IST Programme

Project Number: IST-2001-36006

Acronym: MobiHealth

A project funded by the European Community under the “Information Society Technologies” Programme (1998-2002)

Overall evaluation of the GPRS and UMTS infrastructure for the new services

(D5.2)

Editors: Dimitri Konstantas

Authors: Richard Buls, Katarzyna Wac, Val Jones, Jens Eliasson, George Koprinkov, Aart van Halteren, Nikolai Dokovsky, Dimitri Konstantas

Preparation Date: April 29, 2004

Version: 1.2

Contract Start Date: May 01, 2002

Duration: 22 months, (February 29, 2004)

Project co-ordinator: Rainer Herzog, Ericsson GmbH
# TABLE OF CONTENTS

TABLE OF CONTENTS ................................................................................................................. 2  
Executive Summary ...................................................................................................................... 4  
List of abbreviations and definitions .......................................................................................... 5  
1  Open questions that need a reply ........................................................................................... 8  
2  Introduction to the rational and context .................................................................................. 9  
   2.1  Context ............................................................................................................................ 10  
   2.2  Research Questions ......................................................................................................... 11  
   2.3  Approach ....................................................................................................................... 11  
3  Transport System Performance Evaluation ............................................................................ 13  
   3.1  Measurement Methodology ............................................................................................ 13  
   3.2  Modelling Methodology ................................................................................................. 16  
4  Transport System Services Modelling ................................................................................... 18  
   4.1  Objectives ....................................................................................................................... 19  
   4.2  System Description and Boundaries .............................................................................. 20  
      4.2.1  System Decomposition .......................................................................................... 21  
      4.2.2  System under Test .................................................................................................. 22  
   4.3  Service Description ......................................................................................................... 24  
      4.3.1  Service Decomposition .......................................................................................... 24  
      4.3.2  Service Characteristics Analysis ............................................................................ 26  
   4.4  Evaluation System .......................................................................................................... 27  
   4.5  Performance Criteria of Interest .................................................................................... 32  
      4.5.1  ITU-T 3 x 3 matrix approach ............................................................................... 33  
      4.5.2  Selection of Performance Criteria and Parameters ........................................... 35  
   4.6  System Parameters of Influence .................................................................................... 37  
   4.7  Workload and System Parameters ................................................................................. 40  
5  Performance Measurements ................................................................................................. 47  
   5.1  Measurements Instrumentation ....................................................................................... 48  
      5.1.1  SoD and SUT instances ......................................................................................... 48  
      5.1.2  Evaluation System Instrumentation ........................................................................ 55  
      5.1.3  Evaluation System Synchronization ....................................................................... 73  
   5.2  Measurements Setup and Execution ............................................................................... 78  
      5.2.1  Setup ...................................................................................................................... 79  
      5.2.2  Execution ............................................................................................................... 88  
      5.2.3  Results of the EDR data analysis .......................................................................... 99  
   5.3  Measurements Evaluation ............................................................................................... 109  
6  Performance Modelling and Evaluation .............................................................................. 117  
   6.1  Transport System Model ................................................................................................. 118  
   6.2  Delay Analysis .............................................................................................................. 126  
      6.2.1  Conclusive Graphs .................................................................................................. 126  
      6.2.2  Derivation of Conclusive Graphs ........................................................................... 128  
      6.2.3  Quantitative Analysis ............................................................................................ 131  
      6.2.4  Influence of System Parameters .......................................................................... 132  
   6.3  Bottleneck Analysis ....................................................................................................... 143  
      6.3.1  Conclusive Graphs .................................................................................................. 144  
      6.3.2  Derivation of Conclusive Graphs .......................................................................... 146
6.3.3 Quantitative Analysis ................................................................. 147
6.3.4 Influence of System Parameters .................................................. 149
6.4 Scalability Analysis ........................................................................... 156
  6.4.1 Conclusive Graphs ..................................................................... 157
  6.4.2 Derivation of Conclusive Graphs ............................................... 161
  6.4.3 Quantitative Analysis ................................................................. 162
7 Digest of results .................................................................................. 167
8 Recommendations and overall conclusions ........................................... 172
  8.1 Hospitals ......................................................................................... 172
  8.2 Operators ......................................................................................... 173
  8.3 Technology suppliers ....................................................................... 175
References ............................................................................................ 177
Appendix 1 System Parameters of Influence ....................................... 178
Appendix 2 V3GNL Leased line infrastructure ...................................... 178
Executive Summary

This document reports on the technical evaluation of the GPRS and UMTS networks and their capability to support the mobile health applications developed in the MobiHealth project. The starting point of this report is the questions raised during the user evaluation (“Open Questions” in Deliverable 5.1) and more specifically the communication related questions. We first describe the evaluation methodology and the theoretical background and assumptions. Next we present the model developed for the theoretical analysis of the network and the performance measurements taken during project trials. Finally we present the evaluation results and our conclusions and recommendations.

The developed model provides an abstraction of the overall system (in terms of communications), decomposing it into four different communication parts. The first being the Intra-Ban communication network (bluetooth), next the public operator network (UMTS), then the internet backbone (fiber,cable …) and finally the enterprise intranet network. In mobile health applications what is finally important is the end-to-end performance of the communication system. However from the 4 network parts we have very good knowledge of the behaviour of the 2 of them (internet backbone and enterprise intranet) while we have full control of the Intra-Ban network. Thus the only unknown factor is the public operator’s network : the UMTS network. For the performance evaluation we considered different parts of the network chain as black-boxes and we concentrated only on the parts that are of interest, that is the UMTS network.

Our measurements were taken into 2 different situations: during the life trials (passive measurements) and during the communication tests (active measurements). The passive measurements provide us with information of the network behaviour in real life usage, but do not provide us with detailed measurements of the communication parameters (like delay and bandwidth). The active measurements replicate the end-user system behaviour, in terms of data production, but under a fully controlled environment where we can measure communication parameters. The active measurements were taken using different packets sizes, at different hours of the day, under different loads and different profiles. The network behaviour was recorded and we have extracted the UMTS network performance under different conditions.

The main results regarding the support capabilities of the UMTS network for mobile health applications can be summarized in the following:

- The UMTS network is stable and can be used for (limited) wireless medical services
- Several parameters defining the behaviour of the UMTS network need to be clarified by the operators, so that the applications can be fine tuned
- More uplink bandwidth is needed for medical services (reversed consumer-producer problem)
- The hardware manufactures of terminals (PCMCI cards, UMTS telephones) must provide programming interfaces (APIs) giving access to network parameters and control
- Communication standards for medical applications need to be defined.

NB: This work is part of the Master Thesis of Richard Buls and Katryna Wac, done at the University of Twente, and which will be present for defence in summer 2004.
### List of abbreviations and definitions

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>Access Point Network Identifies a Packet Data Network (PDN) that is accessible from a GGSN node in a GPRS/UMTS network [TechWeb].</td>
</tr>
<tr>
<td>BAN</td>
<td>Body Area Network “Collection of (inter) communicating devices, which are worn on the body, providing an integrated set of personalized services to the user” [Val Jones and Richard Bults, WWRF BoV 2001]</td>
</tr>
<tr>
<td>BANip</td>
<td>BAN interconnect protocol Protocol for interaction between the MBU surrogate (object) and the MBU device [Doko2003].</td>
</tr>
<tr>
<td>CUS</td>
<td>Component Under Study A specific part of the SUT of which the influence on the quantitative aspects of the SUT is studied [Hoek1997].</td>
</tr>
<tr>
<td>E1</td>
<td>The European counterpart to T1, which transmits at 2.048 Mbits/sec [TechWeb].</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service An enhancement to the GSM mobile communications system that supports data packets. GPRS enables continuous flows of IP data packets over the system.</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol The network layer protocol in the TCP/IP communications protocol suite. IP contains a network address and allows messages to be routed to a different network or subnet. IP does not ensure delivery of a complete message.</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider An organization that provides access to the Internet [TechWeb].</td>
</tr>
<tr>
<td>ITU-T</td>
<td>International Telecommunication Union – T</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine A Java interpreter. The Java Virtual Machine (JVM) is software that converts the Java intermediate language (bytecode) into machine language and executes it.</td>
</tr>
<tr>
<td>Kbps / KBps</td>
<td>Kilo Bits Per Second / Kilo Bytes Per Second One thousand twenty four (1024) bits per second. Upper case “B” in KBps means kilobytes per second.</td>
</tr>
<tr>
<td>MBU</td>
<td>Mobile Base Unit</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum Transmission Unit (Maximum Transfer Unit) The largest frame size that can be transmitted over the network [TechWeb].</td>
</tr>
<tr>
<td>NTP</td>
<td>Network Time Protocol A protocol used to synchronize the realtime clock in a computer [TechWeb].</td>
</tr>
<tr>
<td>PCMCIA</td>
<td>Personal Computer Memory Card International Association Standard for connecting peripherals to portable computers [TechWeb].</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit The technical name of a frame of data transmitted over the data link layer (layer 2) in a communications network [TechWeb].</td>
</tr>
<tr>
<td>PPP</td>
<td>Point-to-Point Protocol The most popular method for transporting IP packets over a serial link between the user and the ISP.</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>SAP</td>
<td>Service Access Point</td>
</tr>
<tr>
<td>SE</td>
<td>Service Element</td>
</tr>
<tr>
<td>SoD</td>
<td>System of Discourse</td>
</tr>
<tr>
<td>SP</td>
<td>Service Primitive</td>
</tr>
<tr>
<td>SUT</td>
<td>System Under Test</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transmission Control Protocol / Internet Protocol</td>
</tr>
<tr>
<td>TOS</td>
<td>Type Of Service</td>
</tr>
<tr>
<td>TTL</td>
<td>Time To Live</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunications System</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td>V3G</td>
<td>Vodafone 3G Netherlands</td>
</tr>
<tr>
<td>VTT3G</td>
<td>Vodafone, Telia, Telefonica, 3G networks</td>
</tr>
</tbody>
</table>
1 Open questions that need a reply

In deliverable 5.1 the users provided us with a series of open questions concerning different parts of the system. These questions can be categorized as questions concerning the User Interface, functional characteristics, communication issues etc. While the questions related to user interface, functionality of the system and ergonomics provide the directions for refinement of the overall system and the improvement of the system, the communication questions require a further in-depth analysis of the communication infrastructure (UMTS, GPRS). In addition some questions coming from the users, although they seem to be simple functionality questions they are just the tip of the communications infrastructure iceberg. For example, due to limited bandwidth the buffers of the MBU were overflown resulting in system shutdown or blocking. This from the user point view is just a “software problem/bug” while in reality it is due to bandwidth shortage. Nevertheless, in order to verify where is the real problem in the overall MobiHealth service, a detailed and in-depth analysis of the network was performed.

In addition to the communications questions posed by the users:

- Why did the BAN sometimes break off contact by itself?
- Why has the equipment broken off contact while traveling between visits? Coverage?
- Why during startup in UMTS was no contact established between the iPAQ and the phone, although batteries had just been changed?

A number of more technical questions were raised in for the communication infrastructure evaluation, and namely:

- What is the bandwidth available in the UMTS network and what are the communication delays?
- What is the behaviour of the UMTS network at heavy load?
- What is the behaviour of the UMTS terminals available in the market? What is missing?

All the above questions provide us the starting point for the evaluation of the UMTS network. For the evaluation of the UMTS network we used data coming from the Vodafone (NL), Telia (S) and Telefonica (E) UMTS pre-commercial networks. Here we must note that in all the networks we were running under best case situation, since the UMTS network was empty or almost empty (in the Netherlands for example we were the only users of the UMTS network in the Twente region). Nevertheless the results obtained provide a very good indication on the problems and issues that need to be resolved for the development of robust medical services.

In this report we present a generic methodology for performance evaluation of transport systems that support the MobiHealth BANip (Body Area Network interconnect protocol). A performance methodology is outlined to perform measurements of a selected transport system. The obtained measurements are than used to derive a high level performance model of the service delivered by this transport system. Both measurements and model are used for performance evaluation of the selected transport system, e.g. delay analysis, bottleneck analysis and scalability analysis.
2 Introduction to the rational and context

Patient monitoring outside a hospital is practically non-existing today. If a patient requires measurements of various vital signs (e.g. blood pressure, heart rate, blood glucose level) at regular intervals, this patient has to visit the hospital frequently. Even patients who are not at immediate risk are obliged to visit the hospital and stay there for some time to undergo a series of regular measurements. This may result in high costs for the hospital and healthcare insurers and loss of work hours and morale of the patient.

Remote monitoring of patients may be a solution to reduce health care costs, increase productivity of working patients and increase the quality of life of chronically ill patients. We use the term “m-health” in this context, but what is m-health? The answer of this question lies in the understanding of e-health. Immediately the question rises: What is e-health? The term e-health (abbreviation of electronic health) is commonly used by people, but it’s difficult to find a clear generally accepted definition for this comparatively new term. A few years ago this term was barely used and now seems to serve as a general "buzzword," used to characterize not only "Internet medicine", but also virtually everything related to computers and medicine. An Internet survey delivers many descriptions of e-health but only a few useful definitions. One description for example is provided by one of the leading Internet technology providers (Intel), they refer to e-health as "a concerted effort undertaken by leaders in health care and hi-tech industries to fully harness the benefits available through convergence of the Internet and health care". It is clear that the scope of e-health is broader than the deployment of Internet related (ICT) technology in the healthcare domain. The definition presented by the Journal of Medical Internet Research applies to the broader scope:

E-health is an emerging field in the intersection of medical informatics, public health and business, referring to health services and information delivered or enhanced through the Internet and related technologies. In a broader sense, the term characterizes not only a technical development, but also a state-of-mind, a way of thinking, an attitude, and a commitment for networked, global thinking, to improve health care locally, regionally, and worldwide by using information and communication technology [Eyse2001].

The utilization of different modern information and (tele) communication technologies (ICT) will play a significant role in the realisation of any e-health service. A significant proportion of the health demands of next future e-health care models will be satisfied through (private or public) broadband wireless networks. The application of these networks and the related technology (e.g. mobile terminals: PDA’s, phones) and services can be categorized as mobile based health services, or in short m-health services. The following definition of m-health services will be used in this document:

M-health services are e-health services that are based on the deployment of wireless mobile telecommunication technology to realise mobile based health (care) services.
M-health services are typically delivered by telematics systems. One of the major characteristics of these systems is the geographical distribution of system components which are interconnected by a (data oriented) transport system (i.e. telecommunication system). The mobility aspect of the m-health service is usually supported by a wireless component with limited resources in the transport system. These transport systems are typically hybrid systems that consist of wireless and wired communication technologies.

Telematics systems in the Internet domain generally use standardised Internet Protocol (IP) based communication protocols. Internet applications supported by telematics systems therefore use IP based application protocols to exchange application specific information (e.g. m-health service data) in a predefined way. Hence, the design of application protocols used by telematics systems that deliver m-health services to end-users must be optimised to the characteristics of the hybrid transport system. The protocol designer must make well motivated decisions to balance the application protocol functionality with the resources offered by the transport system.

This prerequisite for a successful application of m-health services makes it necessary to understand the behaviour of the transport system. Performance evaluation and assessment of performance data obtained from measurements can contribute to the understanding of the behaviour of complex transport systems.

2.1 Context

MobiHealth is a European project (IST-2001-36006, www.mobihealth.org) that explores the possibilities of GPRS and UMTS broadband wireless networks to support emerging m-health services. The service delivered by the MobiHealth system is an instantiation of an m-health service, where standards based TCP/IP Internet networking protocols are used to transport the m-health service data over these next generation broadband wireless networks.

The MobiHealth service is based on a telematics system that consists of four major building blocks: 1) BAN (Body Area Network), 2) BEsys (Back-End system), 3) transport system (i.e. data communication infrastructure) and 4) PortiLab2 end-user application. The MobiHealth team has used the BAN as a healthcare monitoring tool to measure vital sign data of ambulatory patients. BANs are controlled by the BEsys which makes the vital sign data arriving from a particular BAN available to an (remote) end-user (e.g. medical doctor). Transportation of vital sign data and control data is supported by the BANip application protocol which in turn is supported by a transport system.

