes the context, and reacts accordingly by a four-step process: observation of the context, interpretation (understanding) of the acquired information, calculating the reaction, and response/intervention.

Soon after the introduction of context-awareness models, two problems were identified. Communication boom means that increasing the size of the system, the number of propagated messages (context/state information) increase exponentially; resulting that protocols that worked well for a small system are often not applicable for a real-life, large system. The other problem is about the understanding: can we assume a common ontology being present in the background that guarantees that each ACE will understand the received message correctly (in case it doesn't drop it as unknown)? A possible solution is to have a common environment model that includes the message ontology (which may change or refine in a flexible way with the environment model).

ACE is a situation-aware component.

Locality concept

The concept of locality is closely related to pervasiveness. Locality means that – due to self-organization, mobility, etc. – the service provider and its user are near to each other in the system, so as a result, only local communication is needed (messages are to be propagated to nearby parts of the system as those interested in it are there anyway).

Another interpretation is that the value of an information atom is the highest around its origin place, as we go farther in terms of time and space, its importance/accuracy/correctness decreases (for example, a statement like “it’s 5 o’clock” will be less and less accurate as time goes on, and the truth of “I’m in London” decreases as I’m flying back to Hungary).

In pervasive systems, the locality concept – meaning either logical or physical locations – is intrinsic, as the source of the service is in the direct environment of the user. In CASCADAS, locality concept is intrinsic in the service access model (pervasive services), besides, the knowledge network also supports a kind of local behaviour.

Mobility

In the CASCADAS vision, mobility is a basic property of the ACE. In order to avoid misunderstandings, by “mobility” we do not mean physical mobility (that the ACE leaves the WLAN covered area) but logical mobility (that the ACE is able to move freely among the places of the network that are able to accept it).

3. Tools

Let us discuss in more detail, how CASCADAS is aiming to realize its goals. The tools are: knowledge network building, self-organization, pervasive supervision and a distributed security system.

3.1. Knowledge network

Approaching the problem from the low level, the ACE acquires environment-descriptive information from the knowledge network. But the knowledge network is more than a pure information store or environment abstraction, it is able to organize the knowledge and to optimize itself. To follow the self-similarity paradigm, the knowledge network is built up from the ACEs; and if an ACE has any information that is worth to put into the knowledge network, it can place it there.

Knowledge network supports the concept of locality, so that stored information is important first of all for the neighbourhood, and its value/usefulness decreases with time/distance. Locality can be of logical or physical nature. The difference comes to the surface when ACEs are moving: in physical mobility, the originating location is important, while in logical mobility it is the originator ACE that counts. So, in logical mobility, the information should follow the moving ACE (which is the

easiest to implement if the ACE stores the concerning information in itself).

Knowledge network also supports non-local, self-organizing operation, which is drafted through hierarchical, self-organizing overlays. As for now, intra-overlay communication is local, and non-local communication can be achieved via inter-overlay flows.

The knowledge network consists of knowledge atoms which are the basic building blocks of the knowledge self-organization.

3.2. Self-organization

Self-organization may have several goals, e.g. to help the service and the client to find each other (moving them physically closer), to increase service quality (e.g. by replication of the service, by grouping similar services and using load balancing), or to create new complex services. The basic operation of self-organization is the aggregation which may be weak or strong. Strong aggregation is exclusive (the component is forbidden to take part in other compositions when it participates in a strong composition).

Self-organization may result in structures that are technically overlays. The big difference to common overlay networks is that in the CASCADAS ACE network, there are autonomic elements in the higher levels as well (and as such elements, they’re not guaranteed to work always deterministically from the external point of view).

3.3. Pervasive supervision

Pervasive supervision assures self-healing and self-optimization abilities. The supervisor authority observes the context and the operation of the system (or rather just those ACEs which agreed to be supervised), looking for errors, misbehaviours or processes that can be optimized. If it is needed, the supervisor can also interfere: it may instruct to review a possibly wrong self-organization, or ask an ACE to move, or initiate the healing of a faulty environment-model etc. Supervision is based on a contract, where the supervised ACE binds itself to unconditionally execute the supervisor’s commands (so in some way, its autonomy is limited in the supervised period). Why do we call the supervision “pervasive”? It is, because the pervasive is ubiquitous, it’s present on all levels at the same time (network, communication, content, self-organization, social aspects, etc.).

From the theoretical point of view, supervision is one of the most important tools; it enables ACEs to get out from a quasi stochastically self-organized, error sensitive, somewhat inflexible state and gain chance for self-healing and self-optimization.

3.4. Security

The primary goal of the security subsystem in CASCADAS is to add protection to the system (authentication, authorization, message integrity, DoS protection).

As secondary goal, it contributes to the ACE environment model, through a reputation based trust model. A reputation registry is maintained about the past of ACEs. Knowing the past may help in the current co-operation (good reputation means trust, bad reputation warns).

4. The ACE model

ACE is the common abstraction which the four tools are organized around. ACE is a component model and its mapping to an architecture (architecture is not covered in this paper).

4.1. The basic ACE model

ACE can be examined on different abstraction levels. Let's start from the abstract ones and move towards the direct models [5].

Common part, specific part

CASCADAS ACE consists of two parts: a Common part and a Specific part (Figure 2). The Common part contains those functionalities that are present in all ACE instances. The Specific part contains everything behind that, e.g. service providing ability. The functionality of the Specific part can be explored through communication with the Common part.

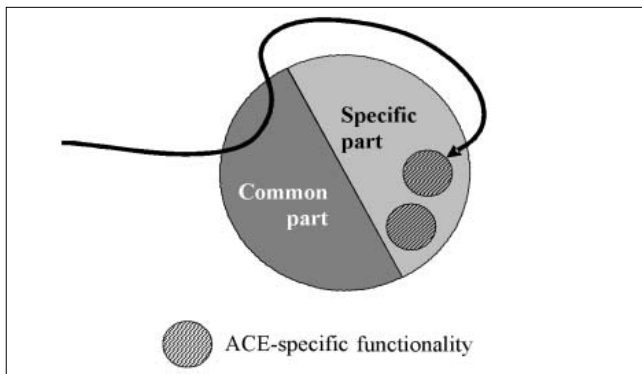


Figure 2. Two parts of the ACE

How does an ACE work?

The operation of ACE is based on its two models: the Self-Model and the Environment Model (Figure 3). Self-Model describes its own operation and goals; while Environment Model models the environment. Both can vary in time: adapt, refine, and get reviewed. As the ACE knows its own possibilities and goal, it is able to create a plan (or more than one plans and choose the best one) and behave as it prescribes. Actions may be reactive (answer for an incoming request/signal) or proactive (there is no external trigger to it). The internal intelligent part of the ACE determines the actions based on the Environment Model and the Self Model (which – besides message sending – can also be the review/altering of a model).

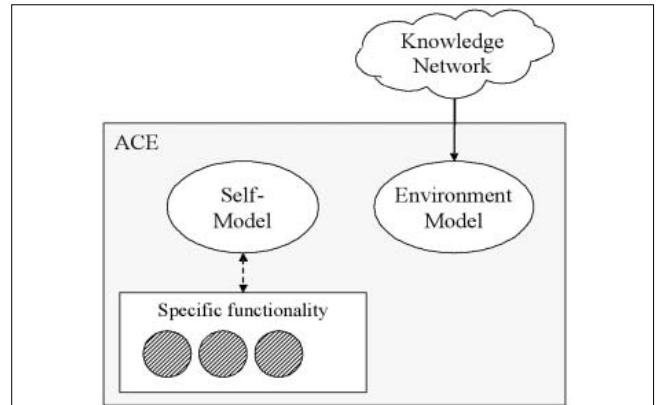


Figure 3.

ACE operates based on the Self- and Environment model

Conceptual model

The conceptual ACE model summarizes the background as a set of short, clear statements (as visualized in an UML diagram as well), see Figure 4.

- ACE provides service towards other ACEs.
- Each ACE resides on one location at the same time (of course, the location may change as the ACE moves).
- ACEs use message based communication. Although the communication is theoretically a 3-step process: discovery (the parties locate each other), contracting (agreeing on the interaction conditions), interaction. In simple cases the steps can be implicit and overlapping (e.g. in case of a broadcasted question and a returned answer, we can say that the question contained a default contract which was accepted as the responder returned the answer), in complex cases, phases may become explicit. There may be intrinsic contracts that are valid from the time of instantiation (e.g. allowing for the usage of SEE ACE services, see later).
- The ACE has two models: a Self-Model and an Environment Model.
- ACE is self-similar in the meaning of a possible aggregation (the aggregate is an ACE, too).
- The ACE creates and manages plans in order to realize its goals.

4.2. Execution Environment

For security and other considerations, a special ACE type was defined: the SEE ACE (Service Execution Environment ACE). The SEE is obligatory to be the first ACE on the location (e.g. on the computer); it is instantiated explicitly, and unable to move. All non-SEE ACEs are free to move with the limitation that on the destination location there must be already at least one ACE (which condition is trivially satisfied by the SEE ACE). So, in other words, ACEs can freely move amongst SEEs.

As it is guaranteed to have an SEE ACE on each location where ACEs may occur, there's a possibility to place a kind of "platform functionality" in the specific

part of it. The mobile ACE first explores the new SEE, and then uses the platform functionality through it.

4.3. The ACE model and the CASCADAS tasks

Knowledge network

The ACE model gains information about its environment (context) through the knowledge network (KN), this is the source of the environment model. KN may answer concrete questions or may provide a subscription based notification service.

From the technical point of view, two models are possible for the relationship of ACEs and the KN: either all ACEs belong to the KN, or there are ACEs that are outside of the KN (but may use it). Our current opinion is that the obligatory KN membership could make the component model too heavy (while a lightweight model is intended). So, in our actual model, the specific part of KN-member ACEs contains the functionality to put knowledge into the KN, to organize it, and to query and delete information.

Self-similarity

ACEs are able to create weak and strong aggregations. In case of weak (or lazy) aggregation, the cooperating parties don't stop autonomous elements, they just bound themselves to a kind of cooperation. According to the cooperation contract, parties may have confidential information about each other (e.g. they know the abstraction of the others' Self-Models), in order to achieve a more successful cooperation. The same ACE can participate in more than one weak aggregation at the same time.

In case of strong aggregation, one can differentiate between the container ACE and the contained ACEs. The concept of the cooperation is that the container ACE has full access rights and control on the contained elements; it is able to access the specific parts of them. Of course, in order to make use of the contained specific functionalities, the contained Self-Models (and Environment Models) need to be integrated into the

container ACE. Please note that in case of strong cooperation, the autonomy of the contained ACEs is basically lost, as all decisions are made by the container, and the contained ACEs are not directly accessible anymore. On the other hand, strong aggregation makes it possible to achieve formerly unreachable things, e.g. the container ACE can freely combine the contained specific part functionalities. Participating in a strong aggregation requires exclusivity.

4.4. Pervasive supervision

Supervision needs to access more information than any other member of the system. It may monitor everything, not only messages, but also the internal parts and processes of the ACE, the Self-Model, the Environment Model, the flow of decision making, and the interaction (output). The supervised ACE obliges itself to make the concerning information available for the supervisor. Theoretically, supervision is able to supervise the common part only, as there are no preliminary assumptions on the specific part (it may be anything: a Prolog engine, some machine level code, or even a disguised human future teller). So, the specific part can be monitored through the input/output channels only, and the observed things can be compared with the abstract description of the functionality (that is part of the Self-Model). It is also possible that the ACE doesn't publish its complete self-model to the supervisor, but only an abstraction of it. In this case, the possibilities of the supervisor are limited (a deterministic error may be observed as non-deterministic²).

Naturally, in order to assure self-similarity, the supervisor is also an ACE – it just meets stricter requirements than a "normal" one (e.g. security).

5. Sample scenarios

Results will be demonstrated through sample scenarios.

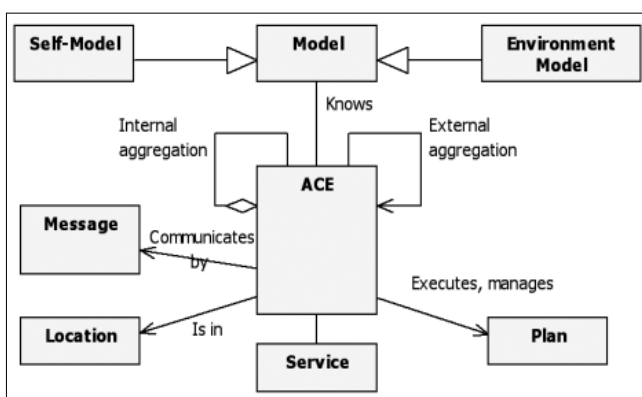
5.1. Pervasive content sharing

Pervasive content sharing consists of several sub-scenarios: pervasive advertisement, friend search and a pervasive (tourist/museum) information system.