The success of the MobiHealth service depends strongly on the performance (e.g. speed\(^1\), accuracy\(^2\) and dependability\(^3\)) of the transport service (i.e. service delivered by the transport system). Under the assumption that the MobiHealth service is used for patients with a critical condition, the performance of the transport service is of vital importance to the health of the patient. It is therefore necessary to study the behaviour and scalability of the transport service in a quantitative way in order to determine the impact of this service on the overall performance of

\(^1\) Time interval used to transport data from a source to a destination.
\(^2\) The degree of correctness with which a service is performed.
\(^3\) The degree of certainty with which the service can be used regardless of speed or accuracy.
the MobiHealth service. Hence, if the behaviour of the transport service can be predicted in relation to the m-health service data transported, the quality of the MobiHealth service delivered to the end-user can be predicted and adapted according to end-user (pre)defined quality of service profiles.

2.2 Research Questions

The goal of the MobiHealth project was to design, implement and try an m-health service instantiation that uses GPRS and UMTS wireless network infrastructures. The MobiHealth system architecture was designed in May 2002. Although GPRS network infrastructures were commercially available, the characteristics and behaviour of these networks in relation to transportation of large volumes of vital sign data was unknown. Furthermore, the characteristics and behaviour of GPRS networks differ significantly between operators! A pre commercial UMTS network became available the 1st of May 2003 to the MobiHealth project consortium members in the Netherlands. No empirical data was available about the behaviour of this UMTS network in relation to the support of m-health services.

The MobiHealth system architects used “second source” (practical or theoretical) information about characteristics and behaviour of GPRS and UMTS network infrastructures. As a consequence assumptions were made about the behaviour of the GPRS and UMTS network infrastructures used during the life time of the MobiHealth project. The design of the MobiHealth BANip was one of the crucial steps in the MobiHealth system design. One of the major concerns was the selection of the packet size for this application protocol: Fixed packet size or variable packet size? An ad hoc decision was made: variable packet sizes in range of 300 to 1800 bytes.

The authors had access to Vodafone’s commercial GPRS network and pre commercial UMTS network infrastructures in the Netherlands from May 1, 2003 until April 1, 2004 to answer the following research questions:

1) What performance evaluation methodology can be applied to assess a transport system supporting the MobiHealth BANip?
2) What are the performance criteria that should be considered for measurement, modelling and evaluation of the transport system?
3) What model can be used to represent the transport system and what are the conditions under which the model is valid?
4) Consider the MobiHealth BANip as a user confirmed application protocol: What is the optimal PDU size of a UMTS based transport system?
5) Consider MobiHealth BANip as a user unconfirmed application protocol: What is the optimal PDU size and PDU transmission rate of a UMTS based transport system?
6) What are the relevant performance issues related to the transport system?

2.3 Approach

In order to give an answer to the research questions an approach is followed that consists of six activities:
1) Description of system architecture and protocols.
2) Performance evaluation methodology selection/development (for specified performance criteria).
3) Specification of the transport service characteristics and description of the environment.
4) Performance measurement instrumentation.
5) Measurements execution and processing.
6) Performance evaluation of the transport system for confirmed and unconfirmed application protocols following the selected performance evaluation methodology (ies).

In the first activity background research is performed on the MobiHealth system architecture with a focus on the MobiHealth BANip. Part of this activity is also to develop a basic understanding of the GPRS and UMTS wireless networks operated by Vodafone in the Netherlands.

To derive the performance characteristics of the transport system, a performance evaluation methodology selection or development activity (2) was performed. This methodology is a recipe for performance evaluation of the transport system and must include the following steps: objectives determination, planning, instrumentation, execution and evaluation activities. These five steps form the basic ingredients for remaining activities.

In activity 3 the transport service characteristics are specified. For hybrid transport systems that include GPRS or UMTS networks, the transport service offered may have in general asymmetrical characteristics (i.e. different “upstream” and “downstream” transport capacity). Furthermore, the transport service is assumed to be reliable (i.e. based on TCP Internet protocol). The description of the transport system environment specifies how the transport system is being used. For the MobiHealth BANip the “upstream” interaction between the MobiHealth BAN and the transport system is of vital importance: vital sign data is send from a MobiHealth BAN over the transport system to the BEsys.

The instrumentation of the performance measurements is the focus of activity 4. It is based on available or to be developed tools and/or equipments. The specification of the transport service characteristics and description of the environment in activity 3 will be used as a starting point for this activity.

Activity 5 focuses on the performance measurements execution and processing. The objectives are: 1) to provide an extensive set of graphs visualizing the (raw) data obtained from each individual measurement activity, and 2) to generate the statistical data based on the (raw) data obtained from each individual measurement activity.

The final activity (6) focuses on the last two research questions. The performance of a UMTS transport system will be assessed for confirmed and unconfirmed application protocols. The selected performance evaluation methodology (ies) in activity 2 will be used for this purpose. A high level abstract model is derived based on activity 3 and validated/verified by means of performance measurements. The transport system abstract model verification aims to check
whether it fulfills the quantitative requirements that have been identified. The model is used for delay, bottleneck and scalability analysis of the transport system.

3 Transport System Performance Evaluation

In the first few months of the MobiHealth project the system architects were designing the MobiHealth BANip application protocol based on “second source” (practical or theoretical) information about characteristics and behaviour of GPRS and UMTS network infrastructures. There was practically no “first hand” public information available about the behaviour of these networks in relation to the transport of m-health related data. The system architects had to base the BANip design on assumptions rather then facts about the real-world behaviour and performance of the MobiHealth transport systems of interest.

Obtaining facts about the real-world behaviour and performance of GPRS and/or UMTS based transport system is a complex and daunting activity. The “art of performance evaluation” of transport systems (e.g. data communication networks) is in many cases based on mathematical models (e.g. queuing models, Petri Nets and state machines [Have1998]) and simulation of these systems. According to [Hoek1997] a precise, systematic approach towards performance evaluation of real-world transport systems by means of measurements is needed. Their approach is based on a methodical approach for performance evaluation of computer systems by [Jain1991]. Given the fact that we had the availability over Vodafone’s commercial GPRS network and (pre) commercial UMTS network from July 2003 until March 2004, the choice was made to use measurements for the performance evaluation of the MobiHealth transport system.

We believe that performance evaluation of a transport system by means of measurements can only be successful if a methodological approach towards the measurements activity is used. The approach of [Hoek1997] and [Jain1991] are therefore adopted and used to describe the measurement methodology (section 3.1) for our performance evaluation process.

Deriving a mathematical (i.e. analytical) model from the performance measurements results is an interesting opportunity. The presented modelling methodology in section 3.2 leads to a performance model that will be used as a tool in the overall performance evaluation of the transport system.

3.1 Measurement Methodology

A performance measurement activity aims to find estimates for system performance criteria based on reproducible experiments. These estimates give a quantitative analysis of the behaviour of the system of interest. In this section a methodological approach to the performance measurement activity is presented.

The performance evaluation of a system by means of measurements consists of the following steps [Jain1991]:
1. State the Goals and System Definition

The first phase of the measurement activity is answering the question what system performance characteristics are of importance and why (i.e. what is the objective of the study) and what is the system as an object of the study (i.e. what are the system boundaries). A description of the system should be provided as far as it’s known and relevant to the stated objectives. For complex systems, that consists of many subsystems, this activity can imply the system decomposition process, such as the subsystem of interest is identified, or even the particular component of the system is highlighted as of interest. The objective of the study is the key consideration while system delineation. Moreover, the goal set strongly affects the kind of accuracy required for a performance study.

For example, when considering the goal of speed determination of the PC, the goals and system definitions stated by the: end-user and the system architect will be different. Namely, the end-user would usually state that he is interested in the speed of particular application running on this PC, while the system architect will scale the goal down to the speed determination of the (particular) processor, as a component of this complex system.

2. List services and their outcomes

The next step is to answer the question on what services the system offers and what are their respective outcomes. The decision regarding the services (and their corresponding outcomes) to be considered for evaluation is taken. A studied system provides the set of services which can give number of possible outcomes. Generally, these outcomes can be classified into three categories: the system may perform the service correctly, incorrectly or refuse to perform the service. Some of these outcomes are desirable and some are not. If the system performs the service incorrectly, an error is said to have occurred.

For example the database system provides searching and sorting services. The service of interest may be that database responds to the searching query. It may answer to the query correctly, incorrectly (e.g. due to its inconsistency) or not at all (e.g. due to deadlocks). The outcome of interest may be the correct one.
3. Select Performance Criteria (i.e. Metrics)

The next step is to identify, select and define the performance criteria (i.e. metrics) of the transport system. The system is considered as defined in step 1 and the services (and their outcomes) are considered as defined in step 2. Performance metrics are associated with the three possible service outcomes (i.e. successful service delivery, error, and service unavailability) and they are related to the speed, accuracy and availability of the services offered by the system.

The performance of the transport system can be measured, for example, in terms of availability, speed (e.g. delay, “goodput”) and reliability (e.g. error rate of the data being transported). For each service offered by the system there is a number of speed metrics, a number of reliability metrics and a number of availability of metrics. For the system that offers more than one service the number of metrics grows proportionately.

Let us consider the example of the database system from the previous step. If the system answers to the query correctly, its performance is measured by the time taken to perform the service, the rate at which the service is performed and the resources consumed while performing the service. These three metrics related to the time-rate-resource for successful performance are also called responsiveness, productivity and utilization metrics respectively. The utilization is the percentage of time the resources of the gateway are busy for the given load level. The resource with the highest utilization is called the bottleneck.

The desired accuracy of the evaluated performance metrics is of the vital importance. The question: “How detailed the estimation of particular metrics is required?” is raised. According to the assumed accuracy level, the mean values, variances or complete distributions of measures of interests need to be obtained.

4. List System and Workload Parameters

In this activity the main concern is the set of parameters that affects the system performance. There are: system-specific parameters and workload parameters distinguishable. System parameters include both the hardware and software parameters, which generally do not vary among various installations of the system. Workload parameters are characteristics of user’s requests, which vary from one installation to the next. For example, the buffer sizes (system parameter) and volume of data being transported (workload parameter) affect the transport system performance.

5. Select Factors and their Levels

The list of identified parameters can be divided into these that will be constant during the measurements and these that will be varied. The latter group of parameters are called factors and their values are called levels. The selection criteria of parameters as system factors should be: a) the influence level of particular parameters on the system performance, and b) the feasibility of changing these parameters. Regarding the transport system, the important factors can be: size and arrival rate of the packets to be transported.
6. **Select the Workload**

The workload consists of the list of service requests and their arrival intensities. For particular service request, the values of workload and system parameters are decided. The list of service requests is a representative of the system usage in real life or the planned system usage. For measurements, the workload consists of user scripts to be executed on the system. The measurements done on the transport system need to explore the variety of possibilities of transport service requests.

7. **Design and Execute the Experiments**

The decision on what experiments and in which order to perform in order to obtain the maximum information with minimal effort, is taken. The equipments and/or additional code needed for experiments and their instrumentation need to be listed. The report on: when, where, how and by whom the experiments should be executed follows.

8. **Analyse, Evaluate and Interpret the Data**

At this phase the (raw) measurement results needs to be collected and (roughly) validated. Also the generation of the statistical data based on the (raw) data obtained from each individual experiment is assumed. Data interpretation phase can follow only if the obtained results are explainable and accepted, otherwise the performance evaluation measurement process needs to be revised and, if necessary, repeated (i.e. start from step 1).

9. **Present the Results**

At this point the obtained data are analysed, presented and discussed, conclusions are derived and recommendations are made. It is important that the results be presented in the manner that is easily understood (e.g. in graphic form). Often at this point the knowledge gained by the study may require going back and reconsidering some of the decisions made in previous steps. For example redefinition of the system boundaries or including other factors and performance metrics that were not considered before, would be needed. In this case another cycle through all the steps would be required.

### 3.2 Modelling Methodology

Based on the measurement results, the feasibility of the derivation of the transport system’s high-level abstract performance model is studied. Obtained model will be a tool for a further performance analysis of the system. The model is intended to reflect the system performance characteristics obtained in the measurement phase. For example, if the focus of the measurements activity is to obtain the speed-related characteristics (e.g. system delay, “goodput”) of the transport system, then a queuing model is the most appropriate for quantitative approximation of performance system characteristics.
Transport System modelling methodology is a recipe for the performance evaluation of the system by means of system modelling and it consists of the following steps:

Steps 1 to 6 were already defined and executed by means of the Measurement Methodology. The concepts used while modelling the transport system have to be consistent with the ideas provided while executing the measurement methodology.

7. **Create Model**:

The first question to be answered is “what model is best to yield the desired metrics”? There are three possible abstract representations of the system: queuing models, Petri nets and abstract state machine models. After answering this question the identification of the system specific parameters (e.g. servers, queues, service disciplines, for a queuing model) and workload-specific parameters (e.g. arrival process, service time) should follow.

8. **Parameterise the Model**:

The data obtained from the measurements needs to be carefully interpreted and processed to estimate the model parameters (e.g. workload parameters, service times, etc.).

9. **Validate and Verify the Model**:

The measurement data can also be used to validate the appropriateness and accuracy of the model. This is done by comparison of the results from the model with measurements. If the model needs refinements, then steps 7 (with new assumptions), 8 and 9 are repeated.
4 Transport System Services Modelling

This chapter focuses on the first phase of the measurement methodology presented in section 3.2. It covers step 1, 2, 3 and 4 of the methodology (Figure 4.1).

| 1. State the Goals and System Definition |
| 2. List Services and their Outcomes     |
| 3. Select Performance Criteria (i.e. Metrics): |
| 4. List System and Workload Parameters |
| 5. Select Factors and their Levels      |
| 6. Select the Workload                  |
| 7. Design and Execute the Experiments   |
| 8. Analyze, Evaluate and Interpret the Data |
| 9. Present the Results                  |

Figure 4.1 Measurement Methodology Phase 1.

The performance evaluation goals state the objectives of the measurement activity for a particular system. An answer to the question: “Why is a performance evaluation of the transport system needed?” is provided in section 4.1.

A definition (i.e. description) of the transport system (as a whole) and the decomposition of it into major sub-systems are provided in section 4.2. The “system under test” concept is introduced and used to delineate (setting the boundaries) the part of the transport system that’s subject to the performance measurement activity. Sections 4.1 and 4.2 cover the first methodology step.

Section 4.3 uses the [Viss2000] service concept to describe the services provided by the transport system and “system under test”. These services are identified and their possible outcomes analysed by means of service decomposition into “service layers” (methodology step 2).

Controlled generation of specified messages destined for the “system under test” and measurement of the correlated effects are important functions in the performance measurements activity. These functions are provided by an evaluation system. Section 4.4 describes the functional relation between an evaluation system and the “system under test”.

When the context of the performance measurements activity is clear, an answer to the following question: “What are the performance criteria of interest?” is provided. Section 4.5 introduces the “3 x 3 matrix approach” and applies it to describe the “system under test” performance criteria of interest (methodology step 3).

Performance measurements can be influenced by system parameters (e.g. transport link capacity). In section 4.6 these “system parameters of influence” are classified into “descriptive” and “usage” related parameters. This section will be concluded with a selection and abstract description of the parameters used in the performance measurements.

---

4 distinguishing qualities
In the last section (4.7) of this chapter the workload (“usage” related) parameters of influence are specified. Values will be assigned to the workload parameters and the selected system parameters (“descriptive”) in section 4.6. Sections 4.6 and 4.7 cover methodology step 4.

4.1 Objectives

Performance evaluation of a system based on measurements is clearly a difficult and time consuming activity. Systems may be very complex (i.e. consist of many sub systems) and there are potentially many performance criteria that can be measured (e.g. speed, dependability). It is therefore important to state very clearly what the objectives of the performance evaluation are, that is by answering the question:

Why is a system performance evaluation of the transport system needed?