Maybe the most interesting one – and the most different from other project – is the pervasive advertisement application scenario. ACE-controlled, adaptive advertisement surfaces are spread over the city (e.g. displays), that observe nearby people's preferences (e.g. based on the user preference descriptor ACE running on the mobile phone), and display the most fitting advertisement on the surface.

For example, for a group of young people, an ad of a rock concert is displayed; while in front of managers, the poster of the newest gold watch is shown. Ethic

Figure 4. Conceptual ACE model



² Let's take a very simple example of cooperation. If John gives Jill an apple, Jill will be happy; but if in the meanwhile, John pulls Jill's hair, Jill won't be happy. If the abstract model doesn't contain information about the way of delivering the apple (with or without pulling the hair) the supervisor won't be able to find out the cause of the error (unhappiness).

problems should be taken into account as well (e.g. even if the majority prefers the trailer of a new horror movie it mustn't be displayed in the case even one young kid is present).

This scenario is intended to demonstrate pervasive, situation-aware elements.

5.2. Distributed auctions

In the distributed auction scenario, users intend buying or selling goods, through creating and parameterizing buyer/seller ACEs and sending them out to the network. As ACEs move on over the network, they can join auctions. As the system assumes to have a very big number of participants, auctions are always operated locally (only nearby elements can participate, and only local communication is needed). So, the goal of an ACE is to get to the most optimal location, both from the network point of view (fast network connection, small communication delay) and content point of view (near to the semantically matching partners).

This scenario is to demonstrate self-organization, the usage of the KN (for self-organization or just to look up how much did a product cost last time), the utilization of the reputation information ("is the partner reliable?", "has it ever cheated?"), and pervasive supervision, accordingly.

6. Towards a new communication model

As the ACE communication is under development, this section gives only a small insight to the ongoing work.

The ACE communication is message-based and ACEs know about themselves which message types they understand. There are common message types understood by all ACEs (e.g. service discovery message, heartbeat towards the supervisor).

At least the following addressing schemes are considered:

(1) *Broadcasting*. The message is addressed to everyone/anyone. The trick in the propagation of such a message is that as the propagation is carried out by ACEs – so autonomous elements –, it is possible that the message won't really reach all network members (e.g. in order to avoid the flood overloading).

(2) *Recipient(s)*. The message is addressed to those ACEs where the ACE ID matches with at least one of the recipient IDs. IDs are not required to be unique, so it is possible to use group addressing or property based addressing (if ID is a set of properties).

(3) *Nearby*. The message is propagated to nearby ACEs only (direct neighbours or a few hops away). The sender doesn't need to know the recipients (not even via properties), they are specified through the structure of the network. This addressing scheme has interesting side effects: e.g. if ACE X sends out a message to the nearby ACEs, and one neighbour (ACE Y) replies with a nearby type message, the recipients of the reply may dif-

fer from the recipients of the original message. A possible solution is to specify the centre (e.g. nearby(ACE X)). The nearby addressing significantly differs from usual "IP world" addressing schemes and it seems to fit well to the requirements of pervasiveness.

(4) *OneOf(list)*. The message should be delivered to at least one of the recipients.

7. Summary

This paper had two goals: to promote the CASCADAS project; and, through this project, to give a draft introduction to today's important concepts, principles and keywords in the field of ambient intelligence.

We gave a general summary about the motivations and goals of the CASCADAS project; and besides discussing the general project vision, we also offered insight into the ongoing work (conceptual ACE model, demonstration scenarios).

What is ambient intelligence in CASCADAS? The project goal is to provide a general ambient intelligence model: there are intelligent elements at all points of the network (even at higher levels) resulting in a fully distributed intelligent system. The system is lead by the autonomic decisions, cooperation and aggregation of intelligent elements; resulting in a multi-level autonomic system with self-healing, self-optimizing and self-configuring abilities.

Acknowledgment

Besides all CASCADAS project members we would like to express our personal thanks to Edzard Hoefig and Fabrice Saffre for the discussions that helped us to better understand the project.

References

- [1] CASCADAS website: <http://www.cascadas-project.org>
- [2] F. Sestini, Situated and Autonomic Communication an EC FET European initiative, ACM SIGCOMM Comp. Com. Rev., Vol. 36, Issue 2, pp.17–20., April 2006.
- [3] E. Gelenbe, R. Lent, A. Montuori, Z. Xu, Cognitive packet networks: QoS and performance. In Proc. of the IEEE MASCOTS Conference, Ft. Worth, Opening Keynote Paper, pp.3–12., October 2002.
- [4] A. Manzalini, F. Zambonelli, Towards Autonomic and Situation-Aware Communication Services: the CASCADAS Vision, In Proc. of IEEE Workshop on Distributed Intelligent Systems, Prague, 2006.
- [5] E. Hoefig, B. Wuest, B. K. Benko, A. Mannella, M. Mamei, E. Di Nitto, On Concepts for Autonomic Communication Elements, In Proc. of IEEE International Workshop on Modelling Autonomic Communications Environments, Dublin, 2006.

Applying fuzzy inference in the supervision system of mobile telecommunication networks

LÁSZLÓ T. KÓCZY⁺, JÁNOS BOTZHEIM, RICHÁRD SALLAI, KORNÉL CSÁNYI

Budapest University of Technology and Economics, Dept. of Telecom. and Media Informatics

⁺Institute of Inf. Technology, Mechanical and Electrical Engineering, Széchenyi University, Győr

TAMÁS KUTI

Linecom Ltd., Budapest, Hungary

Keywords: mobile networks, supervision systems, intelligent techniques, fuzzy logic

In mobile telecommunication networks the transmission level is affected by objects located between transmitter and receiver stations in the so-called Fresnel-zone. These obstacles may get temporally or permanently into the zone, they can be artificial or natural objects, or they can be of meteorological origin, too, like rain or fog. It is reasonable to use intelligent decision making subsystems which can decide from the degree of attenuation and from its time dependent behaviour what the reason of the attenuation could be. The paper demonstrates the intelligent module of a network supervision system created in the framework of a successfully completed National R&D Programme project. Intelligent decision support systems and the basics of fuzzy logics are introduced. In the next an application for automatically identifying the weather situation is discussed in some detail.

(In: 2006/12, pp.52–59.)

1. Introduction

It is a widely known fact in connection with microwave networks that the transmission level is strongly affected (namely, decreased) by objects or substances located between any pairs of transmitter and receiver stations, in the so-called Fresnel-zone. These objects and substances, simply obstacles, may get into the zone temporally or permanently. Also, they may be artificial or natural objects, and they might be of meteorological origin, too, like rain or fog. If unexpected decrease of the transmission quality is perceived, an alarm or indication of some action to be immediately done is usually indicated at the operator screen. While the investigation of the reason of such service degradation by human staff is a generally successful method for taking care of the problem, it is far from being the most economical solution. In such cases it is reasonable to use intelligent decision making in the network supervision system which can autonomously decide from the degree of attenuation and from its time dependent behaviour what the original reason of the decrease in the signal level could be. The system might automatically recommend the necessary action in order to eliminate or compensate the unwanted behaviour at the user interface of the supervision system.

This paper presents the intelligent module of a network supervision system created in the framework of a successfully completed National R&D Programme project [5]. This module performs intelligent inference taken from the change of the transmission level (decrease values calculated from the signal levels at the transmitter and receiver end). Such changes are deduced from the values read by the sensors of the supervision system. The result of this inference is present-

ed via the Graphical User Interface (GUI) of the system. Here, we focus only on one issue in this study, namely on the intelligent recognition of different precipitation categories that may occur in the temperate continental climatic zone.

Different alarm levels, which will be illustrated later by some examples, can be divided into two categories. The first one contains hardware failures and functional decay of the equipment at the microwave stations which may occur suddenly or gradually during a longer period of operation time. The system initiates some (human staff related) maintenance action in such cases. The second category covers the phenomena when the received signal level decreases because of various obstacles gotten into the Fresnel-zone. A typical example is the setting up of a poster in an urban environment, which can significantly damage the quality of transmission between two stations. (This often happens on the roof of a building, while stations are also located at various roofs, thus a bigger poster might completely block the visibility of the two stations from each other.) In such a case a measure totally different from the previously mentioned maintenance action is needed, e.g. the relocation of the transmitter and/or the receiver station will be necessary.

Another example for this category of phenomenon is when in a rural neighbourhood a forest located between transmitter and receiver stations gets leaves in the spring. By this the total area of covered cross section within the Fresnel zone increases tremendously, thus decreasing the transmission quality. This phenomenon may be compensated by the automatic resetting of the transmitter performance cyclically and annually. The previous two examples were typical illustrations for an artificial and a natural obstacle.

The scope of problems investigated in our demonstration system belongs to the category of meteorological phenomena. The separation of the phenomenon of fading caused by rain/fog and the transmission problems caused by different obstacles can be done by recognising the more or less isotropic behaviour of the decrease of the received signal level occurring in a geographically closed area. The lack of directionality and simultaneousity of the decrease are usually good indicators of rainfall in the neighbourhood of a given station. Often a group of stations with a number of station pairs are observed at the same time in order to recognise isotropy.

It is worth mentioning that these two phenomenon groups may exceptionally result in a combined effect, such as when a leafy forest located between the transmitter and receiver stations receives rain that is accumulated on the surface of the leaves, and so, even for longer period after the rainfall, may cause strong anisotropic fading because of the water drops on the leaves that act together as thousands of tiny refractors for the microwave signals.

Next, the background of the computational intelligence method applied to intelligent decision making is discussed. In Section 2, the foundations of fuzzy systems are introduced. Hierarchical fuzzy systems are briefly presented in Section 3. The fuzzy system applied for the supervision of mobile telecommunication network is demonstrated in Section 4.

2. Foundations of fuzzy systems

Human reasoning and some other phenomena cannot be accurately described by two-valued logic. The desire for extending two-valued logic to multiple valued ones came up long time ago. The basic concept was that instead of using only the “true” and “false” logical values, the use of other values between true and false should also be allowed. There are many statements that cannot be evaluated as true or false, only their respective “degree of truth” can be determined.

This idea led L. A. Zadeh to the creation of fuzzy logic in 1965 [2]. Nowadays, in computers and in many areas of life, the classical binary (Aristotelean or Boolean) logic is used. However, if we want to create more intelligent tools, better results can be achieved if the behaviour of the systems is described in a way that is closer to human thinking. Fuzzy logic is an extension of the classical logic. A fuzzy logic variable may assume any value between 0 and 1. Here, 0 means that the statement is “totally false”, while 1 means that it is “totally true”. According to this definition, the value 0.5 corresponds to “half true”, and value 0.9 to

“almost true”. The operations of classical logic can also be extended to fuzzy logic. Based on this concept, fuzzy sets can be defined, fuzzy rules and fuzzy inference systems can be created. The next sections give a brief overview of the basic ideas (for more detailed description refer to [1]).

2.1. Fuzzy sets

A fuzzy set A defined over a universe of discourse X is characterised by the so-called membership function (which is the extension of the characteristic function of ordinary sets). The membership function μ_A assigns a real value from the closed unit interval to every element of X describing the degrees for elements x to which they are belonging to the fuzzy set A :

$$\mu_A : X \rightarrow [0, 1]$$

μ_A unambiguously characterises the fuzzy set A if the universe of discourse X is also known. In the practical applications the most commonly used shapes for membership functions are triangular, trapezoidal, or sometimes more general piecewise linear (like in the very first real application by Mamdani [3]) and symmetrical or asymmetrical Gaussian.

In *Figure 1* examples for simple attenuation “values” described by fuzzy sets can be seen. Three categories are distinguished here, “moderate”, “medium” and “high”. Obviously these fuzzy values are rather extended intervals which comprise whole sets of concrete values which are considered to be more or less equivalent from the point of view of a certain application. Very likely, in another application the number of sets and corresponding labels and the extension of each set will be different.