Context
The success of the MobiHealth service depends strongly on the performance (e.g. speed, dependability) of the service provided by the transport system. Under the assumption that the MobiHealth service is used for patients with a critical condition, the performance of the transport service is of vital importance to the health of the patient. It is therefore necessary to study the behaviour and scalability of the transport service in a quantitative way in order to determine the impact of this service on the overall performance of the MobiHealth service. Hence, if the behaviour of the transport service can be predicted in relation to the m-health service data transported, the quality of the MobiHealth service delivered to the end-user can be predicted and adapted according to end-user (pre)defined quality of service profiles.

Transport system of interest
The MobiHealth system design abstracts from a transport system’s constitution\(^5\) and character\(^6\) for transportation of BANip application protocol data. The only prerequisite of the transport system is to support the transport of IP protocol datagrams. Assuming that a MobiHealth transport system consists of multiple sub-systems, it becomes necessary to limit the performance evaluation activity to one transport (sub) system of interest:

Vodafone’s\(^7\), Telia’s, and Telefonica’s (pre) commercial UMTS networks\(^8\) (VTT3G)

MobiHealth’s BANip - transport system interaction
The BANip protocol is responsible for the transportation of BAN data (i.e. management data and patient vital signs data) on a real time\(^9\) basis. The current implementation of the BANip requires the transport system to provide a reliable transport service (section 2.3). The real time transportation of data requirement can only be fulfilled by a transport system that supports the transmission of BANip packets (a.k.a. Protocol Data Units) within a maximum delivery time.

---

\(^5\) Transport system that consists of multiple interacting transport systems acting as a whole.

\(^6\) Interacting transport system may be implemented in a different way; i.e. use different types of data communication technologies. This results in a hybrid transport system used by the MobiHealth BANip application protocol.

\(^7\) Vodafone announced on February 12, 2004 the launch of it’s commercial UMTS service on February 16, 2004h

\(^8\) We must note that the major part of data come from the Vodafone network.

\(^9\) As fast as required. A realtime system must respond to a signal, event or request fast enough to satisfy some requirement [TechWeb]. The response time requirement for the MobiHealth system is defined in chapter 2.
The BANip packet size and the transport system’s delivery time may be correlated; small packet sizes may have a different delivery time than large packet sizes. Under the assumption that the delivery time of small packet sizes is different from large packet sizes, the following question can be raised:

What would be the maximum BANip packet size and packet rate for a specified (maximum) delivery time?

Considering the current MobiHealth BANip implementation (version 3.1.1) is using packet sizes in range of 300 to 1800 bytes, the answer to the previous question will also provide the answer to the next question:

Are the packet sizes of the current MobiHealth BANip implementation adequate?

Objectives

Now that the context is clear, the first question regarding our objectives is put into this context and answered:

Consider 3Gnet as a MobiHealth transport system for the BANip application protocol:

Why is a performance evaluation of this system needed?

1) To characterize the behaviour of this system as a MobiHealth transport service.
2) To determine the maximum BANip packet size and packet rate for a specified (maximum) delivery time.
3) To determine if the packet sizes of the current BANip implementation are chosen adequately.

4.2 System Description and Boundaries

The MobiHealth system architecture discussed in chapter 2 presented a transport system as a “cloud” of one or more (interacting) transport systems. Recall that BANs and the BEsys are geographically dispersed, the BANs are being worn by roaming patients and the BEsys is typically located in an enterprise network (LAN) of a (healthcare) service provider. The probability of having one transport system provider delivering the transport service to the MobiHealth system is therefore low. In general a MobiHealth transport system consists of at least three (transport) sub-systems controlled by three (or more) organizations: Mobile Operator Network (e.g. UMTS network), Internet, and enterprise network, respectively, controlled by a mobile operator, Internet Service Provider (ISP) and healthcare care service provider.
4.2.1 System Decomposition

The transport system of interest is Vodafone’s, Telia’s and Telefonica’s (pre) commercial UMTS networks in the Netherlands, Sweden and Spain (VTT3G). Vodafone acts as the transport service provider (i.e. mobile operator) for mobile or roaming users; in case of MobiHealth the mobile users are roaming patients wearing a BAN. Assume the BEsys is located at the campus network (i.e. enterprise network) of the University of Twente (UTnet) and the university fulfills the role of a health care service provider that operates the MobiHealth service. Somehow a connection has to be made between VTT3G and the UTnet for the MobiHealth service to come into operation. A third party is identified acting as an Internet service provider, offering a connection service to Vodafone and the University of Twente. Hence, the whole MobiHealth transport system has a hybrid character and consists in this example of three interacting sub-transport systems: VTT3G, Internet and UTnet.

The MobiHealth transport system can be considered as a whole and presented as an abstract design model to master the complexity of the system. However, it is necessary to decompose this system into sub-systems to evaluate the contribution of each individual sub-system (in particular the transport system of interest) to the overall performance of the transport system. If each sub-system is responsible for a section of the ‘end-to-end’ IP communication path between the IP-capable MobiHealth system components, then the ‘end-to-end’ communication path can be decomposed according to Figure 4.2.

Note: Recall that the MobiHealth BANip application protocol is based on the IP protocol (section 2.4), therefore the MobiHealth transport system (and its sub-systems) must support the transport of IP datagrams (i.e. provide an IP service).

In version 3.1.1 of the MobiHealth system, the MBU is the only IP entity in the BAN able to communicate with the BEsys, therefore the intra BAN communication is considered outside the scope of the performance measurements. The MBU will be directly connected to the VTT3G; i.e. the transport system of interest and the BEsys will be connected to the UTnet.

Note: The intra enterprise communication between the host of the end-user and the BEsys is outside the of the performance measurements.

---

10 The V3GNL transport system actually delivers a UMTS (i.e. link layer service) service to mobile system users and an IP service to the Internet. For simplicity reasons we assume that the V3GNL transport system delivers a UMTS based IP service to the MBU. Section 4.3 provides an argumentation for this assumption.
4.2.2 System under Test

This section discusses a conceptual framework that will be used to delineate the MobiHealth transport system and its sub-systems that are important to the performance measurement activity. The conceptual framework consists of a system of discourse\textsuperscript{11} containing a “system under test” (i.e. the transport system of interest) and is based on the “black box – white box” principle. According to [Hoek1997] the conceptual framework consists also of an evaluation system. This system is described in section 4.4.

Black box – white box principle

The “black box – white box” principle is applied to the decomposed abstract design model of the MobiHealth transport system (Figure 4.2) to generate a conceptual service oriented view\textsuperscript{12} of this system. The MobiHealth transport system as a whole can be represented as a black box that provides certain types of transport services. From a transport service user perspective it is of no interest how the transport system delivers the service, as long as it delivers the transport service at a certain (sometimes pre-defined) quality level. Hence, the transport system is a “black box” providing specified service types to the user.

For performance evaluation purposes the black box model is in many cases not useable. This depends of course largely on the objectives of the performance evaluation. Our objective is to evaluate the performance of the VTT3G sub-system which is part of the transport system as a whole. Furthermore, Figure 4.2 provides knowledge about other sub-systems of the transport system and as a consequence discloses the “black box”. The inner working or one possible way of implementation of the “black box” is presented. Hence, the MobiHealth transport system as a “black box” is converted to a “white box” (Figure 4.3). The transport system’s sub-systems are each represented by a “black box”. For our performance evaluation objectives the internals of the sub-systems are of no interest\textsuperscript{13}. We are solely interested in the IP service delivered by these systems. The interactions between the systems (includes MBU and BEsys) take place at the boundary of each system by means of a IP Service Access Point (IP_SAP); i.e. IP protocol datagrams can only be exchanged between systems via their individual IP_SAPs.

\textsuperscript{11} System of Discourse: an inclusive class of entities that is tacitly implied or explicitly delineated as the subject of a statement, discourse, or theory [Webster]
\textsuperscript{12} Based on [Viss2000].
\textsuperscript{13} Vodafone kindly refused to disclose their V3GNL network for commercial reasons.
System of Discourse and System under Test

The MobiHealth transport system is considered the system of discourse (SoD). Inside this SoD the subject of the measurement activities is identified as the System Under Test (SUT). Figure 4.4 provides an abstract representation of the SoD including the SUT.

The SUT is defined as:

That part of the system of discourse of which the quantitative aspects of behavior are under study (keeping the qualitative aspects of the behavior the same) [Hoek1997].

The non-SUT part of the SoD may influence the performance measures of the SUT. Hence, a description of the non-SUT part is considered to be relevant. It also provides the possibility for the non-SUT part to play a role in the workload generation. This possibility will be used in the description and instrumentation of the evaluation system (section 5.1). Recall that our objective is to evaluate the performance of the VTT3G transport system. Hence:

The VTT3G transport system is defined as the SUT.

The “black box – white box” model of the MobiHealth transport system (Figure 4.3) is now converted to the SoD conceptual framework and presented in Figure 4.5. The MBU and BEsys in Figure 4.3 are respectively replaced by an abstract “service user 1” and “service user 2”.

---

14 The reader is invited to read sections 4.2.1 and 7.2 of [Hoek1997] for the argumentation.
4.3 Service Description

In general, system users are interested in the particular service(s) delivered by this system. They are not interested in the implementation aspects of the system. We are from a performance evaluation perspective interested in the service(s) of the SUT. These services are defined as the external observable behaviour of the SUT. Figure 4.5 showed the SUT being embedded in the SoD. The SUT delivers an IP service to another system embedded in the SoD and to (external) service user 1 of the SoD. Figure 4.5 abstracts from the service interaction mechanism between the SUT and the SoD. Since the non-SUT part of the SoD may influence the performance measures of the SUT, a description of the service(s) delivered by this part is considered to be relevant.

4.3.1 Service Decomposition

Recall that the SoD as the MobiHealth transport system delivers a reliable transport service to the MobiHealth MBU and BEsys sub-systems. We assume that the Transmission Control Protocol (TCP) is used to realise this service. TCP uses the (lower level) datagram transport service (i.e. IP service) provided by the IP protocol (see section 1.7.2 [Kuro2001]). The SoD service is decomposed according to the Internet Protocol Stack and presented in Figure 4.6.

Note: A layer in the Internet Protocol Stack model provides a service (i.e. a function). Services are represented by rounded rectangles. The ellipse "SAP" (Service Access Protocol) is used to represent the service provided by a layer.
Point) between services represent a point of interaction. Interactions between the IP service inside the Internet and UTnet sub-systems are considered to be irrelevant and therefore not modelled.

In section 4.2.1 we assumed that the SUT (i.e. VTT3G) delivers an IP service. Closer study of the SUT reveals that this is not the case. The service delivered to the mobile user is a UMTS service that supports higher level services; e.g. link layer service (PPP). The service delivered to the Internet remains the IP service. Figure 4.7 provides the SoD – SUT service decomposition that closer matches reality. The objective of the SUT performance evaluation is focussed on the BANip application protocol. The BANip is supported by the reliable transport service of the SoD. It is possible to consider the transport service, IP service and host-to-network service as part of the SoD or SUT.

We prefer to separate the transport service from the other services and model it as part of the SoD (as presented in Figure 4.6). This provides the possibility to have alternative implementations of the transport service (e.g. UDP protocol). Furthermore, we prefer the SUT to deliver one type of service at its boundary which is as close as possible to the actual type of service delivered by the VTT3G network (i.e. the network-to-host service and IP service). Therefore, the IP service and host-to-network service are modelled as part of the SUT. To elaborate on this choice, we have to leap forward to the instrumentation activity:

The host-to-network service is implemented on pre commercial UMTS terminals provided by Vodafone. No other UMTS terminals were available during the performance evaluation of V3GNL. Vodafone guaranteed that the UMTS terminals were interoperable with the V3GNL network. Hence, there was an indissolubly relation between the terminal and the SUT and modelling the host-to-network service as part of the SUT seems logical.

The IP service is not implemented on the UMTS terminals but on the computer systems that interact with the terminals. It seems therefore logical to model it as part of the SoD. We

---

16 The host-to-network service represents the combined services of the link layer and physical layer of the Internet Protocol Stack.
17 This rules out the transport service being modelled as a CUS inside the SUT.
consider the IP service implementation in the operating system running on the computer systems as transparent; i.e. of no influence on the behaviour of the SoD. Therefore, the location of the IP service in the SoD – SUT service decomposition model is not bound to the SoD. We prefer the SUT to deliver one type of service to its users: the IP service. Hence, the IP service is modelled as part of the SUT (assumed to be implemented by an Access Point Network device).

### 4.3.2 Service Characteristics Analysis

The service decomposition of the SoD and SUT described in the previous section abstracts from the detailed services characteristics. These characteristics are relevant for the performance criteria of interest selection (section 4.5). The SoD may hide the service characteristics of the SUT for the service user. For example, the IP service delivered by the SUT does not guarantee in sequence delivery of IP datagrams (packet). The TCP service of the SoD hides this characteristic by reordering the received IP datagrams before delivering them as a segment (message or part of a message) to the service user.

Figure 4.6 is used as the reference model to analyse the service characteristics of the SoD and the SUT. It is inevitable that some of the SUT service characteristics are hidden by using the TCP based transport service between the SUT and the service user. Recall, the choice for TCP as a transport service is the consequence of the fact that version 3.1.1 of the BANip application protocol is based on TCP.

*Note: If future implementations of the BANip are based on the UDP transport service, the characteristics of the SUT will be disclosed. UDP is a “light weight” protocol providing a connection less service that enables the transmission of independent packets of data from one system to the other, without guarantee about delivery.*

The service characteristics analysis of the SUT is based on publicly available information of the VTT3G transport system. Furthermore, we consider the objectives of the performance measurements as the focal point of the analysis and therefore assume certain characteristics of the SUT and SoD. The service characteristics of the SUT and SoD are presented below.

**SUT**

1) IP datagram service. A connectionless service that delivers IP datagrams from the ingress SAP (service user side) to the egress SAP (SoD sub-system side) and vice versa, without guarantee of delivery (assumed implemented by an APN).

2) Asymmetrical transport service with different “upstream” and “downstream” transport capacity. The ingress SAP receives “upstream” datagrams from the service user and the egress SAP receives “downstream” datagrams form a SoD sub-system (e.g. the Internet). Figure 4.8 presents the maximum transport capacity of the SUT’s “upstream” (64 Kbps) and “downstream” (384 Kbps) links.

3) Transport service capacity correlates to “upstream” and “downstream” datagrams. The SUT reacts to changes in the volume and rate of the datagrams offered at its SAPs. It supports two “UMTS bearers” for “upstream” and four for “downstream”. These are identified as
coloured rectangles in Figure 4.8. The “upstream” link has bearer capacities for 0-16 Kbps and 16-64 Kbps, and the “downlink” for: 0-16 Kbps, 16-64 Kbps, 64-128 Kbps and 128-384 Kbps\(^{18}\).

Figure 4.8 SUT upstream and downstream transport capacity.

**SoD**

1) TCP service at the system boundaries (SAPs). Connection oriented service that enables TCP segment (service user message or part of a message) exchange between transport service SAPs in a reliable (i.e. no segment loss) and in sequence fashion\(^{19}\) (the order of segments send is identical at the receiving end).

2) Service is available. The SoD and its sub-systems are always available for performance measurements independent of the type of measurements.

3) Service is reliable. The SoD performance its function as a transport service in a reliable manner. No loss of data is induced by any of the SoD sub-systems service providers (e.g. due to maintenance).

4) Service correctness. The probability of traffic corruption induced by the SoD sub-systems service providers is zero.

### 4.4 Evaluation System

The SoD is distinguished from an evaluation system that is used to study the quantitative aspects of (parts of) its behaviour. The evaluation system stimulates the SoD to “reveal” its behaviour and in parallel measure this behaviour. A functional model of the evaluation system is presented in Figure 4.9. It consists of two functional building blocks\(^{20}\): a workload generator and measurement function.

---

\(^{18}\) Source Vodafone NL.

\(^{19}\) See section 3.5 of [Kuro2001] for a detailed discussion of the TCP protocol.

\(^{20}\) These functional building blocks are also denoted as sub-systems.
Figure 4.9 SoD and evaluation system, functional view.