In this example the shape of the membership functions is trapezoidal. Two important basic definitions need to be mentioned in connection with fuzzy sets, namely the *support* and the *core* of a fuzzy set. The support of fuzzy set A is the (ordinary) set of those elements of the universe which have positive membership value in A :

$$\text{supp}(A) = \{x \in X \mid \mu_A(x) > 0\}.$$

The core of A means those elements of the universe whose membership value is equal to 1, i.e. which belong to the fuzzy set in the ordinary sense or completely:

$$\text{core}(A) = \{x \in X \mid \mu_A(x) = 1\}.$$

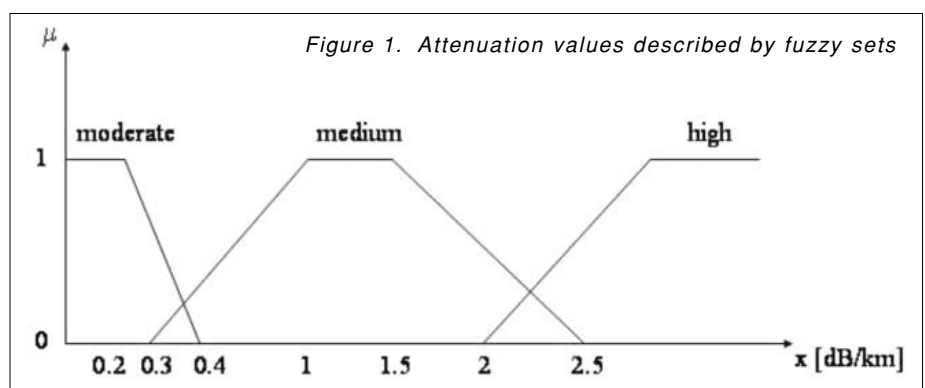


Figure 1. Attenuation values described by fuzzy sets

The three main set operators union, intersection and complementation of classical set theory can be extended to fuzzy sets in several (in fact infinite many) ways. The most commonly used ones are the standard definitions introduced originally by Zadeh [2], but the so-called algebraic operators have also some advantageous properties. The *standard complement* of fuzzy set A on a universe X is \bar{A} , where for each

$$\{x \in X : \mu_{\bar{A}}(x) = 1 - \mu_A(x)\}.$$

The *standard intersection* of fuzzy sets A and B is given by

$$\mu_{A \cap B}^Z(x) = \min(\mu_A(x), \mu_B(x)),$$

while the *standard union* is

$$\mu_{A \cup B}^Z(x) = \max(\mu_A(x), \mu_B(x)).$$

The *algebraic intersection* can be calculated from

$$\mu_{A \cap B}^I(x) = \mu_A(x) \cdot \mu_B(x),$$

and the *algebraic union* is defined by

$$\mu_{A \cup B}^I(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)$$

There is no separate definition for the algebraic complementation, moreover, the standard complementation satisfies De Morgan's Laws together with the two algebraic operations, just like it does the same with the standard operations. Such triplets of fuzzy operations are referred to as De Morgan triplets.

2.2. Fuzzy rules

Fuzzy knowledge bases form the essence of fuzzy control and decision support. They are constructed from a set of fuzzy rules. Fuzzy rules can be formulated by fuzzy sets and linguistic labels using often natural human language. In a supervision system for telecommunication networks, e.g. the following rule can be formulated:

“If the decrease of the received signal level is *moderate* **and** the elapsed time is *short* **then** the precipitation is *moderate rainfall*.”

If the membership functions for “moderate”, “short”, and “moderate rainfall” are exactly defined over the universal sets “received signal levels”, “time between two observations” and “amount of precipitation”, then fuzzy rules are obtained. The general form of a fuzzy rule with one input and one output dimension is as follows:

R : **If** x is A **then** y is B ,

where $x \in X$ is the input and $y \in Y$ is the output variable, X is the universe of discourse for the input and Y is the universe of discourse for the output variable. A and B are linguistic labels that are expressed by fuzzy sets. Set A is the antecedent while B is the consequent of rule R . The general form of a fuzzy rule with multiple inputs and one output dimension can be written in the following, so called Mamdani type orthogonally decomposed form [3]:

R : **If** x_1 is A_1 **and** ... **and** x_n is A_n **then** y is B ,

where $x = (x_1, \dots, x_n)$ is the input vector, $x_j \in X_j$, $X = X_1 \times \dots \times X_n$ is the n -dimensional universe,

$A = (A_1, \dots, A_n)$ is the antecedent vector, $A \tilde{\subset} X$, $y \in Y$ is the output variable, Y is the universe for the output and B is the consequent set, $B \tilde{\subset} Y$. (Here $\tilde{\subset}$ denotes fuzzy subsethood.) A rule can be applied if every input variable has a positive membership value in its corresponding antecedent set. In case of multiple output rules, the outputs are independent from each other, thus this kind of rules can be decomposed to fuzzy rules with one single output, reducing the computational demand in this way.

2.3. Fuzzy inference systems

The fuzzy sets based approach is suitable for describing (very) complex systems which cannot be modelled analytically. By fuzzy sets, operations and rules, inference systems may be created which imitate in some sense the ways of everyday human thinking. Such systems are referred to in the literature as *fuzzy systems*. The structure of a typical fuzzy system is illustrated in Figure 2.

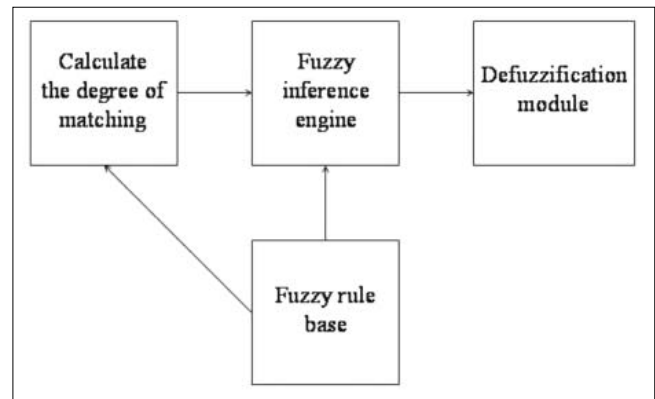


Figure 2. Structure of a fuzzy inference system

The first (top left) module compares the actual observation with the antecedent parts of the fuzzy rules in the rule base (located in the bottom block of the system). Based on this comparison, the inference engine (top middle unit) determines the resulting output fuzzy set by some inference algorithm. There are some well-known inference techniques, however, the Mamdani method is the most commonly used one in practical applications [3]. The inference engine may be viewed as a special kind of generalised function generator as it maps the set of all possible input fuzzy sets into the set of all possible fuzzy outputs. The output is converted to a so-called “crisp” value by the defuzzification module (top right). The Mamdani inference algorithm is illustrated with fuzzy membership functions in Figure 3.

At the beginning of the inference the degree of matching between the observation and the rules is determined. Each component of the observation vector is compared to the same component of the antecedent of each rule. Let A^* be the n -dimensional observation vector. The degree of matching (firing) in the j^{th} dimension in the i^{th} rule can be computed as:

$$w_{j,i} = \max_{x_j} \left\{ \min \left\{ A_j^*(x_j), A_{j,i}(x_j) \right\} \right\}$$

where $A_{j,i}$ is the membership function of the i^{th} rule in the j^{th} dimension. If the observation is a crisp vector then the above calculation is simpler: in case of state-vector x^* , the degree of matching in the j^{th} dimension is:

$$w_{j,i} = A_{j,i}(x_j^*).$$

After the degree of matching was calculated in each dimension, the resultant for the whole antecedent is determined. The degree of applicability of a rule is affected by the degree of matching of its each dimension. Thus, the firing degree of the i^{th} rule can be computed by taking the minimum value of the degrees of matching of the rule's antecedents:

$$w_i = \min_{j=1}^n w_{j,i}.$$

w_i shows that how important the role of rule R_i will be in the calculation of the conclusion for observation A^* .

After the degree of firing was determined for each rule, each conclusion is separately calculated. This can be made by cutting the consequent fuzzy set of the rule at height w_i :

$$B_i^* = \min(w_i, B_i(y)).$$

The conclusion for the whole rule base can be computed by taking the union of the previously calculated sub-conclusions:

$$B^*(y) = \max_{i=1}^r B_i^*(y).$$

After the inference a $B^*(y)$ conclusion fuzzy set was obtained. However, in most of the cases, the expected conclusion is not a fuzzy set, but a crisp value. Hence, the crisp value needs to be determined, which describes the conclusion fuzzy set in the best way. This procedure is called defuzzification. There are many different defuzzification methods described in the literature, in this particular application the Centre of Gravity (COG) method is

applied, which is one of the most commonly used defuzzification techniques in practical applications. The COG method provides a crisp result that can be calculated as follows:

$$y_{COG} = \frac{\sum_{i=1}^r \int_{y \in B_i} B_i^*(y) dy}{\sum_{i=1}^r \int_{y \in B_i} B_i^*(y) dy}$$

In the next section a more advanced approach is briefly sketched that is suitable for handling systems with a very large number of components.

3. Hierarchical fuzzy systems

The idea of structuring very large system in a hierarchical way came up at the beginning of 1990s. Hierarchical fuzzy systems have been successfully applied for some special problems, where the hierarchical structure is more or less obvious, eminently the unmanned helicopter control experiment by Sugeno [4].

The basic idea of using hierarchical fuzzy rule bases is the following: If the multi-dimensional input space $X = X_1 \times X_2 \times \dots \times X_m$ can be decomposed, so that some of its components, e.g. $Z_0 = X_1 \times X_2 \times \dots \times X_p$ determine a subspace of X ($p < m$), where in Z_0 a partition $\Pi = \{D_1, D_2, \dots, D_n\}$ can be determined: $\bigcup_{i=1}^n D_i = Z_0$

E.g. in the unmanned helicopter control application, different variables are dominating the behaviour when hovering, landing, or flying forward and each of these manoeuvres means a different local subsystem.

In each element of Π , i.e. D_i , a sub-rule base R_i can be constructed with local validity. In the worst case, each sub-rule base refers to exactly $X/Z_0 = X_{p+1} \times \dots \times X_m$.

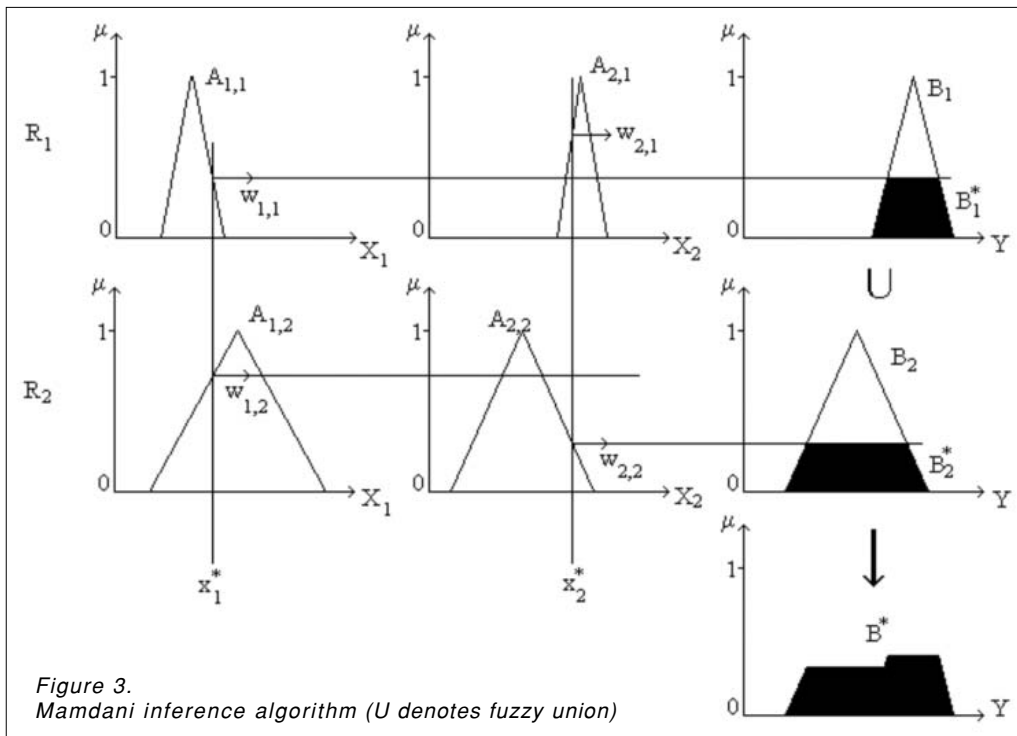


Figure 3. Mamdani inference algorithm (U denotes fuzzy union)

The complexity of the whole rule base $O(T^m)$ is not decreased, as the size of R_0 is $O(T^p)$, and each $R_i, i > 0$, is of order $O(T^{m-p}), O(T^p) \times O(T^{m-p}) = O(T^m)$.

A way to decrease the complexity would be finding in each D_i a proper subset of $\{X_{p+1} \times \dots \times X_m\}$,

so that each R_i contains only less than $m-p$ input variables. In some concrete applications in each D_i a proper subset of $\{X_{p+1}, \dots, X_m\}$

can be found so that each R_i contains less than $m-p$ input variables.