Note: The SoD, SUT and evaluation system are considered to provide a particular function. Functions are represented by rounded rectangles. Arrows between functions denote the flow of information. Flows may split or join which is represented by a white diamond and a black square respectively. They enter or exit functions via triangular input and output interfaces.

The workload generator has an interaction relation with the SoD. It influences the behaviour of the SoD by generation of stimuli and is also able to react to stimuli from the SoD. The measurement function measures the reaction to generated stimuli. It has no interaction relation with the SUT and should influence the behaviour of the SUT as little as possible. The fact that the workload generator and the measurement function have different relations to the SoD, results in the conclusion that they can not interact (i.e. co-operate) with each other.

Note: If the measurement function is able to interact with the workload generator, therefore influencing the behaviour of this generator, the measurement function is able to influence the behaviour of the SUT indirectly via the workload generator. This should be avoided!

In the previous text the implicit assumption was made of the existence of a one-to-one relation between the SoD and the evaluation system. This is indeed the case from a high abstract view; the evaluation system is considered as a whole, but may consist of many interacting sub-systems. [Hoek1997] section 4.2.1 provides only a high abstract view on the relation between the SoD and the evaluation system. We believe that multiple instances ([*] symbol in Figure 4.9) of evaluation systems may be related to one SoD.

Performance measurements of a SoD with a distributed character (i.e. consists of geographically dispersed components) may require more than one evaluation system. For example: performance measurements with a scalability objective (How many nodes can be supported by the SUT?) may use only one SoD but many instances of the evaluation system. Each workload generator of an evaluation system instance is competing for SoD resources.

Figure 4.10 presents an example scenario for two evaluation systems and one SoD, and abstracts from the SoD service access points (SAPs). Having more than one evaluation system creates interesting instrumentation problems:

How to synchronise the behaviour of the evaluation system instances to create co-ordinated actions?
How to synchronise the functional building blocks of the evaluation system instances in time?

We deal with these questions in section 5.1.3.

The workload generators of the evaluation system instances may interact with each other by means of mutually generated but correlated stimuli for the SoD, for example:

1) workload generator 1 generates stimulus 1 for the SoD
2) the corresponding behaviour of the SoD is to:
   a) forward a data item to workload generator 2, and
   b) generate stimulus 1 for workload generator 2
3) workload generator 2 reacts to the SoD stimulus 1 by generating stimulus 2 for the SoD
4) the corresponding behaviour of the SoD is similar to step 2.

The instances of measurement functions may interact between each other, but are not allowed to interact with any of the workload generator instances (recall previous note).

Recall the functional view of the evaluation system as presented in Figure 4.9. This model is converted to a service oriented view. The interaction points between the Sod and the evaluation system are modelled as SAPs (Service Access Points). The “distance” between the SAPs and the SUT is modelled as a transport service providing function block. Figure 4.11-A presents the service oriented model. The transport service between the evaluation system’s measurement function and the SUT provides an “ideal” service; i.e. transports information between the MF_sap
and the IP service SAP with no influence to the performance measurements. The transport service between the WG_sap and the IP service SAP implements the TCP protocol entity (determined in section 4.3.2).

We prefer the evaluation system’s measurement function to interact at the same “level” as its workload generator. To explain this choice, we have to think about the instrumentation of the evaluation system and its functions:

The SoD service decomposition presented in section 4.3.1 described the TCP transport service SAP as the (only) interaction point between the SoD and its service users. The evaluation system can be seen as a service user. This means that the workload generator and measurement function have to interact with the SoD by means of TCP SAPs.

As a side effect, the instrumentation of the evaluation system may be simplified. There is not need for the measurement function to interact at the IP SAP. This interaction is considered more difficult to realise than an interaction with a TCP SAP (i.e. tcp_socket). For example: Consider the scenario where multiple instances of an evaluation system are interacting with the SoD and SUT. The measurement function of each instance needs to filter inbound and outbound information on the IP service SAP to determine which information belongs to its instance.

The disadvantage of using the SoD TCP SAP for measurements is the possible influence of the TCP protocol entity on the accuracy of the performance measurements. This will be discussed in section 4.6 System Parameters of Influence.

In Figure 4.11-B the revised interaction between the evaluation system and the SoD is presented. The measurement function and the workload generator interact with the SoD by means of their “own” SAP, respectively, the MF_sap and the WG_sap.

The functional view in Figure 4.10 presented two evaluation systems interacting with the SoD. Based on the service oriented model presented in Figure 4.10-B, a service view with the complete
SoD can be constructed (Figure 4.12). Each workload generator and measurement function of an evaluation system instance is interacting with the SoD via its “own” SAP. The common resource of the workload generators is the SoD.

Workload Generator

The primary function of the workload generator is to stimulate the SUT by providing a “workload” to the SUT_sap. It can take the initiative (generates a stimulus), or responds to a stimulus received from the SUT_sap (multiple instances of evaluation system scenario). A workload generator can be envisioned as a service user and therefore many different workloads can be offered to the SUT.

In principal the workload generator should generate SUT-service related interactions at the SUT_sap to “reveal” the behaviour of the SUT. These interactions consist of service primitive (SP) exchanges. If SP’s are logically grouped with a prescribed temporal ordering, they are considered as a service element (SE). Interactions are in this case SE exchanges; this is the most common representation of the service related interaction and therefore used in the rest of this chapter. If it does not generate SE interactions, there are no events to be measured. The workload
provided to the SUT must at least consist of multiple executions of the SUT related SE’s to which the measure pertains. It seems therefore natural to specify the workload as a sequence of particular SUT related SE’s issued to the SUT.

Ideally, the workload generator interacts directly at the SUT’s SAP (i.e. SUT boundary). It may also interact indirectly by using the capabilities of the non-SUT part of the SoD (see Figure 4.10-B). For example, it may not be possible to “get access” to the SUT’s SAP due to instrumentation issues (section 5.1). The workload generator interacts with the SUT by means of the WG_sap defined at the SoD boundary. As a consequence, the behaviour of the SoD as a response to the generated workload may influence the behaviour of the SUT. This may result in a decrease of the overall performance measurements accuracy. The workload generator may also (optionally) introduce background load into the SUT. We are not using this option in our performance measurements.

The workload offered to the SUT can be parameterized. A particular choice of a workload is defined by assigning values to the so called “workload parameters”. They can be described as characteristics of user requests (e.g. SE request, SE rate, length of SP) which vary from one SoD and SUT instantiation to the next. Identification of the workload parameters requires knowledge about the usage of the SoD and SUT.

SoD and SUT instantiations can also be parameterised by “system parameters” (a.k.a. “system parameters of influence, section 4.6). These include both hardware (e.g. transport link capacity) and software (e.g. queue depth or buffer size) parameters which generally don’t vary among various instantiations of the system. A particular workload choice is called a “workload instance”. A performance measurement action can only take place if both a system and a workload are selected and parameterised.

Measurement Function

The measurement function of the evaluation system solely operates at the SoD or SUT SAP level. Figure 4.10-B showed that the measurement function is interacting only with the SoD through its dedicated MF_sap. Hence, the non-SUT part plays a role in the measurement action. The measurement action should be performed passively; i.e. the measurement function “listens” on the MF_sap for events. An event is considered to be the arrival of a SP or SE. The primary goal of the measurement function is to measure events which “happen” at the MF_sap but find there origin at a workload generator’s WG_sap.

4.5 Performance Criteria of Interest

A performance criterion of a particular SUT can be defined as a characterizing mark or distinguishing quality of that system. For example, the performance criterion of interest of the SUT can be “speed”. In general a performance criterion is an abstract collection of performance parameters which share the same context, e.g. the “speed” context in this example. The ITU-T provides a systematic method of identifying system performance criteria and their related performance parameters for digital networks (including ISDNs). This method is known as the 3 x 3 matrix approach [ITU1993]. Although we have the impression that the focus is aimed at circuit

---

21 The number depends on the (anticipated) behaviour of the SUT and the accuracy of the statistical data that need to be obtained.
switched networks, it can and will be used to select the performance criteria of interest of the SUT. We start with a brief overview of the 3 x 3 matrix\textsuperscript{22} before selecting the criteria and their parameters of interest.

### 4.5.1 ITU-T 3 x 3 matrix approach

In this approach the following definitions are obtained used:

**Primary performance parameter:**

*A parameter or a measure of a parameter determined on the basis of direct observations of events at services access points or connection element boundaries.*

**Derived performance parameter:**

*A parameter or a measure of a parameter determined on the basis of observed values of one or more relevant primary performance parameters and decisions thresholds for each relevant primary performance parameter.*

The 3 x 3 matrix approach is presented in Figure 4.13. The main features are:

1) Each row represents one of the three basic and distinct communication functions.
   *Note: The access function represents the connectionless as well as connection-oriented services which are possible with transport systems. We will focus only on the connection-oriented service implemented by the TCP protocol.*

2) Each column represents one of the three mutually exclusive\textsuperscript{23} outcomes possible when a function is attempted.

3) The 3 x 3 matrix parameters are defined on the basis of events at the SoD/SUT boundaries (i.e. SAP) and are termed “primary performance parameters”. “Derived performance parameters” are defined on the basis of a functional relationship of primary performance parameters.

4) Transport system primary performance parameters should be defined so as to be measurable at the SAPs of the SoD/SUT to which they apply.

5) Availability is a derived performance parameter.

![Figure 4.13 3 x 3 matrix approach](image)

\textsuperscript{22} The description of the matrix approach is copied from [ITU1993] document and modified it for our purposes.

\textsuperscript{23} Outcomes being related such that each excludes or precludes the other [Webster].
Description of the basic communication functions

access

The access function begins upon issuing an access request signal or its implied equivalent at the interface between a user and the communication network. It ends when either:

1) a ready for data or equivalent signal is issued to the calling users; or
2) at least one bit of user information is input to the network (after connection establishment in connection oriented services).

It includes all activities traditionally associated with physical circuit establishment (e.g. dialling, switching, and ringing) as well as any activities performed at higher protocol layers.

user information transfer

The user information transfer begins on completion of the access function, and ends when the “disengagement request” terminating a communication session is issued. It includes all formatting, transmission, storage, error control and media conversion operations performed on the user information during this period, including necessary retransmission within the network.

disengagement

There is a disengagement function associated with each participant in a communication session: each disengagement function begins on issuing a disengagement request signal. The disengagement function ends, for each user, when the network resources dedicated to that user’s participation in the communication session have been released. Disengagement includes both physical circuit disconnection (when required) and higher-level protocol termination activities.

Description of the performance

speed

Speed is the performance criterion that describes the time interval that is used to perform the function or the rate at which the function is performed. (The function may or may not be performed with the desired accuracy.)

accuracy

Accuracy is the performance criterion that describes the degree of correctness (e.g. error probability) with which the function is performed. (The function may or may not be performed with the desired speed.)

dependability
Dependability is the performance criterion that describes the degree of certainty (e.g. blocking probability) with which the function is performed regardless of speed or accuracy, but within a given observation interval.

### 4.5.2 Selection of Performance Criteria and Parameters

Recall the performance evaluation objects presented in section 4.1:

1) Characterize the behaviour of the SUT (i.e. VTT3G) as a MobiHealth transport service,
2) determine the maximum BANip packet size and packet rate for a specified (maximum) delivery time, and
3) determine if the packet sizes of the current BANip implementation are chosen adequately.

The focal point of the objectives is aimed at the SUT’s communication function user information transfer and its performance criterion speed. The BANip application protocol mentioned in the objectives will be substituted by the workload generator of the evaluation system as the transport service user. The [ITU1993] description of performance criterion speed can be specified to match our objectives:

*Speed is the performance criterion that describes the (maximum) delivery time that is used to successfully perform a transfer function and the rate at which this transfer is performed. The transfer function realises the transfer of a SE generated by a workload generator at a WG_sap1 (source) to another WG_sap2 (destination). The measurement function “listens” at its MF_sap for arriving SEs destined for WG_sap2.*

The speed criterion is linked to the primary performance parameter delay and derived performance parameters jitter and “goodput”. Figure 4.14 presents the selected performance criterion and the performance parameters.

![Figure 4.14 Performance criterion and parameters of interest.](image)

**delay**

Primary performance parameter which is a measure for the time interval used to transfer a SE from:

- a) WG_sap1 (source) to another WG_sap2 (destination), or
- b) WG_sap1 (source) to another WG_sap2 (destination) and vice versa.
Figure 4.15 presents a time sequence diagram for the transfer of a “user confirmed” SE. The time interval used to transfer the SE is the sum of the request-indication time interval and the response-confirmation time interval; in formulae:

\[
\text{SE}_{\text{user\_confirmed\_delay}} = \text{SE\_delay, upstream} + \text{SE\_delay, downstream}
\]

\[
\text{SE}_{\text{user\_confirmed\_delay}} = (T_{\text{RECV, WG\_sap2/MF\_sap2}} - T_{\text{SEND, WG\_sap1/MF\_sap1}}) + (T_{\text{RECV, WG\_sap1/MF\_sap1}} - T_{\text{SEND, WG\_sap2/MF\_sap2}}).
\]

The time interval for the transmission of a “user unconfirmed” SE can be calculated by the formulae:

**upstream**

\[
\text{SE}_{\text{user\_unconfirmed\_delay}} = \text{SE\_delay, upstream}
\]

\[
\text{SE}_{\text{user\_unconfirmed\_delay}} = (T_{\text{RECV, WG\_sap2/MF\_sap2}} - T_{\text{SEND, WG\_sap1/MF\_sap1}})
\]

**downstream**

\[
\text{SE}_{\text{user\_unconfirmed\_delay}} = \text{SE\_delay, downstream}
\]

\[
\text{SE}_{\text{user\_unconfirmed\_delay}} = (T_{\text{RECV, WG\_sap1/MF\_sap1}} - T_{\text{SEND, WG\_sap2/MF\_sap2}})
\]

**jitter**

Derived performance parameter, which is a measure for the variation in delay.

**“goodput”**

Derived performance measure, determined by the observed rate at which a particular SE is being transferred from WG\_sap to MF\_sap and multiplied by the size of the data carrying component (payload) of this SE.
4.6 System Parameters of Influence

Recall the description of the workload generator at the end of section 4.4. We introduced the notion of “system parameters” being related to instantiations of a SoD and SUT. This section describes a special set of system parameters which may influence the outcome of the SUT performance measurements. This set is identified as “system parameters of influence”. The concept of a SoD and SUT instantiation will be described and related to the “system parameters of influence”.

System parameters of influence

These set of system parameters influence the outcome of the performance measurements iff the values of these parameters are changing. For example, if the capacity of a transport link in the SUT is reduced by 50%, the primary performance parameter delay will change (i.e. increase) for the same volume of information send. According to [Hoek1997] this intuitive approach will not result in a proper identification of all candidate system parameters of influence. Furthermore, a precise definition of a system parameter of influence is needed to clearly understand what these parameters are influencing. [Hoek1997] uses the probability theory to answer this question and concludes with a clear definition of system parameters of influence. We believe the following quotations are crucial for the reader to grasp the meaning of the definition:

1) The performance measure is either a random variable, i.e. a variable having some probability distribution function (PDF) or a probability. In case of a random

---

24 “if and only if”
25 We prefer “primary performance parameter measure”.

variable, e.g. delay, one may think of the performance measure as described by the moments of the PDF of the random variable, such as mean delay and variance of the delay.

2) A value of the measure being the outcome of a single probabilistic experiment is taken as an observation for statistical inference.

3) A sequence of observations is called a ‘sample’. It results from repeating the probabilistic experiment.

A measure is defined as:

A random variable or a probability which quantitatively characterises the behaviour of the SUT [Hoek1997] section 4.6.2.

The system parameter of influence is defined as:

A parameter whose value is a priori considered to affect the probability distribution function (PDF) of the measure (significantly).

This definition implies a good (as possible) understanding of the SoD and SUT (sections 4.2 and 4.3). It also limits the set of parameters to those that clearly (i.e. significantly) affect the probability distribution function. Identification of candidate system parameters of influence pertaining to SE is based on Table 10 in [Hoek1997] is presented in Figure 4.16:

<table>
<thead>
<tr>
<th>Parameter class</th>
<th>Parameter sub-class</th>
<th>Candidate system parameters of influence</th>
</tr>
</thead>
<tbody>
<tr>
<td>system description related parameters</td>
<td>SUT structure</td>
<td>computer system&lt;&gt;intra comm. system&lt;&gt;UMTS terminal&lt;&gt;V3GWL network&lt;&gt;APN&lt;sup&gt;26&lt;/sup&gt;</td>
</tr>
<tr>
<td>SUT internal behaviour</td>
<td>IP service + PPP service Computer system&lt;&gt;Intra comm.. service&lt;&gt;PPP service UMTS terminal&lt;&gt;V3GWL service&lt;&gt;APN IP service (incl. routing function)</td>
<td></td>
</tr>
<tr>
<td>SUT external behaviour</td>
<td>IP service: a) SP_send(header, payload), b) SP_recv(header, payload), c) TOS field: not used (Best-Effort QoS) d) TTL field: expect probability of being “0” low. e) No options field: no header extension f) MTU size = unknown</td>
<td></td>
</tr>
<tr>
<td>quantitative aspects of the SUT</td>
<td>“upstream” transmission capacity “downstream” transmission capacity buffer capacity N+1 transport service layer: unknown buffer capacity N-1 host-to-network service layer: unknown Max PDU length (MTU): unknown Max number of concurrent service users per UMTS cell</td>
<td></td>
</tr>
<tr>
<td>system usage related parameters</td>
<td>workload</td>
<td>see section 4.7.</td>
</tr>
</tbody>
</table>

Figure 4.16 Identification of candidate system parameters of influence.

<sup>26</sup> More detailed knowledge was not provided by Vodafone (V3GLN + UMTS terminal) or Nokia (UMTS terminal) for commercial reasons.
We also identify SoD candidate system parameters of influence that resides in the transport service and in the non-SUT part of the SoD (e.g. Internet and UTnet in Figure 4.6). According to the [Hoek1997] approach used to identify the system (i.e.SUT) parameters of influence, these SoD parameters should not be part of the selected system parameters of influence. We acknowledge the fact that mixing SoD and SUT parameters of influence is not in line with the approach; however, we do combine them because the identified SoD may influence the performance measurements of the SUT.

The candidate system parameters of influence in Figure 4.16 are completed with SoD system parameters of influence and analysed to derive a set of “selected system parameters of influence”. Only parameters for which the effects on the performance measurement are investigated or for which the quantitative aspects are known will be selected. Figure 4.17 presents the overview of SoD and SUT system parameters of influence. The “factors and levels” of these parameters are specified in section 4.8.

<table>
<thead>
<tr>
<th>Parameter class</th>
<th>Parameter sub-class</th>
<th>Selected system parameters of influence</th>
</tr>
</thead>
<tbody>
<tr>
<td>system description related parameters</td>
<td>SUT structure</td>
<td>computer system&lt;&gt;intra comm. system&lt;&gt; UMTS terminal&lt;&gt;V3GNL network&lt;&gt;APN</td>
</tr>
<tr>
<td></td>
<td>SoD structure</td>
<td>Computer system &lt;&gt;SUT&lt;&gt;Internet&lt;&gt; UTnet&lt;&gt;Computer system</td>
</tr>
<tr>
<td></td>
<td>SUT internal behaviour</td>
<td>Assume not of influence.</td>
</tr>
<tr>
<td></td>
<td>SoD internal behaviour</td>
<td>Assume not of influence27</td>
</tr>
<tr>
<td></td>
<td>SUT external behaviour</td>
<td>Assume not of influence</td>
</tr>
</tbody>
</table>
| | SoD external behaviour | TCP service:  
a) SP_send(header, payload),  
b) SP_recv(header, payload),  
c) assume TCP specific options are of no influence |
| quantitative aspects of the SUT | “upstream” transmission capacity  
“downstream” transmission capacity  
number of concurrent service users per UMTS cell |
| quantitative aspects of the SoD | application send/receive buffer sizes (above transport service)  
TCP (socket) send/receive buffer sizes |
| system usage related parameters | workload | see section 4.7. |
| | background load | SUT:  
assumed zero for pre commercial phase,  
SoD (Internet + UTnet):  
existing, but assumed to be of no influence (see SoD internal behaviour). |

Figure 4.17 Selected SoD and SUT parameters of influence.

SoD and SUT instance

A service delivered by a system discloses the external behaviour of this system. The internal mechanisms of the system contributing to the service provisioning are hidden. From a service

27 The transport capacity of the Internet and the UTnet part of the SoD is assumed to be significantly larger then the workload generated during the performance evaluation measurements. We do however validate this assumption in Chapter 6 (delay analysis).
user perspective the internal mechanisms or implementation of the system are irrelevant. However, from a performance evaluation perspective the implementation may play a significant role in the quality (e.g. speed) in which the service is delivered.

One implementation of the system may perform better then another. The selected system parameters of influence are considered to be parameters that modify the implementation of a particular system. Different combinations of system parameters of influence create multiple implementations of a system that delivers the same type of service with (anticipated) different service quality level. Each implementation of a SoD and SUT is considered to represent a particular case or instance. For example, Figure 4.18-A presents an instance of the SoD structure containing the SUT, the Internet and the UTnet. Figure 4.18-B shows an alternative instance containing the SUT and the UTnet. Section 5.1.1 presents all the different SoD and SUT instances that are considered to be used in the overall performance evaluation activity and will be linked to the separate performance measurements.

4.7 Workload and System Parameters

Workload parameters

Figure 4.17 presents workload as a “system usage related parameter of influence”. Workload parameters are therefore parameters for which the effects on the performance measurement can be investigated. In [Jain1991] section 6.1 workload parameters are defined as:

28 In our performance measurements the SoD and SUT are tightly coupled acting as one instance.
Measured quantities, service requests, or resource demands, which are used to model or characterise the workload.

If workload parameters characterise the workload, then they specify the parameters and rate of the SE’s generated by the workload generator. Recall, the workload generator of our evaluation system will provide multiple executions of SUT related SE’s. These parameterised SE’s are typically generated as a sequence of SE’s to derive a statistical sound measure of a primary performance parameter (in our case delay).

Base on the second performance evaluation objective (section 4.1), the workload parameters selected are size and rate. These workload parameters will disclose the “upstream” and “downstream” behaviour of the SUT for different sizes and rates of SE’s; hence, meeting the first objective. A workload parameter which is not easy to identify from the objectives, is the number of workload generator instances (i.e. service users). The number parameter discloses the behaviour of the SUT for concurrent operating workload generators in one UMTS cell. It instruments a scalability performance measurement of the SUT.

Section 2.4 described to correlation between the real world domain of the BANip and the conceptual domain of service elements. Recall that the BANip was characterised as an unconfirmed service; the application protocol sends sequences of messages to the transport service without waiting for an acknowledgement from its receiving protocol peer. Therefore, the workload generator has to implement an unconfirmed service to characterise the BANip message sequences correctly. Hence, we consider the unconfirmed service as a workload parameter.

Note: Because previous implementations of the BANip were based on a confirmed service concept, we also include the “confirmed service” as a workload parameter.

Resume, the specified workload parameters are:

size, rate, number, unconfirmed service and confirmed service.

Workload Parameters and Values

Workload parameters need to receive a value to have a meaning in the execution of performance measurements. In [Jain1991] section 2.2 the activity of assigning values to the workload parameters is described as: assigning levels (i.e. values) to factors (i.e. parameters to be varied). We use “factors” and “workload parameters” as the same concept; the same applies to “levels” and “values”. For every workload parameter we select the values used during the performance measurements.

size

In order to characterise the “upstream” and “downstream” behaviour of the SUT for different sizes of this workload parameter, we select two different ranges of SE sizes:

29 SE’s per second.
“upstream”: 174 bytes up to 8192 bytes  
“downstream”: 174 bytes up to 48208 bytes

The increment of the “upstream” and “downstream” SE sizes is chosen such that the quantitative aspects of the SUT are disclosed. The 20 x 20 matrix presented in Figure 4.19 presents an overview of the selected values for the size workload parameter. The rows indicate sizes for “upstream” measurements and the columns indicate sizes for “downstream” measurements. The number “500” in the top left corner indicates the number of observations per measurement (section 5.2.1). Because of the large number of observations (20x20x500) needed to execute this performance measurement, the 20 x 20 matrix is only executed for one particular instance of the SoD/SUT and one “workload instance”. We will refer to this particular performance measurement as the “benchmark”. A smaller 8 x8 matrix (Figure 4.20) will be used for different instances of the SoD/SUT. The results of these measurements can be correlated to the “benchmark”. The 20 x 20 matrix and the 8 x 8 matrix will only be used in combination with the confirmed service workload parameter.

In order to characterise the “upstream” behaviour of the SUT for different SE rates, we select only one SE sizes that is assumed to be the MTU of the SUT: 524 bytes. The generated rate of the SE offered to the SUT’s SAP is calculated based on a percentage of the (theoretical) maximum “upstream” capacity of the SUT. This percentage is known as the saturation factor of the “upstream” link. We use 10 different saturation factors: 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.5 and 2.0. The increments of the factors are chosen such that the behaviour of the SUT is disclosed for “normal”, “medium to high” and “overloaded” situations.

The workload generator will generate a sequence of the specified SE for a time interval of 40 seconds\footnote{In the MobiHealth system supports the use of Evaluation Data Records (EDR). Each record represents the gross number (before data compression) of bytes forwarded by a MBU to the transmission systems SAP over a time period of 20 seconds. We wanted to be sure that the rate measurements could be compared to the EDR data, therefore a double time period of 40 seconds was used.}. The number of generated SE’s will match the calculated rate and the SE’s are evenly distributed over a time span of one second. The workload generator is not anticipating any acknowledgement from a receiving entity. Hence, the workload generator is anticipating an unconfirmed service from the SUT.
The **number** of workload generator instances (i.e. service users) discloses the behaviour of the SUT for concurrent operating workload generators in one UMTS cell. The numbers chosen are: 1, 5 and 10. The 8 x 8 matrix is used to specify the size of the generated SE’s and the **confirmed service** workload parameter is used as a service type.
unconfirmed service and confirmed service

These workload parameters are only used in conjunction with the size, rate and number parameters (see above).

Figure 4.21 presents a resume of the workload parameters and their values.

<table>
<thead>
<tr>
<th>Parameter class</th>
<th>Parameter sub-class</th>
<th>Workload parameters values</th>
</tr>
</thead>
<tbody>
<tr>
<td>system usage</td>
<td>workload</td>
<td>size: 20 x20 matrix, 8 x 8 matrix and 524 Bytes</td>
</tr>
<tr>
<td>related parameters</td>
<td></td>
<td>saturation factor (calculated rate): 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.5 and 2.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number: 1, 5 and 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>service type: “confirmed service” and “unconfirmed service”</td>
</tr>
</tbody>
</table>

Figure 4.21 Workload parameter values.

System parameters

System parameters are “system parameters of influence” which are classified as system description related parameters. For every performance measurement of a SoD/SUT instantiation these parameters must not change. This creates the possibility to compare the performance measurements of different SoD/SUT instantiations and study the effect of a system parameter of influence. This approach can be used to optimize the performance (e.g. decrease of delay) of the SUT. Figure 4.16 specified four sub-classes of system parameters of influence:

- **SUT structure**
- **SoD structure**
- **quantitative aspects of the SUT**
- **quantitative aspects of the SoD**

For each sub-class different system parameters can be specified.

**SUT structure**

The SUT structure has a number of sub-systems which are considered to be system parameters of influence of special interest. These parameters are related to the host-to-network service part of the SUT (Figure 4.7). The influence of the “UMTS terminal” system parameter on this service is measured.
A computer system in combination with a UMTS terminal is used to provide the IP service at the SUT’s SAP. Different physical media can be used to realise a communication link between the two devices. The system parameter “intra comm. system” is used to measure the effect of using different intra communication systems between a computer system and a UMTS terminal. Closely related to the “intra comm. system” parameter is the “computer system” parameter. The influence of this parameter will be measured for three different types.

The last system parameter of influence that will be studied is the APN of the SUT. The reasoning behind this choice is related to the SoD structure (see below).

SoD structure

The SoD structure consists of three sub-systems: SUT, Internet and the UTnet. The “Internet” and the “UTnet” are considered to be system parameters of influence. The mechanism to select different SoD structures (e.g. “Internet<>UTNet” and “UTnet”) is realized by means of the SUT APN parameter.

quantitative aspects of the SUT and quantitative aspects of the SoD

As discussed in section 4.3.2, the SUT provides a transport service with a capacity that is correlated to the “upstream” and “downstream” volume and rate of the IP datagrams offered at the SUT SAP’s. The system parameters of influence responsible for the change in capacity are controlled by Vodafone.

System Parameters and Values

The assignment of values to the system parameters is presented in Figure 4.22. We assume theoretical values for the quantitative aspects of the SUT and quantitative aspects of the SoD parameters.
<table>
<thead>
<tr>
<th>Parameter class</th>
<th>Parameter sub-class</th>
<th>System parameters values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUT structure</strong></td>
<td>computer system:</td>
<td>notebook, pc1, pc2 and iPaq</td>
</tr>
<tr>
<td></td>
<td>intra comm. system:</td>
<td>USB, Bluetooth and PCMCIA</td>
</tr>
<tr>
<td></td>
<td>UMTS terminal:</td>
<td>Nokia 6650 and PCcard</td>
</tr>
<tr>
<td></td>
<td>APN:</td>
<td>“utwente.nl” and “web.vodafone.nl”</td>
</tr>
<tr>
<td><strong>SoD structure</strong></td>
<td>computer system:</td>
<td>notebook, pc1, pc2 and iPaq</td>
</tr>
<tr>
<td></td>
<td>Internet:</td>
<td>“present” and “not present”</td>
</tr>
<tr>
<td></td>
<td>UUnet:</td>
<td>“present” and “not present”</td>
</tr>
<tr>
<td><strong>quantitative aspects of the SUT</strong></td>
<td>“upstream” transmission capacity:</td>
<td>0-16 Kbps (common bearer) and 16-64 Kbps (dedicated bearer)</td>
</tr>
<tr>
<td></td>
<td>“downstream” transmission capacity:</td>
<td>0-16 Kbps (common bearer), 16-64 Kbps (dedicated bearer 1), 64-128 Kbps (dedicated bearer 2) and 128-384 Kbps (dedicated bearer 3).</td>
</tr>
<tr>
<td>number of concurrent service users per UMTS cell:</td>
<td>1, 5 and 10</td>
<td></td>
</tr>
<tr>
<td><strong>quantitative aspects of the SoD</strong></td>
<td>application buffer sizes (above transport service):</td>
<td>send - 32 KByte and 64 KByte receive - 32 KByte and 64 KByte</td>
</tr>
<tr>
<td></td>
<td>TCP (socket) buffer sizes:</td>
<td>send - 32 KByte and 64 KByte receive - 32 KByte and 64 KByte</td>
</tr>
</tbody>
</table>

Figure 4.22 System parameter values.
5 Performance Measurements

This chapter will focus on the second phase of the measurement methodology presented in section 3.2. It covers step 6 and 7 of the methodology (Figure 5.1).

1. State the Goals and System Definition
2. List Services and their Outcomes
3. Select Performance Criteria (i.e. Metrics)
4. List System and Workload Parameters
5. Select Factors and their Levels
6. Select the Workload
7. Design and Execute the Experiments
8. Analyze, Evaluate and Interpret the Data
9. Present the Results

Figure 5.1 Measurement Methodology Phase 2.

In this chapter the execution of the reproducible performance measurements on the real-world instances of the SoD are presented. The goal of measurements was to obtain the estimates for the performance measures of interest; particularly estimates for the delay performance measures as identified in section 4.5.

Particularly the entire chapter is dedicated for activity 7 (Figure 5.1), the result on the workload selection (activity 6) is given in section 5.2.1.