The rule base has the following structure:

R_0 : **If** z_0 is D_1 **then** use R_1
 If z_0 is D_2 **then** use R_2
 ...
 If z_0 is D_n **then** use R_n

R_1 : **If** z_1 is A_{11} **then** y is B_{11}
 If z_1 is A_{12} **then** y is B_{12}
 ...
 If z_1 is A_{1r1} **then** y is B_{1r1}

R_2 : **If** z_2 is A_{21} **then** y is B_{21}
 If z_2 is A_{22} **then** y is B_{22}
 ...
 If z_2 is A_{2r2} **then** y is B_{2r2}

...

R_n : **If** z_n is A_{n1} **then** y is B_{n1}
 If z_n is A_{n2} **then** y is B_{n2}
 ...
 If z_n is A_{nrn} **then** y is B_{nrn}

where $z_i \in Z_i$, $Z_0 \times Z_i$ being a proper subspace of X for $i = 1, \dots, n$. The fuzzy rules in rule base R_0 are termed meta-rules since the consequences of the rules are pointers to other sub-rule bases instead of fuzzy sets.

If the number of variables in each Z_i is $k_i < m - p$ and $\max_{i=1}^n k_i = K < m - p$, then the resulting complexity will be $O(T^{p+K}) < O(T^m)$, so the structured rule base leads to a reduction of the complexity.

The task of finding such a partition is often difficult, if not impossible. (Sometimes such a partition does not even exist). There are cases when, locally, some variables unambiguously dominate the behaviour of the system, and consequently the omission of the other variables allows an acceptably accurate approximation. The bordering regions of the local domains might not be however crisp or even worse, these domains overlap. For example, there might be a region D_1 , where the proper subspace Z_1 dominates, and another region D_2 , where another proper subspace Z_2 is sufficient for the description of the system, however, in the region between D_1 and D_2 all variables in $[Z_1 \times Z_2]$ play a significant role ($[. \times .]$ denoting the space that contains all variable that occur in either argument within the brackets).

In this case, sparse fuzzy partitions can be used, so that in each element of the partition a proper subset of the remaining input state variables is identified as exclusively dominant. Such a sparse fuzzy partition can be described as follows:

$$\tilde{\Pi} = \{D_1, D_2, \dots, D_n\} \quad \text{and} \quad \bigcup_{i=1}^n \text{Core}(D_i) \subset Z_0$$

in the proper sense (fuzzy partition).

Even $\bigcup_{i=1}^n \text{Supp}(D_i) \subset Z_0$ is possible (sparse partition).

If the fuzzy partition chosen is informative enough concerning the behaviour of the system, it is possible to interpolate its model among the elements of $\tilde{\Pi}$.

Each element D_i will determine a sub-rule base R_i referring to another subset of variables.

A part of the hierarchically fuzzy rule base in the above mentioned helicopter control problem is:

R_0 : **If** *distance* (from obstacle) is *small* **then** *hover*
 Hover: **If** (helicopter) *body* rolls *right*
 then move *lateral* stick *leftward*
 If (helicopter) *body* pitches *forward* **then**
 move *longitudinal* stick *backward*

4. Application of the fuzzy system approach

As it has been already mentioned, in the frame of this project [5] a fuzzy system was applied as the supervision module of a telecommunication network. In this paper just one function is described. The system determines the type of precipitation in the geographical area under supervision, based on the available transmitted and received signal levels coming from the telecommunication network. Based on these data, a human operator can get a clear idea of the network's actual status, and therefore the operator can make an optimal decision about the necessary action. The system makes the decision based on two input parameters, the first one being the decrease of the received signal level, the second one the elapsed time.

The application is able to receive data from base stations, and these data are stored in a database together with the corresponding time stamps. It gets the data from the stations cyclically, and it determines the change of signal level and the corresponding time by the comparison of the new and old data, and it makes an inference based on these informations. The possible results of the inference may be divided into two groups. The first group contains conclusions which refer to events of meteorological origin, while the other one is related to some breakdown that might cause an alarm signal.

The alarms will usually cause some activity triggered by a human operator while the conclusion that unambiguously refers to transmission trouble caused by weather conditions clearly excludes the necessity of any maintenance type action rather than the mere compensation of the decreased transmission level by pushing up the power on the transmitter side. Any technical problem will have a very different behavioural pattern.

The mathematical relationship between the change of signal level and the amount of precipitation can be calculated as follows: $\gamma = kR^\alpha$ where γ is the decrease of signal level, R is the amount of precipitation, k and α are parameters, which depend primarily on the used frequency.

We would like to replace the inverse of this equation by a fuzzy model, i.e. to determine the amount of precipitation from the decrease of signal level taking into account the elapsed time, as well. The relationship between individual variables can be described better by using fuzzy rule bases, because the fuzzy sets used in the rules separate the possible values of the variables not by a crisp border but in a smooth gradual way similarly to everyday human thinking.

4.1. Determination of the fuzzy sets

The fuzzy rule based system uses two input variables for each pair of stations, the decrease of the signal level, and the elapsed time. The decrease of the signal level is divided into six categories:

- very moderate attenuation: 0 – 0.05 dB/km
- moderate attenuation: 0.03 – 0.18 dB/km
- medium attenuation: 0.15 – 0.7 dB/km
- significant attenuation: 0.5 – 2.5 dB/km
- high attenuation: 1.8 – 5.5 dB/km
- very high attenuation: 3.3 – 18 dB/km

The fuzzy sets are determined by these intervals. The intervals give the support of the fuzzy sets. The fuzzy sets for the decrease of the signal level can be seen in *Figure 4*. It can be observed in *Figure 4* that the supports of the fuzzy sets overlap each other, which

means that the borders between the different attenuation groups are not crisp.

The elapsed time is divided into four categories: short, medium, long, very long. The corresponding approximate intervals are:

- short: 0 – 1 hour
- medium: 0,5 – 4 hours
- long: 3 hours – 4 days
- very long: 3 days – 1 year

The fuzzy sets are constructed from the intervals in a similar way as previously. The fuzzy sets for the elapsed time are illustrated in *Figure 5*. The supports overlap in this case, too.

The output of the fuzzy system contains the reference to different precipitation types/intensities. The precipitation can be categorised as follows:

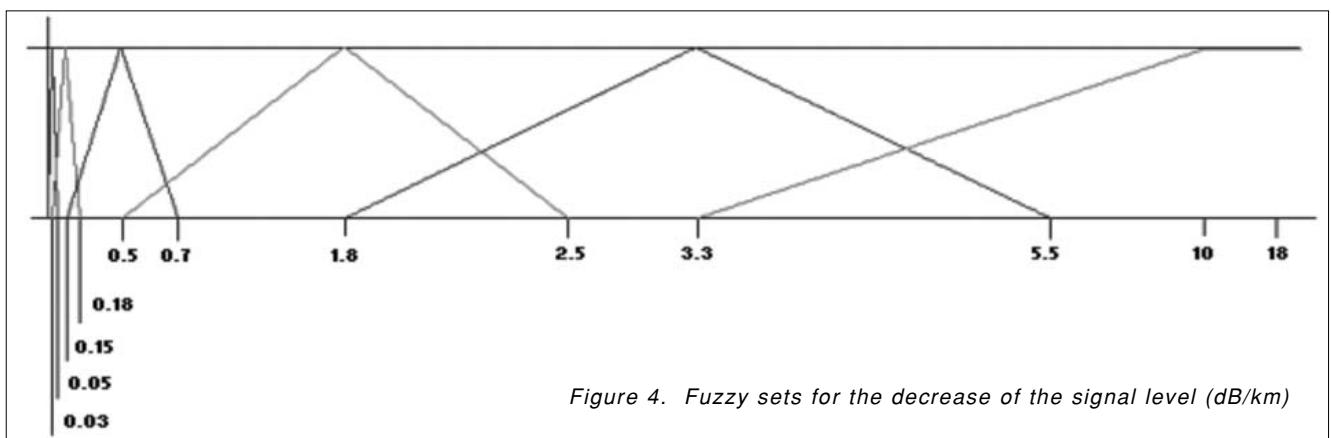


Figure 4. Fuzzy sets for the decrease of the signal level (dB/km)

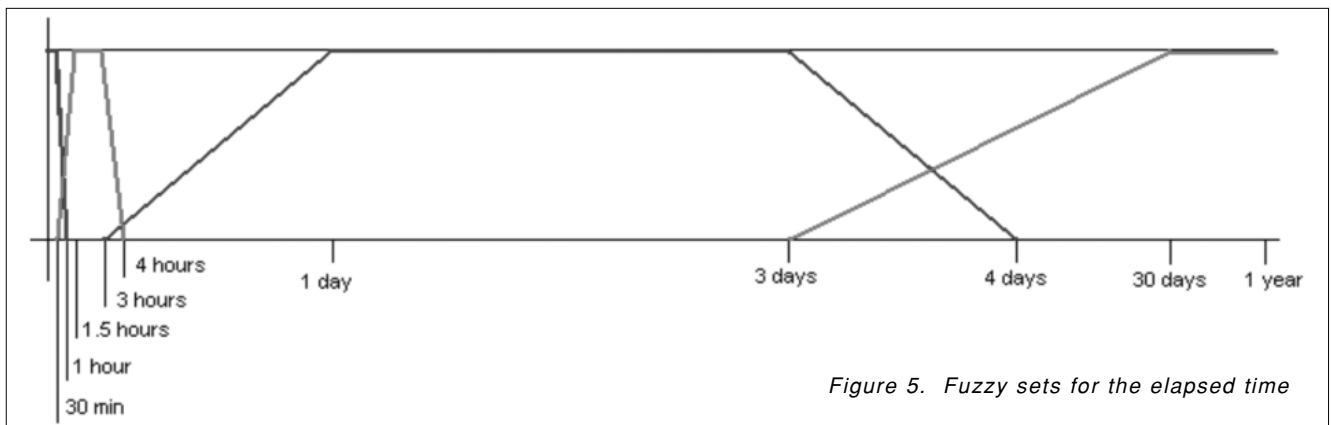


Figure 5. Fuzzy sets for the elapsed time

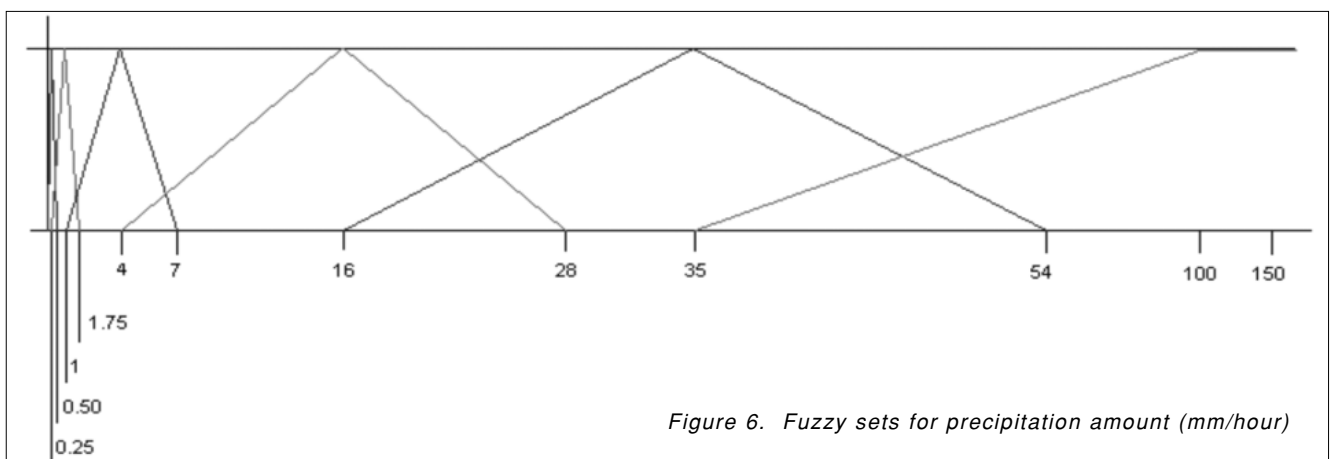


Figure 6. Fuzzy sets for precipitation amount (mm/hour)

- drizzle: 0 – 0,5 mm/hour
- moderate rainfall: 0,25 – 1,75 mm/hour
- medium rainfall: 1 – 7 mm/hour
- heavy rainfall: 4 – 28 mm/hour
- thunderstorm: 16 – 54 mm/hour
- intensive thunderstorm: 35 – 150 mm/hour

The fuzzy sets belonging to the precipitation categories can be seen in *Figure 6*.