The first activity in designing of experiments is the selection of SoD instances on which the measurements are executed. There are different SoD instances based on the system parameters identified in section 4.7. The results of SoD selection are presented in section 5.1.1.

Recalling Figure 4.12, besides the SoD/SUT the evaluation system component is identified. The evaluation system needs to be instrumented before the measurements execution can start. Firstly, the evaluation system requirements are identified, and then the functional model of system to be implemented is given. Furthermore most important implementation details, proving the particular functionality provision, are given as well. The result of these activities is presented in the section 5.1.2. The following section (5.1.3) deals with the evaluation systems’ synchronization. Evaluation system instance has to be synchronized in time (system clock). Moreover the functionality of the evaluation system needs to be managed.

In the section 5.2 the interactions between the SoD, SUT and evaluation system are presented in the measurement setup(s). After the selection of combined workload and system parameters the experiments are defined (section 5.2.1). The experiment’s execution (“where, when and who”) is presented in section 5.2.2.
In the section 5.3 the measurements (raw) data evaluation phase is presented, which consists of derivation of the statistics regarding the data and visualization of raw results. Based on the raw data evaluation, some problems in measurement procedure were identified and eliminated. In section 5.3 these problems and possible solutions to them are highlighted.

5.1 Measurements Instrumentation

In this section the description of the system being measured (section 5.1.1) and then instrumentation of the evaluation system (section 5.1.2) are presented. In section 5.1.3 the solution for the synchronization of the evaluation systems’ components is presented.

5.1.1 SoD and SUT instances

Recall the Figure 4.6, the system which performance characteristics are going to be studied is a System of Discourse (SoD) consisting of the components: SUT (VTT3G), Internet and the UTnet. The goal of this section is to show the identification process of the SoD and especially the (embedded) SUT, and the points at which evaluation system will be attached to the SoD.

As it has been mentioned in section 4.4 the instrumentation at the SUT_SAPs was not possible. Insights of the VTT3G networks (e.g. like APN) were not accessible for the measurements. Therefore, the ingress and egress SAP of SoD instrumented for measurements as ingress and egress SAP of the SUT; the ingress and egress point of the SoD has been implemented as the ingress and egress SAPs of the transport service offered by the transport system. Identified SAPs are indicated in Figure 4.6.

After the SAPs to the SoD (SUT) have been identified, the attention has been directed to the SoD insights’ implementation details, which depend on the system parameters configuration. Recalling the Figure 4.18, SoD consists of two or three independent sub-systems. The existence of the Internet subsystem in SoD relied from the chosen APN in the VTT3G network. APN web.vodafone.nl provides access to the public data networks and it’s associated with the Internet access. APN utwente.nl provides access from the VG3NL to the (private) data network of University of Twente. APN is a system parameter pointed in Figure 4.22. There can be a number of SoDs instances indicated, depending on the other system parameters’ configuration in the SoD. Due to e.g. time constraints, only a selected number of combinations of system parameters were taken into account. The selection criteria of the SoD/SUT instances of interest were:

1) resource equipment constraint: there was limited number of equipment available e.g. only one notebook (nb) and two PCs (pc1, pc2). Moreover some of the combinations of the system parameters (hardware platforms and technologies) were not feasible to be set. For example the iPAQ platform does not offer: a) the USB interface or b) the PCMCIA interface for the Vodafone Mobile Connect UMTS/GPRS Card due to the lack of drivers for the platform’s OS.
2) priority criteria: some of the SoD instantiations were of higher importance to be studied than others due to possible indication (by elimination) of contribution of
Internet and UTnet subsystems in the delay performance characteristics of SoD (i.e. SoD instance is setup such the SUT performance characteristics can be derived) or their importance for the MobiHealth project (e.g. ipaq platform used currently as an MBU in BAN).

As the result of the selection of the instances of SoD/SUT, the following table has been obtained as presented at Figure 5.2.

<table>
<thead>
<tr>
<th>computer systems</th>
<th>nb, pc1</th>
<th>nb, pc1</th>
<th>nb, pc1</th>
<th>nb, pc1</th>
<th>nb, pc1, pc2</th>
<th>ipaq, pc1</th>
<th>nb, pc1</th>
</tr>
</thead>
<tbody>
<tr>
<td>intra comm.</td>
<td>usb</td>
<td>usb</td>
<td>usb</td>
<td>bluetooth</td>
<td>PCMCIA</td>
<td>usb</td>
<td>bluetooth</td>
</tr>
<tr>
<td>UMTS terminal</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>PC Card</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
</tr>
<tr>
<td>APN</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>buff. sizes: appl.socket</td>
<td>64.64</td>
<td>32.64</td>
<td>32.32</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
</tr>
<tr>
<td>SoD/SUT instance</td>
<td>SoD_1</td>
<td>SoD_2</td>
<td>SoD_3</td>
<td>SoD_4</td>
<td>SoD_5</td>
<td>SoD_6</td>
<td>SoD_7</td>
</tr>
</tbody>
</table>

Figure 5.2 SoD/SUT system parameters selection.

The meaning of particular setups is as follows. Computer system (i.e. evaluation system called server) at the ingress point of SoD: nb or ipaq uses intra communication: usb, bluetooth or PCMCIA to access the UMTS terminal: Nokia 6650 or PC card and through this terminal to access the VTT3G network. APN used determines then the internal infrastructure of SoD. At the SoD egress point there is a computer system: pc1 or pc2 (i.e. evaluation system called client). All computer systems used in SoD setup have the same buffer sizes (as indicated in table).

Possible implementations of SoD are explained in the following sections. The explanation will start from instance SoD_8 as available commercially. Then the explanation of SoD instances 1 to 7 will follow. All possible instances are given at Figure 5.3.

SoD instance **SoD_8** web.vodafone.nl

The commercial implementation of SoD is available for all VTT3G users. As indicated at Figure 5.3 the computer systems (i.e. evaluation systems) accessing the SoD are configured as in Figure 5.2 for the SoD_8. At the ingress point of VTT3G there is a notebook with USB interface to Nokia 6650 terminal and thus the UMTS interface to VTT3G network. If notebook acts as a data
generator; generated data is transported thru the VTT3G network, Internet to the data sink placed at the UTnet. Both computer systems have assigned 64 KByte application and TCP buffers size.

V3GNL has the following features:

a) public/shared resource
b) “best effort” service
c) asymmetrical, multilevel (with capability to switch between contiguous levels) link capacity:
   upstream capacity  16 or 64 Kbps
downstream capacity  16 or 64/128/384 Kbps
d) IP address assignment for the system attached at the network V3GNL ingress point: the “private” Class A IP address, for example: 10.77.86.235; system is not reachable from the outside of the V3GNL network.

The Internet sub-system in the SoD provides an IP service at the boundaries. This service is implemented by means of IP router interfaces. Inside the Internet, there can be networks of different providers distinguished, for example based on the one possible instance of the Internet presented at Figure 5.4, UUNet and SURFnet ISPs core networks were identified.

<table>
<thead>
<tr>
<th>Trace route</th>
<th>Node Address</th>
<th>RTT [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>simplon (130.89.10.1)</td>
<td>0.559 / 0.305 / 0.389</td>
</tr>
<tr>
<td>2</td>
<td>1F-cc-routing.utwente.nl (130.89.254.113)</td>
<td>0.353 / 0.347 / 0.309</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>so-0-2-0.TK2.AMS2.ALTER.NET (146.188.3.217)</td>
<td>5.153 / 5.181 / 5.164</td>
</tr>
<tr>
<td>10</td>
<td>so-6-0-0.XK2.AMS5.ALTER.NET (146.188.8.89)</td>
<td>5.160 / 5.206 / 5.157</td>
</tr>
<tr>
<td>11</td>
<td>so-0-0-0.CR2.AMS7.ALTER.NET (212.136.176.102)</td>
<td>5.311 / 5.290 / 5.286</td>
</tr>
<tr>
<td>12</td>
<td>312.XMG-0-0.GM1.AMS7.ALTER.NET (212.136.176.138)</td>
<td>5.614 / 5.612 / 5.507</td>
</tr>
<tr>
<td>13</td>
<td>Vodafone-gw.customer.ALTER.NET (146.188.37.130)</td>
<td>7.522 / 6.031 / 6.113</td>
</tr>
</tbody>
</table>

Figure 5.4 Path from the UTnet to the V3GNL through Internet.

The UTnet has an IP-based interface. UTnet has the direct connection to the Surfnet ISP core network (1Gb link capacity). UTnet a fast (100 Mbps) local area network, with the IP addresses range: 130.89.0.0 - 130.89.255.255.

SoD instance **SoD_1**

The dedicated implementation of SoD is available only for users associated to the UT. Comparing to the web.vodafone.nl instance, this SoD does not include the Internet sub-system. Due to the characteristics of utwente.nl SoD, it is used in different configurations in majority of measurements. The computer systems accessing the SoD are configured as indicated at the Figure 5.2 for SoD_1. At the ingress point of V3GNL there is a notebook computer system with

---

31 Used tracert program under the Windows XP OS
32 RTT between: the node which initiated the trace and the receiving node for the three separate journeys. RTTs are not cumulative along the trace.
33 APN web.vodafone.nl, Sending node: notebook, receiving: Vodafone’s infrastructure node: 62.140.137.62
USB interface to Nokia 6650 terminal and thus the UMTS interface to V3GNL network. If
notebook acts as a data generator; generated data is transported thru the V3GNL network to the
data sink placed at the UTnet (computer system pc1). Both computer systems have assigned
64kByte application and TCP buffers size.
For the period of 8\textsuperscript{th} July 2003 – 1\textsuperscript{st} April 2004 a leased line\textsuperscript{34} was connected by the VTT3G to the UTnet (see Appendix 2 for schematic overview). A leased line may be a telephone line that has been leased for private use. Leased line has been rent from the telephone message carrier KPN.

Features of the leased line are:

a) dedicated resource
b) QoS provision; leased line tend to only suffer from physical rather than operational failures
c) symmetrical link capacity: upstream/downstream capacity 2048 Mbps (E1).

Due to the direct connection between the VTT3G and UTnet, the IP address assignment to the system attached to the VTT3G is different from the previous case. Namely, each system attached to the VTT3G via \textit{utwente.nl} APN gets the Class B IP address in range of the UTnet addresses, for example: 130.89.146.90. That means system is reachable from the outside of the VTT3G network.

Based on the (one possible) instance of the route presented at Figure 5.6, UTnet and Vodafone’s routers at the boundary of the UTnet were identified.

<table>
<thead>
<tr>
<th>Trace route</th>
<th>RTT [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 wurzen.cs.utwente.nl</td>
<td>&lt;1 &lt;1 &lt;1</td>
</tr>
<tr>
<td>2 if-cs.routing.utwente.nl</td>
<td>&lt;1 &lt;1 &lt;1</td>
</tr>
<tr>
<td>UTnet / Vodafone</td>
<td>1 1 1</td>
</tr>
<tr>
<td>3 130.89.248.145</td>
<td>6 5 5</td>
</tr>
<tr>
<td>4 130.89.248.141</td>
<td>6 6 6</td>
</tr>
<tr>
<td>5 130.89.248.129</td>
<td>346 335 345</td>
</tr>
<tr>
<td>6 130.89.146.90</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.6 Path instance from the UTnet to the V3GNL through the leased line\textsuperscript{35}.

As mentioned in previous section, the UTnet has IP-based interface. UTnet has the direct connection to the Surfnet ISP core network (1Gb link capacity). UTnet a fast (100 Mbps) local area network, with the IP addresses range: 130.89.0.0 - 130.89.255.255.

SoD instances \textbf{2 to7} - alternatives of \textbf{SoD \_1}

In the section above the case of \textit{utwente.nl} APN with parameters defined as for SoD \_1 has been presented. At Figure 5.2, there are more SoD instances with APN \textit{utwente.nl} and different remaining system parameters considered. Particularly, there are SoD \_2 to SoD \_7 defined, which differ in the configuration of the computer systems accessing the SoD. The instances of SoD: SoD \_2 to SoD \_7 are presented at Figures 5.3 as well.

For SoD \_2 there is a notebook with USB interface to Nokia 6650 terminal and thus the UMTS interface to V3GNL network. If notebook acts as a data generator; generated data is transported

\textsuperscript{34} A private communications channel leased from a common carrier.

\textsuperscript{35} APN \textit{utwente.nl}, Sending node: notebook, receiving: UMTS node: 130.89.146.90
through the VTT3G network to the data sink placed at the UTnet (computer system pc1). Both computer systems have assigned 32kByte application buffer size and 64kByte TCP buffers size.

For SoD_3 there is a notebook with USB interface to Nokia 6650 terminal and thus the UMTS interface to V3GNL network. If notebook acts as a data generator; generated data is transported through the V3GNL network to the data sink placed at the UTnet (computer system pc1). Both computer systems have assigned 32kByte application and TCP buffers size.

For SoD_4, there is a notebook with bluetooth interface to Nokia 6650 terminal and thus the UMTS interface to V3GNL network. If notebook acts as a data generator; generated data is transported thru the V3GNL network to the data sink placed at the UTnet (computer system pc1). Both computer systems have assigned 64kByte application and TCP buffers size.

For SoD_5, there is a notebook with PCMCIA interface to Vodafone Mobile Connect UMTS/GPRS PC card terminal and thus the UMTS interface to V3GNL network. If notebook acts as a data generator; generated data is transported thru the V3GNL network to the data sink placed at the UTnet (computer system pc1). Both computer systems have assigned 64kByte application and TCP buffers size.

The SoD_6, is an interesting case for measurements. It contains the notebook computer system with bluetooth interface to Nokia 6650 terminal and UMTS interface to V3GNL, pc2 in between the V3GNL and UTnet and pc1 at the UTnet. If notebook acts as a data generator; generated data is transported thru the V3GNL network to the data sink pc1, where data is altered such pc2 acts as a data generator to the data sink placed at the UTnet (computer system pc1). All three computer systems involved have assigned 64kByte application and TCP buffers size. This SoD case is an implementation of the service view as presented at Figure 4.12-B. The PC2 is placed in between the Vodafone’s router and the UTnet router (thus creating the Virtual LAN) in order to have a measurement point as close as it is possible to the egress point of the V3GNL. This configuration on one hand allows for indication of the contribution of the UTnet subsystem in the delay performance characteristics of the SoD and on the other hand, implies the split of SoD_6 into two sub-systems: first contains the SUT entirely, and the second contains the UTNet. According to the presented measurement methodology the SoD should be explicitly the one containing the SUT (let’s name it SoD_6’). But regarding the sub-system containing the UTnet, it is also treated according to our measurement methodology and it is seen as another SoD (i.e. SoD_6’’) containing the SUT which is then UTnet. During the measurements activity both SoDs are instrumented. Measurements at the ingress and egress points of this SoD_6’ and SoD_6’’ are taken. Performance characteristics of the VTT3G and UTNet can be derived separately. Performance measures of interest are the same for both SoDs. Parameters of influence in the SoD_6’’ are different, but this consideration is beyond of the scope of this project.

For SoD_7, there is an ipaq computer system with bluetooth interface to Nokia 6650 terminal and thus the UMTS interface to VTT3G network. iPAQ acts as a data generator; generated data is transported thru the VTT3G network to the data sink placed at the UTnet (computer system pc1). Both computer systems have assigned 64kByte application and TCP buffers size.
5.1.2 Evaluation System Instrumentation

In this section the process of evaluation system instrumentation is presented. First step was the identification of the evaluation system requirements, then the decisions regarding the method of instrumenting the system, were taken. Functional model is used to explain the evaluation system’s service. Finally in this section the implementation details of the evaluation system are given.