4.2. The fuzzy rules of the system

The first input of the system was described by six fuzzy sets, the second one by four, therefore the total number of possible combinations is 24, which means that 24 rules are needed to cover all possibilities. In the conclusion part of the fuzzy rules may occur not only precipitation categories, but also various types of alarms. These alarms denote reasons for service deterioration which are of non-meteorological origin.

The first input is characterised by six labels. The decrease of signal level is A_i , where i might be:

- very moderate attenuation: 1
 - moderate attenuation: 2
 - medium attenuation: 3
- significant attenuation: 4
 - high attenuation: 5
 - very high attenuation: 6

The elapsed time is characterised by labels T_i , where i might assume:

- short: 1
- medium: 2
- long: 3
- very long: 4

Using the above labels the following rules were constructed:

- R_1 : If the attenuation is A_1 and the elapsed time is T_1 then the precipitation is *drizzle*
 R_2 : If the attenuation is A_1 and the elapsed time is T_2 then the precipitation is *drizzle*
 R_3 : If the attenuation is A_1 and the elapsed time is T_3 then the precipitation is *drizzle*
 R_4 : If the attenuation is A_1 and the elapsed time is T_4 then *Warning*
 R_5 : If the attenuation is A_2 and the elapsed time is T_1 then the precipitation is *moderate rainfall*
 R_6 : If the attenuation is A_2 and the elapsed time is T_2 then the precipitation is *moderate rainfall*
 R_7 : If the attenuation is A_2 and the elapsed time is T_3 then the precipitation is *moderate rainfall*
 R_8 : If the attenuation is A_2 and the elapsed time is T_4 then *Warning*
 R_9 : If the attenuation is A_3 and the elapsed time is T_1 then the precipitation is *medium rainfall*
 R_{10} : If the attenuation is A_3 and the elapsed time is T_2 then the precipitation is *medium rainfall*
 R_{11} : If the attenuation is A_3 and the elapsed time is T_3 then the precipitation is *medium rainfall*
 R_{12} : If the attenuation is A_3 and the elapsed time is T_4 then *Minor alarm*
 R_{13} : If the attenuation is A_4 and the elapsed time is T_1 then the precipitation is *heavy rainfall*
 R_{14} : If the attenuation is A_4 and the elapsed time is T_2 then the precipitation is *heavy rainfall*
 R_{15} : If the attenuation is A_4 and the elapsed time is T_3 then *Minor alarm*
 R_{16} : If the attenuation is A_4 and the elapsed time is T_4 then *Minor alarm*
 R_{17} : If the attenuation is A_5 and the elapsed time is T_1 then the precipitation is *thunderstorm*
 R_{18} : If the attenuation is A_5 and the elapsed time is T_2 then the precipitation is *thunderstorm*
 R_{19} : If the attenuation is A_5 and the elapsed time is T_3 then *Major alarm*
 R_{20} : If the attenuation is A_5 and the elapsed time is T_4 then *Major alarm*
 R_{21} : If the attenuation is A_6 and the elapsed time is T_1 then the precipitation is *intensive thunderstorm*
 R_{22} : If the attenuation is A_6 and the elapsed time is T_2 then *Major alarm*
 R_{23} : If the attenuation is A_6 and the elapsed time is T_3 then *Major alarm*
 R_{24} : If the attenuation is A_6 and the elapsed time is T_4 then *Major alarm*

This rule base is used as a single unit of a more complex hierarchically structured knowledge base where the (approximate) isotropy of the attenuation observed plays the decisive role concerning the original cause for alarm.

It must also be mentioned that the values of attenuation are not identical with the directly observed ones but are deduced values from interpolating and averaging the values calculated from the direct measurement values according to the physical and geographical locations of the base stations in a particular neighbourhood.

4.3. The inference method

At first, the firing values of the rules are determined based on the comparison of the given observation and the antecedent parts of the rules. The rules can be divided into two groups according to their outputs. If the firing values for those rules are greater than for those which contain an alarm indication in their respective consequent, the conclusion of the fuzzy system will be the same as the conclusion of the alarm type fuzzy rule with the highest firing degree. In the opposite case, when the firing values for those rules are greater than for those, which have weather related information in their respective consequent, a normal Mamdani inference is performed on all the involved precipitation type rules. As the result, the most possible type of precipitation will come out as dominating in the overall consequent.

The conclusion given by the fuzzy system can be displayed in a map. The colours of the cells refer to precipitations.

Colour	Precipitation type	Colour code
	Drizzle	1
	Moderate rainfall	2
	Medium rainfall	3
	Heavy rainfall	4
	Rainstorm	5
	Intense thunderstorm	6

Table 1. Precipitation types and the assigned colours

The assignment of colours and precipitation types can be seen in Table 1. The different alarm categories can also be represented by colour codes. The assignment is shown in Table 2.

The operation of the system is illustrated in Figure 7 with a map, which was created by simulated precipitation. The map of Hungary is covered by a grid, giving a natural, hierarchical structure for the whole system in this way. On such locations where more stations can be found within one cell, an average behaviour is calcu-




Colour	Alarm	Colour code
	Warning	7
	Minor alarm	8
	Major alarms	9

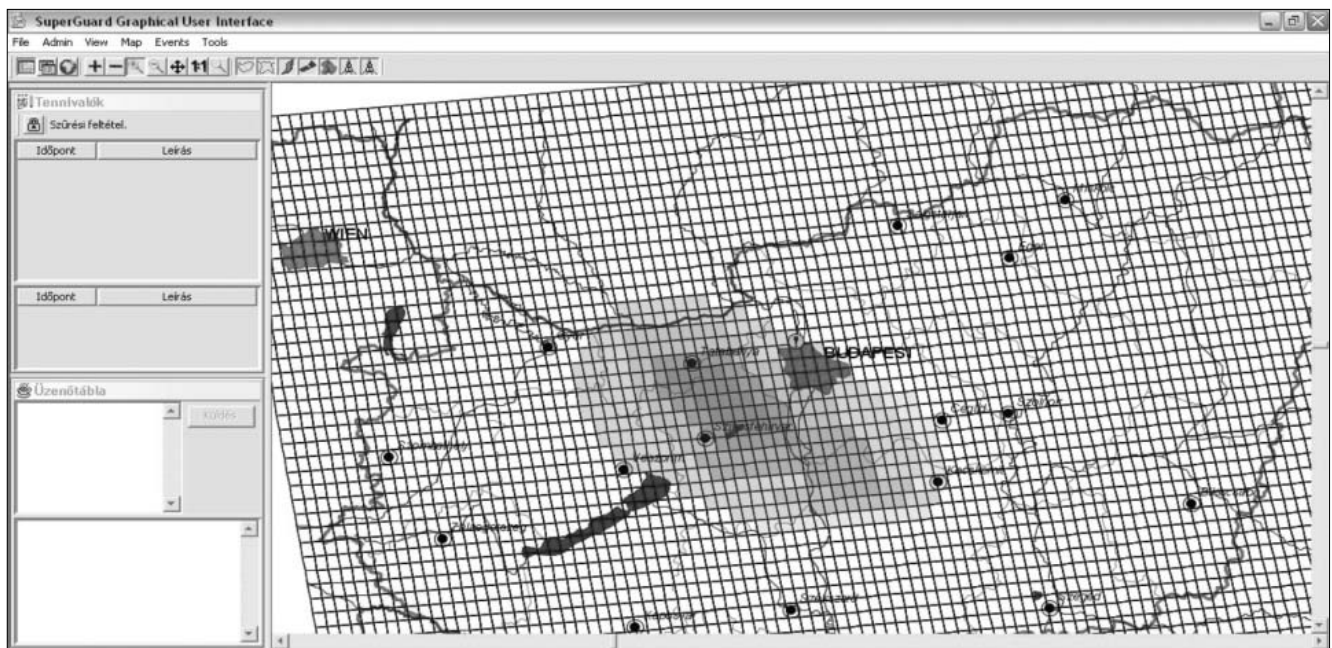
Table 2. Alarms and their colours

lated and displayed, on the other hand, on such places where there is no station within one cell, interpolation is used to estimate the amount of precipitation. The blue and purple colours show the estimated amount of precipitations based on simulation data.

5. Summary

In this study the foundations of fuzzy systems were introduced, hierarchical fuzzy systems were also discussed, and an application example was demonstrated. Fuzzy systems can be used well as a decision support tool in such applications, where the expert knowledge can be easily represented in form of fuzzy rules. In laboratory environment an application was demonstrated, which can separate alarms of meteorological origin from different kinds of hardware failures based on a cyclically refreshed database in a central supervision system of a nationwide microwave telecommunication network, as well as it can identify the intensity

Figure 7. Rain cloud and simultaneous Warning in Budapest on the map of Hungary



map of precipitation on the whole geographical area of the telecommunication network.

Our plan for the future is to extend the intelligent decision support system to proper hierarchical fuzzy rule based system, making the model more general and more sensitive in this way. Currently a network provider in Hungary indicated their possible interest in experimentally integrating the intelligent decision support sub-system into their real operating supervision system.

References

- [1] G. J. Klir, B. Yuan: Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice Hall, 1995.
 [2] L. A. Zadeh: Fuzzy sets, Information and Control, 8(3):338–353., 1965.

- [3] E. H. Mamdani, S. Assilian: An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. Int. Journal Man-Mach. Stud. 7: 1–13., 1975.
 [4] M. Sugeno, M. F. Griffin, A. Bastian: Fuzzy hierarchical control of an unmanned helicopter, 5th IFSA World Congress (IFSA'93), pp.1262–1265., Seoul, 1993.
 [5] Supervision system containing computational intelligence, Project report. Project ID: NKFP-2/0015/2000, Principal Investigator: L. T. Kóczy, Dept. of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Associate Investigator: L. Veres, Linecom Ltd., 2005. (In Hungarian)

"Dennis Gabor" Award for Prof. Gyula Sallai

The prestigious "Dennis Gabor" Award was founded by Novofer Co. 18 years ago. Its title bears the name of the world-famous Hungarian scientist, Dennis Gabor, the inventor of the holography who received a Nobel prize for this invention and for his achievements in information theory. The Dennis Gabor Award is granted yearly to selected Hungarian professionals for their excellence in innovation. It also has an international edition, which is granted once in every three years.

In the past, several prominent professionals, who played a leading role in our Scientific Society and in the Journal of Telecommunications, received this prize, including Professors Géza Gordos, György Lajtha and László Pap.

The award ceremony this year was held in the prestigious building of the Hungarian Parliament. Congratulations to Prof. Dr. Gyula Sallai, the President of the Scientific Society of Infocommunications who was among this year's winners of the Dennis Gabor award. As it was emphasized at the award ceremony, Professor Sallai played a leading role in the radical re-organization of the Hungarian telecommunication industry and contributed to the process of convergence of telecommunications and information technology. At present, Professor Sallai is the Head of the Department of Telecommunications and Media Informatics of Budapest University of Technology and Economics, where he is also Vice Rector for Strategy. He is also President of the Telecommunication Systems Committee of the Hungarian Academy of Sciences.



Adding high availability features to server applications using aspect oriented programming

RÓBERT FAJTA

Nokia Research Center, Budapest, rfajta@gmail.com

PÉTER DOMOKOS, ISTVÁN MAJZIK

Budapest University of Technology and Economics, Dept. of Measurement and Information Systems
{pdomokos, majzik}@mit.bme.hu

Keywords: AOP, fault tolerance, legacy software, FT-scheme, AspectJ, HTTP server

The transformation of existing software to a fault tolerant one typically requires redesign and heavy modifications in the original source code. Aspect-oriented programming (AOP) is an emerging programming paradigm that promotes collecting features that are not related to the business logic, into crosscutting concerns, thus separates them from the original problem domain of the software. We analyzed how to use AOP to add fault tolerance to existing software by organizing the software into recovery blocks or N-version programming fault tolerance scheme. We gathered practical experiences by modification of a complex application software. (In: —)

1. Introduction

Software fault tolerance (FT) techniques are designed to allow a system to tolerate software faults that remain in the system after its development. An important class of software fault tolerance techniques utilizes independently developed but functionally equivalent *software variants* (often called versions). Independent design and implementation of variants is aimed at reducing the probability that more variants contain the same error(s), since the variants will act as redundant components masking or replacing each other in case of an error is activated on a given input. Recovery blocks (RB) [1] and N-version programming (NVP) [2] are well-known software FT schemes. They are used in critical applications ranging from embedded control systems to availability- or security-critical server applications in carrier grade products.

- The RB scheme implements a passive redundancy approach. It contains at least two variants of the same functionality, an acceptance test and an executive. Receiving a request, the executive establishes a checkpoint by saving the state of the application, then executes the primary variant and invokes the acceptance test on the result of the variant. If the result is acceptable then this will form the output of the FT scheme. In case of a failure reported by the acceptance check, recovery (restoration of state) is performed, and the next variant is invoked. Variants are called in this way until a valid result is found. If no variant can provide acceptable result then a failure is reported.

- The NVP scheme implements an active redundancy approach. It contains at least three variants for the same functionality, a voter and an executive. The executive invokes all the variants in a parallel way forwarding to them the input data, then collects the output results and forwards them to the voter. If a majority result exists then it will be the result of the NVP scheme, otherwise a failure is reported. Accordingly, the majority of fault-free variants mask the output of an erroneous one.

There are several extra requirements towards the software that is to be used as a variant in an FT scheme. The communication with the environment, the use of global variables, the implementation of error handling etc. are restricted since *all input and output shall be strictly controlled by the executive* as presented above. Accordingly, variants shall be implemented by taking specific rules into account.

If a service implemented by an existing software (called in the following as *legacy software*) is to be transformed to a fault tolerant one then a natural idea is to *use the existing software implementation (or parts of it) as a variant in the FT scheme*. (Besides this, another variant(s) have to be implemented as well.) Unfortunately, legacy software typically does not satisfy the requirements mentioned above, thus *re-design or modification* become necessary. These modifications usually mix up the concerns of the original functionality (business logic) and fault tolerance, as source code snippets for error detection, fault handling and redundancy management have to be modified or inserted into the original code.

When transforming an existing service to a fault tolerant one, we face the following challenges:

- *Establishing rules to form a variant and modifying the legacy software to satisfy these rules*. To do this, a technique was needed that allows performing these modifications in such a way that is easy to review, verify and change if necessary. Direct modification of the source code was not feasible from this point of view, as clear *separation of the original business logic and the FT extensions* was needed.

- *Elaborating the core logic of the FT scheme* in such a way that is easy to re-use if another service is to be transformed to a fault tolerant one.

There are several approaches that aim at separating functional and non-functional (e.g. fault tolerance) aspects. Library calls to pre-defined mechanisms [3], reflection [4], and meta-object protocols [5] are mature and well-tried techniques that address the separation of func-

tional and dependability requirements. We opted for the emerging paradigm of *aspect-oriented programming* [6] due to the following reasons:

- AOP provides a clear separation of non-functional activities (like redundancy management) by supporting the *modularized implementation* of the crosscutting concerns. In case of library calls, for example, the non-functional activities can be collected in a library, however, the calls to library functions are scattered in the original code.
- AOP has better support (considering programming languages, development and debugging environments) than reflection and meta-object protocols in our application environment.
- Moreover, the previous techniques do not allow fine grade parameterization of the modifications, e.g. by supporting name-based, property-based, location- or caller-specific modifications.

There are several approaches to AOP like the Multi-dimensional Separation of Concerns [7] supported by Hyper/J or the use of Composition Filters [8]. We follow the concepts and terminology of *AspectJ* [9] since in our case there is no need to handle multiple decompositions and the composability of aspects; only a modularized specification of the scattered behavior is needed.

The facilities of AspectJ AOP can be summarized as follows (the interested reader is referred to an introduction published in this journal [10]). The solution for fault tolerance as a crosscutting concern is provided by code snippets (*advices* in AOP terminology) that affect the behavior of the original software during execution.

These advices are applied at specific locations of the source code (called *join points*) as designated by AOP language expressions called *pointcuts*. Pointcuts can refer to call or execution of given methods, set or get of attributes etc. A *before* advice (*after* advice) may be executed before (after, respectively) a specified execution point (e.g. method call). An *around* advice may replace the original code at the join point. (Note that the original code can be executed by using a *proceed* statement in the *around* advice.) In this way, both the *extension of the behavior* and the *replacement or masking* of the original behavior is possible. *Aspects* modularize the pointcuts and advices, and can add members to existing classes/interfaces, as well.

For example, in *Figure 1* the call of the original *service1()* method (on the left) is designated by the *callService1* pointcut (on the right of the Figure). This pointcut is used in an *around* type of advice that replaces in run time the *service1()* method call with a different piece of code including a call to *otherService()*. The pointcut and the advice are modularized in an aspect called *MaskService*.

AspectJ as aspect language enables the entire feature set of the Java language and its libraries, while it adds the new language constructs. Its compiler (weaver) merges the aspects with the original code and generates ordinary Java class files. We examined the implementation of RB and NVP schemes on the basis of legacy software by using AspectJ AOP. After summarizing the general rules to form an FT variant (Section 2) our paper discusses the following:

- *Advantages and limitations of AOP to modify legacy software* to form a variant for an FT scheme (Section 3). AOP is used to insert additional functionality that is needed, and to replace existing behavior (code snippets) that are not allowed in order to satisfy the rules concerning the setup of FT variants. By using AspectJ AOP, the majority of the necessary modifications can be performed. However, it turned out that not all modifications can be implemented by applying the existing language constructs of AspectJ. In these specific cases the direct modification cannot be avoided.

- *Implementation of the core control logic of the FT scheme in a re-usable form* by using AOP (Section 4). Here the language constructs of the AOP are used to integrate the control logic (as a separate aspect) with the business logic of the application.

These results and experiences were gathered in a project in which a real-life application software, an HTTP service was transformed to an FT one. The details of this application are presented in Section 5.

2. Rules to be satisfied by a variant

The executive of the FT scheme shall effectively manage both the calling of the variant and the change of the global state outside of the variant. To be able to do

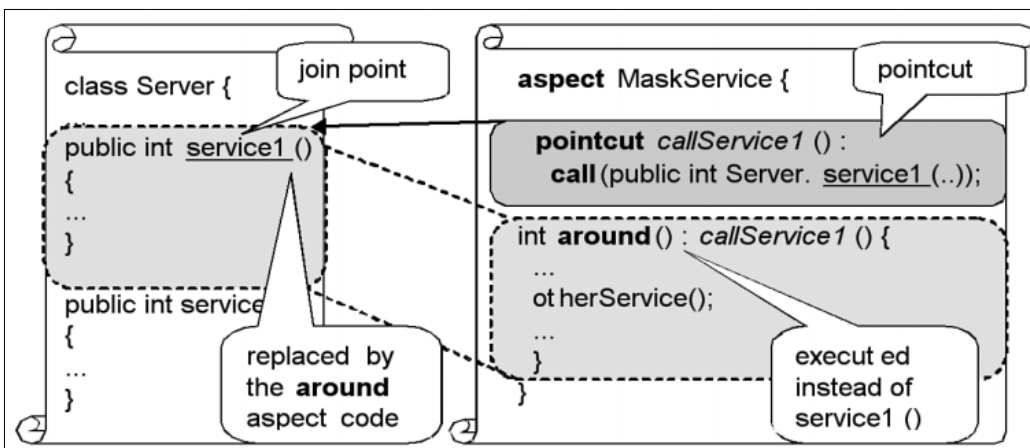


Figure 1. Masking a method by using AOP

so, the functionality provided by a variant shall be available through an external interface in the form of a set of distinct methods. The service of the variant shall be provided by the return values of these methods. Moreover, the following side effects are not allowed:

- changing the global state of the application by modifying global (shared) variables,
- sending or receiving messages,
- using volatile or non-deterministic values,
- performing individual error handling (besides returning error codes or throwing exceptions).

The following section presents how these rules can be satisfied by applying AspectJ aspects to the legacy Java code. Naturally, the identification of the locations where these rules might be violated requires the availability of the program source code (the utilization of bytecode modifications is left for future work).

Two restrictions can be derived even before a detailed analysis:

- If the service to be made FT is not available by a set of methods but spreads across the software then the direct code modification cannot be avoided.
- AspectJ pointcuts are based on attribute operations and methods, and finer level join points cannot be specified. Accordingly, no modification is possible at the level of instructions and control-flow operations. If a rule is violated at this level then the AOP based modification is not possible.

3. Forming a variant from the legacy code

The violations of the rules presented in Section 2 can be handled by AOP in the following cases:

Changing the global state. There are two alternatives to avoid the potentially inconsistent change of global variables by the variants:

- *Emulation of a separate environment* for each variant, which contains a copy of the global variables that the variants would modify. Each modification of the original

variables made in the variants must be redirected to the emulated variables by AOP advices. *Set* and *get* pointcuts referring to the original variables should be defined where a variant attempts to access the original variables. *Call* pointcuts should be used to pick out those points of the variant from where the original variables are accessed via internal setter or getter methods. Additionally, an *around* advice should be attached to these pointcuts to redirect the write and read operations to the emulated variable set. At the end of the successful execution, the emulated global variables of a variant that provided proper result must be copied to the original global variables.

- *Splitting up the variants into functionally equivalent blocks* that do not modify global variables, and performing a local adjudication process among the blocks before executing the modifications (Figure 2). However, this splitting is not always possible, e.g. if the variants access the global variables in different order.

Sending and receiving messages. The legacy software may establish a connection to receive or send messages. These messages get out of control of the FT executive (e.g. by changing the state of the receiver).

The proposed solution is using aspects to redirect message readings and writings from/to the communication channel to buffers. Incoming messages have to be stored in these buffers by the executive and variants can receive messages from the buffer. The outgoing messages are written to the buffer and they are forwarded to the original target by the executive after the successful execution of the variants.

A possible realization with AspectJ is the definition of *call* pointcuts to pick out the method where the variant attempts to retrieve the input and output streams of the communication channel. *Around* advices are attached to these pointcuts to create the buffers in the advice and attach piped input and output streams to them, respectively. A thread is created to store the messages from the communication channel to the input buffer, and another thread is used to replay the contents of the buffer to the variant. The advices return with these piped

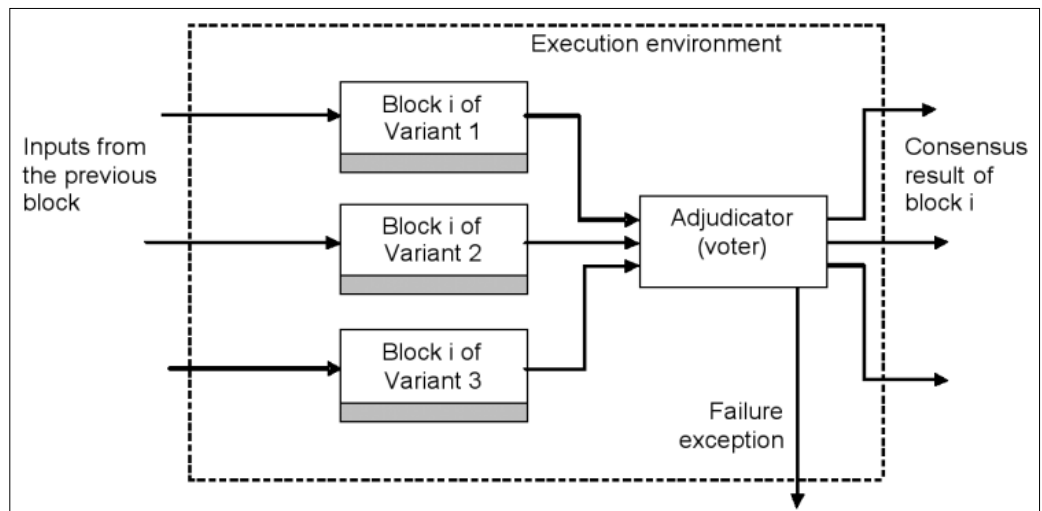


Figure 2. Local adjudication among blocks in an NVP pattern

streams. (Threads are necessary because piped input and output streams attached to each other must be handled in separate threads in order to avoid deadlock.)

If the original software expects an answer to a message, then the variants must be split into blocks (if possible) in which no message is sent/received. The executive applies local acceptance tests (in the case of RB) or voting (in the case of NVP, Figure 2) and then sends the messages and waits for the answer to be processed in the next block.

Error handling including exceptions. The legacy software may contain error handling code. This error handler shall be masked only if it prevents the proper execution of the FT scheme. E.g. an error handler that aborts the execution of the variant by returning null value or throwing an exception to indicate an error should not be modified. Instead, the executive can examine the return value or catch the exception to recognize that the variant failed. However, if the error handler takes actions that have effects outside of the variant (e.g. it sends messages or terminates the program execution), then the error handler must be masked by an advice as follows:

- If the error handler is located in a *distinct method*, then this method can be suppressed by picking it out using an `execution` pointcut to which an empty `around` advice is attached.
- If the error handler in the legacy code is realized as an *exception handler*, then a possible solution is attaching an `around` advice on each call that can throw an exception caught by this exception handler, and surrounding the `proceed` call in this advice by a *try-catch* block. Thus the exception does not reach the original exception handler.

4. Implementing the FT scheme

Conceptually, the FT executive and the wrapper code snippets that are applied to form a variant from the legacy code are implemented in aspects.