Requirements for the evaluation system instrumentation

When attempting the evaluation system implementation phase, there were particular instrumentation requirements identified and they are as follows:

1) no cooperation between the components of the evaluation system: the workload generator and the measurement system

2) control of the interactions between the workload generator and SoD/SUT

3) as little as possible influence of the measurement system on the behaviour of SoD/SUT

4) workload and system parameters as provided by the evaluation service end-user

5) automated generation of different workload parameters values

6) evaluation system clock’s synchronization functionality

7) automated monitoring of events and correlation of measurements to workload parameters by the measurement system

8) measurement system’s capability of measurement data storage for later retrieval

9) evaluation system’s error handling functionality

Implementation of the evaluation system started by the survey of existing workload generators or measurement tools in order to check if they meet the evaluation system’s requirements. It seemed not very likely that there was an existing workload generator and distributed measurement system available that matched these requirements, for example:

1) *tcpdump* and *ttcp* can act as a workload generators but the automated generation of different workload parameter values is not trivial to instrument (requirement no. 6).

2) Using *ethereal* as a component for a measurement system creates serious problem with correlation of measurement data with the workload parameters problem

---

36 Workload generator interacts with the SUT i.e. influences the behaviour of the SUT, but measurement system should not
(requirement no. 8). What is more important, this tool is invasive (inside SUT) which is not allowed in the presented methodology (requirement no. 4).

Regarding the requirements for the evaluation system and results of the survey on available tools, the decision on the development and implementation of the evaluation system by ourselves, has been taken. The next step was to identify what, (i.e. which functionality) and how needs to be implemented.

Recall section 4.4, the evaluation system consists of the workload generator function and the measurement system function. Evaluation system interacts with the SoD at the SAP on SoD. It provides stimuli to it by means of the workload and it measures the reaction to those stimuli by means of the measurement function. The abstract service view on the evaluation system is as shown on Figure 4.10.

The high-level view functional model of the evaluation system delivering the evaluation service is presented on Figure 5.7. There are three functional blocks distinguished: SoD, evaluation service and end-user functions. SoD delivers the transport service and its functionality has been presented in details in chapter 4.

Evaluation service primary function comprises the functionality needed to provisioning of the evaluation service (to its end-user). Evaluation service is delivered by means of evaluating of the behaviour of SoD functionality; evaluation service influences the behaviour of SoD by generation of stimuli to it and measure the reaction to this stimuli. Evaluation service is also able to react to stimuli from SoD.

The secondary function of the evaluation service is the provision of the end-user functions that facilitate the interaction between the end-user and the service, and therefore deal with:

1) the evaluation service requests’ handling; end-user provides particular workload and system parameters (i.e. user control data)

2) end-user notification on the state of the evaluation service execution (e.g. completed, interrupted due to the error, etc.)

![Figure 5.7 Evaluation service functional model.](image-url)
High-level functional view on the evaluation system, results in the following evaluation system implementation requirements:

1) evaluation system needs to comprise the functionality needed to evaluation service provision to the end-user

2) due to the nature of the service provided by SUT (IP service) the evaluation system needs to be running at an IP entity and it needs to interact with SoD by the communication means e.g. IP-based network infrastructure.

3) evaluation system needs to have an interaction means with the service end-user

4) evaluation system implementation must be done with usage of standard hardware/software platforms

5) evaluation system must be portable (interoperable) amongst the platforms specified in requirement d)

Moreover, the resource constraints taken into account while implementing the evaluation system were:

6) equipment constraint: evaluation system implementation must be done with usage of available (and known for executors) hardware/software platforms

7) time constraint: evaluation system needs to comprise the functionality needed to evaluation service provision to the end-user; no additional requirements like e.g. system’s robustness are implemented.

Regarding the given system requirements, and also the equipment constraint (i.e. lack of specialized hardware platform for the evaluation system purposes), the decision on development of the entirely software tool, which is portable amongst different hardware platforms (“write once, run anywhere”), has been taken. Therefore the evaluation system has been implemented in the Java programming language\(^{37}\). Particularly, the system has been implemented as a java application; i.e. Java program that is run standalone\(^{38}\) at the platform [TechWeb]. Developed java application is a mean of which the evaluation system provides the evaluation service.

Java application implements all the functionality needed to evaluation service provision to the end-user (see the following sections for detailed functionalities explanation). Particularly, the java application is running on the networked machine (IP entity), such the interaction between the evaluation system and service provided by SoD is facilitated by means of the (java application) sockets\(^{39}\). Regarding the fact, that evaluation system examines only a reliable transport service

\(^{37}\) Java is an object-oriented programming language based on the virtual machine (JVM or CVM) that offers the same interface independently of the hardware or operating system. The concept of Java programs is: “write once, run anywhere”.

\(^{38}\) The Java Virtual Machine at the machine where the Java application is running is interpreting the instructions.

\(^{39}\) Socket is a receptacle that receives a plug [Webster]
case provided by SoD (recall chapter 4), there is only a reliable socket implemented as a SAP into the SoD.

Reliable transport service provided by SoD implies the usage of TCP protocol at the transport layer, thus the socket implemented in java application is called a TCP socket. Summarizing, the evaluation system is implemented as an application running on top of the TCP/IP protocol stack (i.e. reliable transport service and IP network layer protocol) and it communicates with SoD by means of the reliable socket: influences the behaviour of SoD by generation of stimulus to the socket and measure the reaction to the stimulus from socket.

End-user can interact with the evaluation system by means of keyboard and monitor attached to the machine on which the instance of evaluation system is running. End-user can set the workload and system parameters. There is no GUI implemented, the end-user is assumed to have a knowledge on java programming language, such he/she can change the parameters provided to the evaluation system (i.e. by means of interacting with the java application).

In the section below the details on functionality provided by the evaluation system implemented as a java application are given. Developed java application is a mean of which the evaluation system provides the evaluation service.

**Evaluation System Functionality**

The evaluation system is considered to provide evaluation service functionality as shown on Figure 5.7. In this section the evaluation service provision details are presented. The structure of the section is based on the process of the service ‘step-wise refinement’. It presents a high level functional design model of the evaluation service and introduces the main functional building blocks. Each individual functional building block is then described in general terms and refined into a more detailed design model. The level of refinement is chosen such that a high level design of the individual blocks, and as a consequence the evaluation service, is described.

**Evaluation service**

The *evaluation service* functional block comprises the functionality required for the interaction with: SoD and evaluation service end-user. There are included: the *service logic, workload generator* and *measurement function* as shown at Figure 5.8.

The *service logic* function holds all functionality needed to initialize and then control the external observable behaviour of the *evaluation service*. Based on the parameters provided by the end-user (i.e. workload and system parameters), the *workload generator* and *measurement functions* are initialized (by means of the control data flow).

The *workload generator* comprises the functionality of providing the stimulus to the SoD and reacting to the stimulus from the SoD. The stimulus is generated according to the workload parameters.
The measurement function comprises the functionality of measuring of the SoD reaction to the generated stimulus. The functionality of measurement function is provided based on the control data input set (i.e. system parameters).

From the workload generator and measurement function’s point of view, the stimulus to and from SoD is a message. Whenever the workload generator or measurement function encounter the error while providing their service, the error notification is raised to the service logic functional block. Service logic block contains the error handling functionality which includes the end-user functions notification.

![Figure 5.8 Evaluation service refinement.](image)

A. service logic

The service logic functional block holds the responsibility of managing of the evaluation service execution based on the parameters provided in user control data set and also handling of the service execution status. The service logic function contains four functional building blocks (Figure 5.9), which can be grouped into three functionality groups: handling the user control data, handling the service execution status and ensuring the system clock correctness. The system parameter configuration functional block comprises the system parameters settings provided by the end-user. These parameters are set and seen in the evaluation system as a global factors.

The workload generator/measurement function initialization block enables the initialization of the workload generator and measurement function with the adequate workload/system parameters.

Very important role in the evaluation service provisioning plays the time synchronization function - it can provide the automatic system clock synchronization on the scheduled basis. System clock is a critical issue in the evaluation service provision because all the events need to be generated and timestamped in the accurate and reliable manner.

The error handling functional block enables the interception of the erroneous service delivery; such the error notification is generated to the other functional blocks and to the end-user. In case of error, the workload generator is instructed to terminate, measurement function is instructed to save the measurement data so far and then terminate.
B. **workload generator**

The **workload generator** (Figure 5.10) comprises the functionality of providing the message stimulus to the SoD by means of the *send function* and also functionality of reacting to the message from the SoD by means of the *receive function*. Both functional building blocks are provided with the respective control data by the *service logic* function; send function is provided with the generated workload parameters while receive function is provided with e.g. timeout on receive.

There could be two different *send* and *receive function* relations. *Receive function* triggered by the received message can stimulate the *send function* if this relation has been enabled in the receive control data set. This relation is used for implementation of the confirmed service (recall section 4.7). Otherwise, the *send* and *receive* functions provide services independently, and then the unconfirmed service is implemented.

Both functional building blocks can generate the error notification to the *service logic* when a problem while delivering their service is encountered.
Figure 5.10 Workload generator refinement.

(a) send function

Send function provides the message stimulus to the evaluation system (and thus to SoD). It consists of three functional building blocks (Figure 5.11): send control function, send message function and the timer function.

Send control function cumulates all the knowledge on what (i.e. message parameters) and when (i.e. with which rate, triggered by the receive function notification or not) needs to be generated. The send function gets all required knowledge from the workload generator functional block by means of the control data set. The functionality of the send control function compromise the instruction to the send message function on the message construction (with particular workload characteristics) and triggering the messages sending (via send message notification). Workload characteristics can be organized in sets, then the send control function needs to have a functionality of selecting of characteristics such the whole set of characteristics’ values is explored. Moreover, the control data set can consists of indication of the workload characteristics’ repetition number, so the send control function needs to send a message with the same workload characteristics number of times; the counter function for a number of repetitions is performing.

There are two possible scenarios of the send control function execution: instructing the send message function on construction of a message and then a) sending it and instructing of the timer function on timeout to receive notification from the receiver, or b) sending it and instructing of the timer function on time interval before the notification of sending the next message. In case a) the trigger to generate and send a new message is a receive function notification (confirmed service implementation), while in case b) messages are generated and send with a particular rate (unconfirmed service implementation). If in the first case the receive function notification is not received within the particular time interval, the timing error is raised by the timer function.

The functionality of the send message function block comprises the interaction with the workload generator. Based on the send notification, the clock in the timer function can be reset (i.e. like in
a stop-watch). *Send message function* can also encounter communication problems with sending, and then the error notification is generated.

*Timer function* comprises the functionality of timing the events. Particularly it times the intervals between the consecutive sending of messages, but it also times the service time execution i.e. the global maximum time (set by end-user, given in the send control data set) for the system to provide its functionality.

Summarizing, the *send function* block can raise two kinds of errors: communication and timing error to the *workload generator* function.

![Send function refinement](image)

**Figure 5.11 Send function refinement.**

b. **receive function**

*Receive function* fulfils the *workload generator* functionality of reacting to the message from *evaluation system* (stimulus form *SoD*). The *receive function* consists of two functional building blocks as shown on Figure 5.12. When the message from the *workload generator* is received it is processed in the *receive data processing* block in order to check if the communication error occurred. The notification to the *send function* is generated only if there is no communication error.

The *receive function* is timed by the *timer function* block; if there was no message received within the particular time interval (given as a parameter in the control data set) a timing error notification is generated. The clock in the *timer function* is reset (i.e. like in a stop-watch) by the received message notification. *Timer function* times not only the intervals between the consecutive receiving of messages, but also the service time execution i.e. the global maximum.
time (set by end-user, given in the send control data set) for the system to provide its functionality.

Summarizing, the *receive function* can raise the error notification to the *workload generator* function if the communication or timing error have been encountered.

![Figure 5.12 Receive function refinement.](image)

**C measurement function**

The *measurement function* comprises the functionality of measuring of the SoD reaction to the generated stimuli (by the *workload generator* function). Particularly, the *measurement function* consists of three functional blocks as at Figure 5.13.

*Timestamp function* provides timestamp of the message. Then the message and its timestamp are forwarded to the *measurements handler* function, where a further processing of the measurement data is executed based on the control data set provided by the *service logic* function. Set of the processed and managed (e.g. the data order) measurement data is called a *observation* and when it is ready it is forwarded to the *storage function*. *Measurement handler* has a counter function in order to count the observations in set (required number of observations is given in the control data set).

A sample that consists of a sequence of observations (i.e. measurements) is pushed to the *storage function* together with information on the measurement context (i.e. the workload and system parameters). The *storage function* comprises the functionality of storing of the sample together with associating measurement context data in the file. In case when error is encountered by the *measurement handler* or *storage function*, the error notification is raised to the *evaluation service* function. The *storage function* can be triggered to save data at any time (e.g. for a partial sample). This functionality is used in case if the error in service delivery occurs.
The measurement handler comprises all the functionality needed to process the measurement data in order to facilitate the measurement data storage. It contains of the two functional blocks (Figure 5.14): Measurement data validation, measurement data administration.

The data validation function is triggered by the measurement data (message and its timestamp) from the measurement function block. Received message is interpreted (“send” or “receive”) and its parameters are validated within the measurement context gathered in the control data set. Particularly, if the sequence number of the message doesn’t conform the number expected the measurement data is going to be discarded and validation error is raised to the measurement function.

If the packet validation process give the positive outcome, the measurement data observation is forwarded to the measurement data administration function which is responsible for the proper management of the measurement data (i.e. order of: timestamp and then message parameters), such the ordered observation is composed. Administered observation is then collected within the sample.

Data administration function has a counter function in order to count the collected observations. After the sample is collected (or the error indication in control data has been given which pursues the data storage), it is together with the measurement context data are pushed for storage to the measurement function.
Implementation of evaluation system functionality

In the previous section the functionality of the evaluation system developed in java programming language has been presented. In this section the most important implementation details of the java application are highlighted.

In order to implement all the functionality needed for the evaluation service provision, java application has been structured in a particular way. Application consists of classes, which mainly are build for the purpose of implemented system and partially taken from the class libraries. Classes provide functionalities; they have attributes to assign class’ parameters and methods to perform tasks. Recalling Figure 5.7, there are three high-level functions of the evaluation system distinguished: workload generator, measurements function and service logic function (i.e. system parameters holder). The class diagram of java application with indication which classes (and attributes, methods) fulfil particular functionalities of the evaluation system, is presented at Figure 5.15.

Note: The names of classes indicate the server-client paradigm used in the system; this paradigm is implemented for the case when multiple instances of evaluation system are cooperating (explanation is provided further in the section).

System parameters are given by the end-user in two ways. He/she compounds them by means of the system setup i.e. using particular platforms and technologies for SoD instantiation (see section 5.1.1). Additionally he/she can provide changes regarding the system parameters in the application source code. The most important class, which holds all the parameters, is a Client class (see class attributes indicated at Figure 5.15).

The workload parameters are entirely set by the user in the source code. As given in section 4.7, the workload parameters are generated based on the matrix of packet sizes (e.g. matrix 20x20 at Figure 4.19). It is possible for end user to instruct the evaluation system to start the workload generation with characteristics picked up from any cell in matrix indicated by the coordinates: row.column (where row and column are calculated from 0..19). This option is setup by the end user in the MATRIXCOORDINATES attribute.
There are also some administrative (i.e. measurement function related) parameters in the Client class, which can be of interest for the end-user to change. Client class has a supervising functionality for the evaluation system(s) running at one machine. The single delivery of the evaluation service is called test run (or in short: test). The set of consecutive evaluation service deliveries executed by one instance of the evaluation system is called test cycle. End user can instruct the evaluation system instance to execute number of evaluation service deliveries by means of TEST_OPTION_NUMBER attribute at the Client class. This option is particularly used when multiple instances of evaluation systems are interacting. This case is explained at the end of this section.
Figure 5.15 Evaluation system class diagram.
The object of the `ServerControlFunction` class premises functionality of one evaluation system instance: workload generator, measurement function and service logic. Every object of this class has a `serverName` attribute, uniquely identifying the instance of the evaluation system.