The executive of the FT scheme will represent the services of the legacy software (now forming one of the variants) towards the clients, i.e., the calls to the original legacy methods shall be redirected to the FT executive. To do this, these method executions are picked out by `execution` pointcuts that reveal their parameters. An `around` advice is attached to these pointcuts to mask the direct invocation of the original method and execute the control logic of the FT scheme instead. The original method is invoked from this advice by the `proceed` statement of AspectJ and the parameters are passed to it. If necessary, other variants and the adjudicator (voter or acceptance test) are called as well. The setup is presented in Figure 3.

In the following some specific problems of the implementation are summarized:

Timeout checking. If timeout checking of the call to the original method is implemented in the client then the executive shall translate this timeout value to timeout values for the individual variants that are executed and it shall check their timeliness.

Termination support. The variants shall implement a cooperative mechanism to allow stopping them in case of timeout, since Java does not support forced thread stopping (`Thread.stop()` became deprecated).

Reporting the failure of the FT scheme. If the failure of the variant(s) cannot be tolerated by the FT scheme then the executive has to report the failure to the caller according to the caller's expectation (e.g. by providing an error code or throwing an exception).

Checkpointing and recovery. In the case of the RB and the serially executed NVP schemes the legacy code must be analyzed to find out whether it modifies the state of its environment during execution. The state prior to modification must be saved and later restored.

Design of adjudicators. The acceptance test of the RB or the voter of the NVP shall not contain fault handling code (for example, exception throwing) since fault handling must be provided solely by the executive of the FT scheme. Existing fault handling code in a legacy adjudicator shall be masked by an AOP advice.

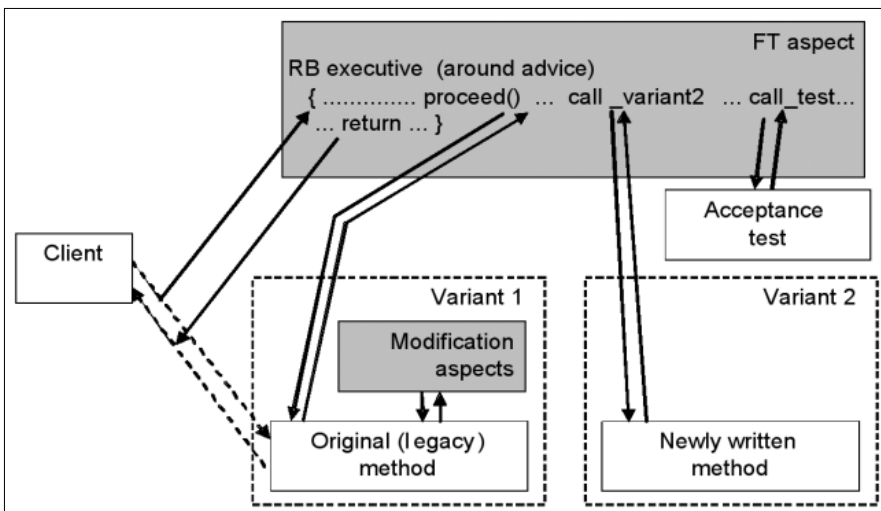


Figure 3. Implementation of the FT scheme

Re-use of the aspect-based modularization of the FT scheme is possible as follows:

- The core control logic implemented by the advice belonging to the executive is directly reusable.
- The application specific parts of the executive are formed by those functions which communicate with the client, call the original variant, handle checkpointing and recovery, and call the adjudicator. These parts must be adapted.
- Adjudicators are typically application-specific thus they cannot be re-used in original form.

The AOP based implementation of the FT scheme does not introduce significant extra cost besides the known costs of the FT solution (i.e., costs of developing additional variants and adjudicator, cost of increased execution time). Moreover, the same dependability bottlenecks are present as in the case of the traditional implementation: faults in the executive and the adjudicator shall be tolerated separately.

5. Implementation experiences

The concepts described in the previous sections were applied in the case of Sun Microsystems' Open Service Gateway Initiative (OSGi) [11] implementation, called Java Embedded Server (JES) that provides a framework for multiple applications (called bundles).

One of the applications is the HTTP Service. It can be used in web based servers as well as in mobile phones to provide extra services for configuration or download. The HTTP Service provides a registration service for other bundles in the framework to enable them to register their own resources and/or Java servlets. The HTTP Service acts as an HTTP server which analyzes the incoming request and maps it to a registered resource or servlet

and delivers the resource or the result of the servlet to the client.

Our task was to make this OSGi HTTP Service fault tolerant by applying the RB scheme. The original HTTP Service implementation (referred to as *jesHTTP*, see *Figure 3*) is considered as one variant, and another implementation (referred to as *nHTTP*) is developed as the second variant.

The fault tolerant HTTP service implementation (referred to as *ftHTTP*) forms a single bundle. After starting the service, it is ready to accept servlet and resource registrations and unregistrations, and client connections. As it uses *jesHTTP* as a variant, there are several functions that the *ftHTTP* must implement, and some others that have to be masked in *jesHTTP*. The latter are not parts of the HTTP service itself, but are responsible for starting up and shutting down the bundle. Moreover, servlet initialization and destruction were implemented in the FT executive and were masked in the *jesHTTP* variant. They belong to the servlet's life cycle and must be performed exactly once. Note that failures occurring in the servlets as external units are out of the scope of the RB scheme.

During the modification of *jesHTTP*, the problems of changing the global state, and sending and receiving messages had to be handled.

Changing the global state. The global state of the HTTP server is represented by the registration database, the session data and the servlet context. In our implementation the registration database and the session data are in the scope of the variants, and their consistency is to be ensured by the executive.

In this way the complex operations of handling the registration database and the session data are performed under control of the RB scheme.

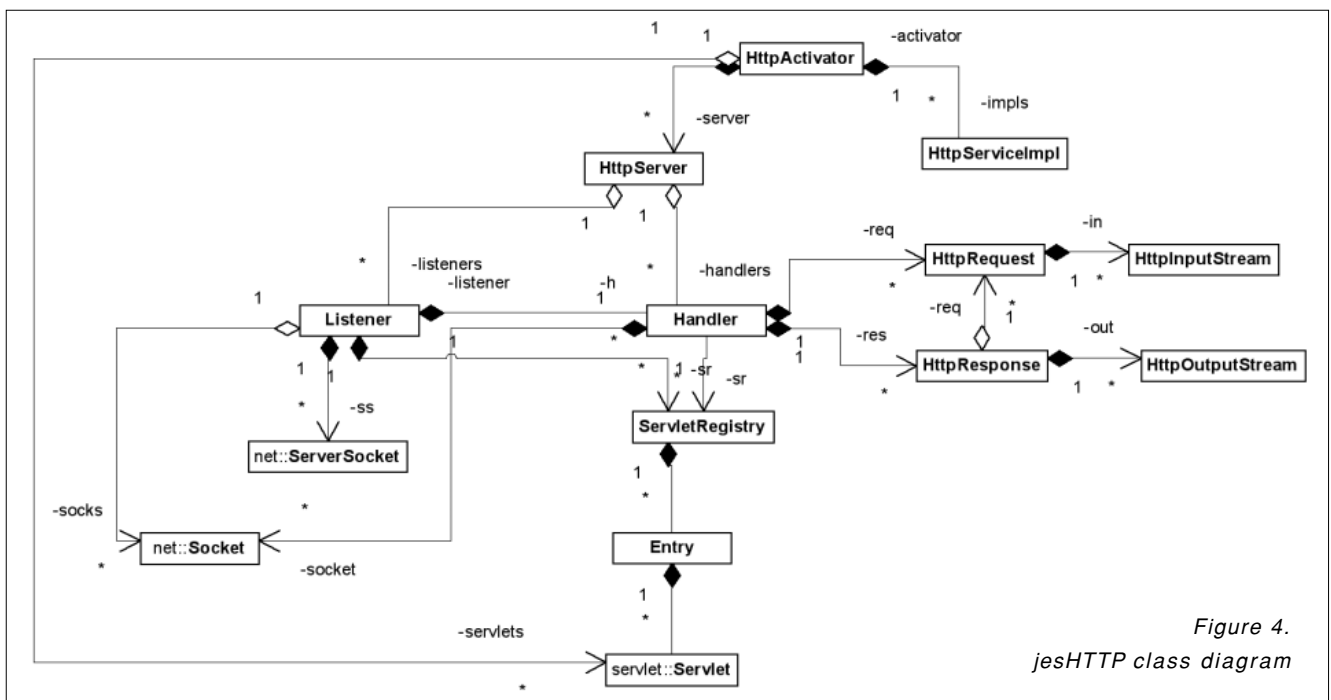


Figure 4. *jesHTTP* class diagram

Registration database: Each variant has to maintain its own registration database. Consistency is ensured by forwarding each registration related request to each variant (Figure 4). If a fault occurs in a variant, e.g. it removes the wrong entry upon an unregistration request, it will not be able to serve the requests regarding the removed entry, instead, it will report an error. However, the other variant may be able to serve the request. In the case of a common registration database this kind of fault tolerance could not be achieved. The drawback of this solution is that the registration of a servlet or a resource becomes slower since multiple registrations are performed. However, this is a rare event under normal operating conditions.

Session data: Sessions are used to store client-related data between requests. They are manipulated by servlets (session data can be used for communication between servlets). Session data must be available to the servlets independently from the currently running HTTP server variant. Before the execution of the first variant, the session data is saved in a checkpoint. If the first variant succeeds, the checkpoint is discarded. If the variant fails, the saved session data is reloaded into the next variant. This is the recovery step of the RB scheme. After the successful execution of this variant, its session data is used in the following. In fact, the session data is considered as global data that is passed to the variants upon execution (as in the case of the emulated environment).

Servlet context: It is part of the servlet configuration maintained by the HTTP server. If a servlet sets an attribute of the servlet context, it must be possible to retrieve this attribute later, even if the servlet is executed by another variant. The execution of the servlet is atomic from the viewpoint of the HTTP service. The implementation of the context management (setting and getting variables) is rather straightforward, therefore it was not put under control of the RB scheme. Although the

servlet context belongs to global data, it is not involved in the checkpointing, because it is not the variant but the servlet that manipulates the context data.

The ftHTTP initializes the servlet with its own servlet configuration implementation that can store references to other servlet configurations. An advice is used to suppress the servlet initialization in the jesHTTP variant, and store a reference to the jesHTTP servlet configuration in the ftHTTP implementation.

If a variant retrieves the servlet configuration, an advice captures this request and it returns the object that was stored in the ftHTTP servlet configuration for the corresponding variant.

Access to the attribute handling methods of the servlet context contained by the servlet configuration of the variant is redirected to the common implementation. This way, if a variant adds extra functionality to the servlet configuration or the servlet context, it can continue to use the extra functionality; while attributes that are part of the servlet context specification remain common between all variants. However, if a variant adds extra functionality to the servlet configuration or the servlet context, it must be analyzed whether it interferes with the services defined by the interface. If it is so, then this interference must be managed.

Sending and receiving messages. The process of the original servlet registration is modified by an advice to suppress creating and starting a listener that would accept connections on a server socket. In the FT implementation, connections are accepted in the ftHTTP executive. The first step of serving a request is storing the request in a buffer, so that it can be replayed to the variants.

The implementation applies advices that replace the input and output streams in the jesHTTP variant (when they are retrieved from the socket) with piped input and output streams. The piped input stream reads from the

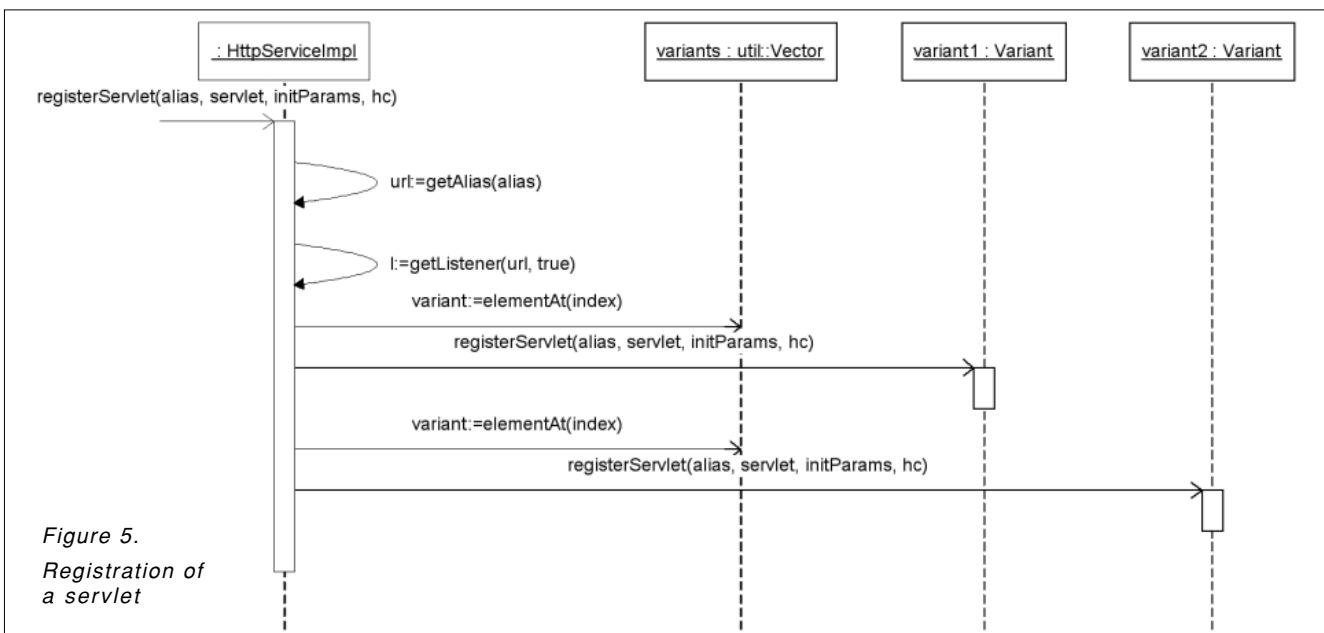


Figure 5. Registration of a servlet

request buffer, the piped output stream writes to the response buffer, thus the variant does not use the streams of the socket directly. To compose the response, the variant invokes the servlet or retrieves the resource. The response of a successfully executed variant will be committed to the socket.

Implementation of the acceptance test. The acceptance test of the RB scheme examines the value of the response code returned by the HTTP server variant. If the response is "OK" then the variant can commit its result to the socket, thus the browser will receive the response to its request. Otherwise, the executive invokes the next variant that will compose its own response. Since the second variant is the last one, the HTML page generated on the basis of the response of the second variant will be forwarded to the client independently whether it is an error page or a proper web page.

Summary of changes. The original source code was changed at 19 points, where the visibility of classes and fields had to be made *public* for referencing them from aspects. These changes could not be made from aspects.

All other necessary changes, namely 5 structural and 15 behavioral modifications, were carried out by aspects. Typically, the aim of these modifications was either to prevent the execution of an action (e.g. initialization or destruction of a servlet in the *jesHTTP* variant), or to modify the data source for an action (e.g. redirecting socket read and write operations to an extra buffer).

The following peculiarities of AOP (that are not fully supported in the other reflective approaches) were utilized:

- Advices can introduce new attributes in legacy classes. E.g. a reference to the common servlet context was added in *jesHTTP*.
- The execution of an advice may depend on the *caller* method. E.g. initialization of a new *ServerSocket* was masked only from the legacy software. Note that reflective extensions are typically applied from the viewpoint of the *called* method.
- It is possible to designate the *call of a method* in an AOP pointcut. In this way those methods could also be masked that are not available in source code form (e.g. *System.exit*). In the case of the *ServerSocket* initialization, this *call* pointcut was parameterized with the name of the caller method.

6. Conclusions

The advantages of the AOP approach manifest directly if legacy systems or parts of legacy systems have to be made fault tolerant. In this case the necessary changes can be made and the selected FT scheme can be applied in a *modularized form by aspects*. They can be han-

dled separately and implemented without drastically modifying the original source code.

In our paper we identified the conditions of applying AOP, discussed the problems that arise during the implementation, and proposed solutions. These results may be used to direct the attention of engineers to those points of the legacy code that has to be modified by AOP advices. In our future work we will investigate the insertion of aspects at Java byte-code level to implement the FT executive in case of commercial off-the-shelf (COTS) variants.

References

- [1] B. Randell, J. Xu:
The Evolution of the Recovery Block Concept.
In M. Lyu (ed.): *Software Fault Tolerance*.
John Wiley & Sons Ltd, pp.1–22., 1995.
- [2] A. Avizienis:
The Methodology of N-version Programming.
In M. Lyu (ed.): *Software Fault Tolerance*.
John Wiley & Sons Ltd., 1995.
- [3] K. J. Birman:
Replication and Fault Tolerance in the Isis System.
ACM Operating System Review, Vol. 19, No. 5,
pp.79–86., 1985.
- [4] G. Agha, S. Frolund, R. Panwar, D. Sturman:
A Linguistic Framework for Dynamic Composition of
Dependability Protocols. In *DCCA-3*,
pp.197–207., 1993.
- [5] J.-C. Fabre, V. Nicomette, T. Pérennou,
R. J. Stroud, Z. Wu:
Implementing Fault Tolerant Applications using
Reflective Object-Oriented Programming.
In *Proc. FTCS-25*, pp.489–498., 1995.
- [6] G. Kiczales et. al.:
Aspect-Oriented Programming.
In *Proc. European Conf. on Object-Oriented
Programming (ECOOP)*. LNCS 1241,
Springer Verlag, 1997.
- [7] H. Ossher, P. Tarr:
Using Multidimensional Separation of Concerns to
(Re)Shape Evolving Software. *Comm. of the ACM*,
(44)10, pp.43–50., 2001.
- [8] L. Bergmans, M. Aksit:
Composing Crosscutting Concerns Using
Composition Filters. *Comm. of the ACM*, (44)10,
pp.51–57., 2001.
- [9] I. Kiselev:
Aspect-Oriented Programming with AspectJ,
Sams Publishing, 2003.
- [10] L. Lengyel, T. Levendovszky:
Introduction to Aspect-Oriented Programming.
Híradástechnika, Vol. LX, 2005/6, pp.18–23.
- [11] Open Service Gateway Initiative,
<http://www.osgi.org>

Call for Papers

16th IST Mobile and Wireless Communications Summit



16th IST MOBILE & WIRELESS
COMMUNICATIONS
SUMMIT
BUDAPEST, HUNGARY
1-5 JULY 2007



Topics of interest include but are not limited to the following:

Physical

- Problems in Coding & Modulation
- Multiuser Detection
- MIMO and Space-Time Techniques
- Adaptive Coding & Modulation
- Capacity Issues

Access

- Radio Access Methods
- Spectrum Management
- Spectrum Sharing & Coexistence
- Radio Resource Management
- Call Admission Control
- QoS & Scheduling
- Service Access & Mobility
- Multiple Access Techniques
- Mobile, Fixed & Nomadic Wireless Access

Networks

- Cross-Layer Techniques & Optimization
- Broadband & Fixed Wireless Networks
- UWB Techniques
- Cellular Networks
- Ad-hoc Networks
- Future Broadband Wireless Communication
- Mesh Networks
- Sensor Networks
- Relaying & User Cooperation
- Fixed & Mobile Convergence
- Wireless LAN and PAN
- Wireless and Mobile IP
- Heterogeneous Wireless Networking
- End-to-End QoS Provision

Techniques and Technologies

- Channel Estimation
- Propagation & Channel Modelling
- Fading Mitigation and Diversity
- Cognitive/Software Radio
- RF Components & Radio Electronics
- Antennas Including MIMO Antennas
- Optical Techniques in Wireless Systems
- Cell & Capacity Planning
- Novel Wireless Switching & Routing Techniques
- Mobility Management
- Broadcast & Multicast Techniques
- Traffic Control & Engineering

Applications

- Applications & Business Models
- Biological & Environmental Aspects
- End-to-End Reconfiguration
- Impact on Society
- Location Based Services and Positioning
- Security & Privacy
- Strategies, Policies & Regulation

System descriptions

- Terrestrial Systems
- Satellite Systems
- High Altitude Platforms Systems
- Broadband Wireless Systems
- Beyond 3G & 4G Systems
- Experimental Systems and Field Trials
- Modelling, Analysis & Simulation of Mobile and Wireless Systems

Contacts

General Chair:
Executive Vice Chair:
Summit Secretariat:

István Frigyes
János Bitó
Ms Mária Tézsla
HTE – Scientific Association for Infocommunications
Kossuth Lajos tér 6-8, Budapest
H-1055, HUNGARY
+36 1 353 1027
+36 1 353 0451
www.mobilesummit2007.org
info@mobilesummit2007.org
tutorial@mobilesummit2007.org
workshop@mobilesummit2007.org
exhibition@mobilesummit2007.org

Address:

Phone:

Fax:

Web:

Email:

Patrons and Technical Co-Sponsors



Gold Patrons

Vodafone
Cisco



Bronze Patrons

Ericsson
Totaltel



Technical Co-Sponsors

Information Technology Society (ITG)
Information Technology Society within VDE

IEEE

IEEE Communications Society

IEICE

1-5 July 2007, Budapest, Hungary,

www.mobilesummit2007.org

Invitation

The Department of Broadband Infocommunications and Electromagnetic Theory of Budapest University of Technology and Economics, and the Scientific Association for Infocommunications, Hungary (HTE) cordially invite you to participate in the 16th IST Mobile and Wireless Communications Summit that will be held 1-5 July, 2007 at the University Congress Centre of Eötvös Loránd University, Budapest, Hungary. The 16th IST Mobile and Wireless Communications Summit is a major conference organized annually in Europe, sponsored by the European Commission. In 2007 it will be held in Budapest, capital of Hungary. It is the first time that the Summit goes to a Central-European country, new member state of the European Union.



General information

One of the aims of the Summit is to give a report on the progress achieved in European research projects in the field of mobile and wireless communications in Information Society Technologies. However, being an open call conference it is by no means limited to these projects. A second aim is to give a forum for researchers, service providers and regulators from all over the world to present their results and to exchange their views. Besides of technical results and progress, techno-economic, regulatory and security issues form also part of Summit's topic. Authors are invited to submit high-quality full papers representing original results in all areas of wireless communications systems and networks, mobile and fixed, terrestrial, satellite and HAP. Fully reviewed papers accepted by the TPC are either for oral or for poster presentation. However, this decision is based on what is more appropriate for the very paper and by no means on its quality; papers accepted as posters are not at all of "lower rank".



Organisers



Budapest University of Technology and Economics



Scientific Association for Infocommunications



European Union Sixth Framework Programme for Research and Development



Information Society Technologies

Paper Submission Guidelines

Manuscripts should clearly describe the novelty and usefulness of the presented results and should be distinct from the authors' previous results. Prospective authors are encouraged to submit full paper, not more than five pages, including graphs, tables and figures using the IEEE template (www.ieee.org/web/publications/authors/transjnl/index.html) in pdf format through the Summit web site (www.mobilesummit2007.org). The author should indicate if he/she prefers oral or poster presentation.

Important dates

Complete Paper Manuscripts Due: **15 January 2007**
 Notification of Acceptance: **31 March 2007**
 Camera Ready Manuscripts Due: **15 May 2007**

Call for Tutorials

The 16th IST Mobile and Wireless Communication Summit invite proposals for tutorials. 3-hour tutorials will be held on the first day of the Summit (1 July 2007). Recognized experts in the relevant area will report on the state of the art in distinct fields. Potential speakers are invited to submit their proposal through the Summit web site (www.mobilesummit2007.org).

Important dates

Tutorial Proposal Submission Deadline: **31 January 2007**
 Notification of Acceptance: **15 February 2007**
 Deadline for Submitting Handouts: **15 May 2007**

Call for Workshops

The 16th IST Mobile and Wireless Communications Summit invite proposals for Workshops. Half day or full day workshops will be held on the last day of the Summit (5 July 2007). Subject of the workshop should be related to any IST project or to other topics relevant to mobile and wireless communications. Workshops are related loosely to the Summit only and are mainly in the responsibility of the organizer. Potential Workshop organizers are invited to submit their proposal through the Summit web site (www.mobilesummit2007.org).

Important dates

Submission of Workshop Proposal: **28 February 2007**
 Notification of Acceptance: **31 March 2007**
 Submission of the Final Program: **15 May 2007**

Exhibition

A scientific, technical and industrial exhibition will be held during 16th IST Mobile and Wireless Communications Summit. The exhibition will run in parallel with the Summit from 1-5 July 2007. It will be held in the same place as the Summit, at the University Congress Centre of Eötvös Loránd University. Constructed and plain exhibition areas are available. Minimum exhibition area to be ordered: 6 sqm. Please find further information on the Summit web site (www.mobilesummit2007.org). If you intend to participate as exhibitor in the event you are kindly asked to contact Exhibition Organiser (exhibition@mobilesummit2007.org).

Important dates

Booth Registration Early Deadline: **2 April 2007**
 Booth Registration Deadline: **31 May 2007**

16th IST Mobile and Wireless Communications Summit