The procedure of automated workload generation, as requested by end-user, is executed by the `ConfirmedServiceTest` and `UnconfirmedServiceTest` (i.e. confirmed, unconfirmed test case). `Packet` class contains the functionality of creation of the workload generator unit. Each generated packet has particular workload characteristics (packet size, rate) which are provided by the `MatrixPacketSizeRate` class. Finally, the `Communication` class facilitate the socket communication, i.e. send and receive packet from SoD (see Figure 5.16 for implementation details). Both functions can raise the communication error.

```java
/**
 * Defines methods for communication over the TCP bearer
 * between two networked hosts (that is client & server).
 */
public class Communication {
    // Atribute(s)
    public final static int COMMUNICATION_ERROR = -273;
    public final static int TIMING_ERROR = -272;
    // Constructor(s)
    /**
     * Communication constructor.
     */
    public Communication() {
    }
    // Method(s)
    /**
     * Implements the "request" and "response" SP of connection-oriented service,
     * For the given Packet object the timestamp in ms on completion of the send action
     * or COMMUNICATION_ERROR is set if the send action failed
     * (that is the communication problem occurred).
     *
     * @param   outChannel    channel to write to
     * @param   packet        Packet object, which data are send
     */
    public void send_tcp( PrintWriter outChannel, Packet packet ) {
        long timeStamp = 0;
        try {
            // timestamp the event of sending a PDU by a SP to the LLS
            timeStamp = System.currentTimeMillis();
            outChannel.println( packet.getPDU() );
            outChannel.flush();
        }
        catch ( Exception ex ) {
            System.out.println( "Error [send_tcp]: Can't write to output channel" +
                                ex.toString() );
            packet.setErrorCode( COMMUNICATION_ERROR );
            packet.setTimeStamp( timeStamp );
        }
    }
    /**
     * Implements the "indication" and "confirmation" SP of
     * the connection-oriented service.
     *
     * @param   inChannel             channel to read from
     * @param   testRunStopTime       stop time for the test run
     *
     * @return  a <code>Packet</code> object with updated parameters:
     *          timestamp in ms on completion of the recv action or
     */
    public void send_tcp( PrintWriter outChannel, Packet packet ) {
        long timeStamp = 0;
        try {
            // timestamp the event of sending a PDU by a SP to the LLS
            timeStamp = System.currentTimeMillis();
            outChannel.println( packet.getPDU() );
            outChannel.flush();
        }
        catch ( Exception ex ) {
            System.out.println( "Error [send_tcp]: Can't write to output channel" +
                                ex.toString() );
            packet.setErrorCode( COMMUNICATION_ERROR );
            packet.setTimeStamp( timeStamp );
        }
    }
}
```
public Packet recv_tcp(BufferedReader inChannel) {
    String pdu = new String("" );
    int errorCode = 0;
    long timeStamp = 0;
    try {
        pdu = inChannel.readLine();
        timeStamp = System.currentTimeMillis();
        if ( pdu.length() == 0 ) {
            System.out.println("Error [recv_tcp]: pdu is empty");
            errorCode = COMMUNICATION_ERROR;
            return new Packet(Packet.constructPCI(0, 0, 0), Packet.constructDataSDU(0),
                               errorCode, timeStamp);
        } else {
            return new Packet(pdu.substring(0, (Packet.PCI_COMPONENTS * Packet.PCI_FRAGMENT_SIZE)),
                              pdu.substring(Packet.PCI_COMPONENTS * Packet.PCI_FRAGMENT_SIZE),
                              errorCode, timeStamp);
        }
    } catch (Exception ex) {
        System.out.println("Error [recv_tcp]: general error "+ ex.toString());
        pdu = "0";
        errorCode = COMMUNICATION_ERROR;
        return new Packet(Packet.constructPCI(0, 0, 0), Packet.constructDataSDU(0),
                           errorCode, timeStamp);
    }
}

public boolean confirmedServiceTest(String serverName, long testStartTime,
                                      int currentTestRun, int row, int column) {
    // start test run...
    // construct packet to be send according to the workload parameters
    sendPacketSize = Server.matrixPacketSizeRate.getPacketSize( r );
    packet.constructPacket(Packet.constructPCI( this.packetSeqNo, sendPacketSize, maxPacketRate ),
                            Packet.constructDataSDU(sendPacketSize ), 0, 0 );
    // send packet...
    communication.send_tcp(Server.dataOutToClient, packet );
    // retrieve timestamp of the event
    requestTime = packet.getTimeStamp();
    //request packet successfully send, receive response packet...
    packet = communication.recv_tcp(Server.dataInFromClient );
    // retrieve timestamp of the event
    confirmTime = packet.getTimeStamp();
}

private boolean upLinkTest(String serverName, long testStartTime, int row) {
    // deleted lines
}

**Figure 5.16** Implementation of the reliable java socket communication.

**ConfirmedServiceTest** consists in execution of the confirmed SEs; the send SP is always followed by the receive SP (Figure 5.17). **UnconfirmedServiceTest** consists in series of unconfirmed SE, i.e. send (or receive) SP followed by each other in particular time intervals. At figure Figure 5.18 the implementation of one SE/SP for both test cases is given.

**Figure 5.17** Confirmed service test case execution.

private boolean upLinkTest(String serverName, long testStartTime, int row) {
    // deleted lines
}
ConfirmedServiceTest and UnconfirmedServiceTest classes have a counter functionality of selecting of the executed SE characteristics such the whole set of workload characteristics’ values given in the workload parameters is explored. Moreover both functions have counter functionality implementing the repetitions of the workload data with the same parameters (see Figure 5.19 where essentials of the procedures are given). Measurement functionality is also partially implemented in these both classes - after each execution of SE data is collected and then sample is saved (indicated in italic at Figure 5.19).
for (int second = 0; second < transmissionWindow; second++) {
    /**
     * generate currentSendPacketRate packets to transmit,
     * calculate the inter packet waiting time timeToWait and wait
     * accordingly between transmission of consecutives packets
     */
    for (int packetNumber = 1; packetNumber <= currentSendPacketRate; packetNumber++) {
        packet.constructPacket(
            Packet.constructPCI( this.packetSeqNo, sendPacketSize, currentSendPacketRate ),
            Packet.constructDataSDU( sendPacketSize ), 0, 0 );
        // send packet
        communication.send_tcp( dataOutToServer, packet );
        requestTime = packet.getTimeStamp();
        // No communication error occurred
        saveInterimTestResult( this.packetSeqNo, currentSendPacketRate,
            sendPacketSize, 0, // recvPacketSize = 0
            requestTime, 0 ); // indicationTime = 0
        /*
         * calculate the inter packet waiting time [ms]
         * requestTime - timeStamp = transmission time of an individual
         * packet
         */
        if ( packetNumber < currentSendPacketRate ) {
            interPacketTime = ( 1000 / currentSendPacketRate ) -
                ( requestTime - timeStamp );
        } else { // last packet
            /* padding ms to roundup to 1000 ms and add BACKOFF_TIME to
             * avoid mutual influence of consecutive test runs
             */
            interPacketTime = 1000 - ( packetNumber - 1 ) *
                ( 1000 / currentSendPacketRate ) -
                ( requestTime - timeStamp );
        }
        waitTime( interPacketTime );
    } // for each packet of the same rate [1..actualRate]
} // for each second in transmissionWindow
} // for (int currentSendPacketRate = startSendPacketRate;
repeat++;)
} // while ( repeat <= Server.SAMPLE_SIZE )

// for N packet sizes execute....
// for each packet size S execute...
// repeat procedure x times
currentTime = System.currentTimeMillis();
if ( Client.MAX_TEST_RUN_TIME > currentTime - testStartTime ) {
    // Confirmed test
    // send packet
dataSocket.setSoTimeout( Client.TIMEOUT_DAT_CHNL );
    // wait to receive packet maximum TIMEOUT_DAT_CHNL [ms]
    // Unconfirmed Test
    // send packet
    waitTime( interPacketTime );
} else { // receiving nor sending possible, service time elapsed!

Figure 5.20 Workload generation and measurement function for unconfirmed service test.

Evaluation service delivery is timed in two ways: workload generation function is timed globally,
such there is a limited time in which the SoD is examined, and also each SE is timed i.e. for a
confirmed service after each send action there is a maximum time for a receive notification to
arrive and for unconfirmed service there is a time calculated in between two consecutive send
actions (based on the current packet rate). The implementation details are as shown at Figure
5.27. If service time elapses, data is saved and procedure is interrupted at any point of its
execution. If SE timeout occur the timing error is raised by the procedure to the workload
generation function.

// for N packet sizes execute....
// for each packet size S execute...
// repeat procedure x times
currentTime = System.currentTimeMillis();
if ( Client.MAX_TEST_RUN_TIME > currentTime - testStartTime ) {
    // Confirmed test
    // send packet
dataSocket.setSoTimeout( Client.TIMEOUT_DAT_CHNL );
    // wait to receive packet maximum TIMEOUT_DAT_CHNL [ms]
    // Unconfirmed Test
    // send packet
    waitTime( interPacketTime );
} else { // receiving nor sending possible, service time elapsed!
saveMeasurementData( serverName, currentTestRun, timeStampTestRun,
    System.currentTimeMillis(), sendPacketSize,
  recvPacketSize, vMatrixOfSendRecvTimes );
return false;
}

Figure 5.27  Implementation of timing peculiarities in evaluation service delivery.

The measurement function is implemented in Communication, Packet and SaverOfPerformanceData classes and partially in confirmed and unconfirmed test functions. Communication and Packet classes hold functionality of timestamping the single events and carrying the timestamp along with the packet. Timestamps are taken in milliseconds as difference between the current system time and midnight, January 1, 1970 UTC (System.currentTimeMillis method). While executing the confirmed/unconfirmed tests, particular observations are collected. When the sample is completed, it’s forwarded to be stored by SaverOfPerformanceData. See Figure 5.19 and Figure 5.20 for details. Therefore the measurement function fulfils the requirement on the automated monitoring of events and correlation of measurements to workload parameters. SaverOfPerformanceData comprises the functionality of storage of the sample and its measurement context data in the file. For later retrieval purposes, the file has meaningful name, which consists on selected system and workload parameters. The example file name and file content are given on Figure 5.28.

Confirmed test example
File name: 01_0100_174.174_1079013261736.txt

File content:
**********************************************************************
* owners           : Kate Wac & Richard Bults, University of Twente
* program version  : 2.3.2
* date             : Thu Mar 11 14:53:51 GMT+01:00 2004
* test run no.     : 1
* *
* hostname         : 01
* service type     : TCP_USER_CONFIRMED
* sample size      : 500
* trans. window    : 40 (unconfirmed service only)
* buffer sizes [B] : socket        65536
*                    application   65536
* timing [ms]       : start         1079013261736
*                    stop          1079013534398
*                    backoff       30000
* link specs.       : link capacity 8192
*                    sat. factor   1.0
**********************************************************************
* Colum legenda:
* Sno = Packet sequence number, Pr = Packet rate,
* SPs = send packet size, RPs = received packet size,
* Rt = send packet request time, Ct = received packet confirm time
*
* Sno   Pr  SPs  RPs  Rt  Ct
* 1     1  174  174 1079013261736 1079013262447
* 2     1  174  174 1079013262447 1079013263268
* * *
* 500   1  174  174 1079013534178 1079013534398

Figure 5.28  Example of the measurement data file.

There is clearly no cooperation between the components of the evaluation system: the workload generator and the measurement system; both functionalities are included o co-existing manner mainly in the Packet, Communicatio, ConfirmedServiceTest and UnconfirmedServiceTest classes. The workload generator function uses socket interface to the SoD, while the measurement function is in passive mode, it just timestamps events observed on the interface.
Implemented evaluation system has an extended functionality of handling the errors; end user is notified on the successful or erroneous service execution, measurement data is saved in case of error. See Figure 5.29 for example of the error handling functionality.

```java
// Client class
error = executeTestCycle( serverName, getRowMatrixCoordinates(),
                         getColumnMatrixCoordinates() );
------/------
// ConfirmedServiceTest class
< deleted lines>
System.out.println( "Info [confirmedServiceTest]: test is running..." );
if ( packet.getErrorCode() == Communication.COMMUNICATION_ERROR ) {
    System.out.println( "Error [confirmedServiceTest]: send communication error occurred " );
    saveInterimTestResult( this.packetSeqNo, maxPacketRate,
                           sendPacketSize, 0, // recvPacketSize = 0
                           packet.getErrorCode(), // error = requestTime
                           0 ); // confirmaTime = 0
    saveMeasurementData( serverName, currentTestRun, time StampTestRun,
                         System.currentTimeMillis(), sendPacketSize,
                         recvPacketSize, vMatrixOfSendRecvTimes );
    return true; // error, test run interrupted
}< deleted lines>
saveMeasurementData( serverName, currentTestRun, time StampTestRun,
                      System.currentTimeMillis(), sendPacketSize,
                      recvPacketSize, vMatrixOfSendRecvTimes );
System.out.println( "Info [confirmedServiceTest]: SE " + sendPacketSize +
                   "," + recvPacketSize + " test run successfully finished in " +
                   ( System.currentTimeMillis() - timeStampTestRun ) / 1000 + " s" );
return false; // no error
---/----
// Client class
testCycleExitStatus = error ? "forced" : "normal";
System.out.println( "Info [run]: " + testCycleExitStatus +
                    " exit test cycle" );
```

Figure 5.29  End user notification on the service delivery status.

### 5.1.3 Evaluation System Synchronization

In the previous section the functionality of the evaluation system instance has been presented. Due to the distribution of the transport system and the nature of delivered transport service, the evaluation system is also distributed. There is always a pair of evaluation system instances that interact in order to deliver the service to end user. The service view on the system containing two evaluation system instances that interact over the transport system is shown at Figure 4.10. The functionality of interacting evaluation systems is slightly different from each other, particularly regarding the workload generator function. Functionality regarding the measurements system in both systems is identical.

Recalling the example scenario of systems interaction given in section 4.4, workload generator of the first instance takes initiative of SoD stimulus generation, when the second instance is responsive for this stimulus, and as a return it generates a new stimulus to SoD. Due to the specific functionality of instance’s workload generators, they are called: server and client. Server (or client) is the name for one particular instance of the evaluation system. Server is an instance of the workload generator which always initializes the transport service and client is an instance, which can respond to the behaviour of the server. Moreover, server is the workload generator responsible for generation of SE’s for the “upstream” link while client is the workload generator responsible for generation of SE’s for the “downstream” link.
The name “server” is inherited from the MobiHealth system architecture and represents the BAN as a data serving object (associated with “upstream”). The name “client” is inherited from the MobiHealth system architecture and represents the BESys as a data obtaining object (associated with “downstream”).

Server and client can provide evaluation service only when their activities are synchronized. Therefore there is important obstacle identified:

How to synchronise the functional building blocks of the evaluation system instances in time?

Moreover, it is assumed that the user of the evaluation system provides workload and system parameters once to the system (regardless of number of instances of evaluation system examining SoD). For every instance of the evaluation system these parameters need to be provided into this new system object such the actions of individual instance of evaluation systems are synchronized. Therefore the other important problem is pointed:

How to synchronise the behaviour of the evaluation system instances to create coordinated actions?

The implemented possible solutions for both problems will be given in the following sections.

**Evaluation system instances time’s synchronization**

The synchronization of evaluation system instances’ actions in time is of vital importance for the successful evaluation service delivery. The real-time clock of the machines, at which the instance of evaluation system is running, has to be synchronized accurately with some external time source by means of Network Time Protocol (NTP) protocol or Simple Network Time Protocol (SNTP).

There were two cases of clock synchronization distinguished depending on:

1) the location of the time source (i.e. NTP server) with relation to SoD
2) interface as used to deliver the evaluation service or dedicated interface for the synchronization.

Inband time synchronization consists in time source inside the SoD and using one interface for synchronization and evaluation service traffics. In contrary, the outband time synchronization consists in time source outside the SoD and using dedicated interfaces for synchronization and evaluation service traffics.

**Inband time synchronization**

*Definition:*
In the public switched telephone network, (PSTN), *in-band signaling is the exchange of signaling (call control) information on the same channel that the telephone call itself is using.*

In case of the evaluation system inband time synchronization time source is placed within the SoD and evaluation system synchronizes its clock over the same link that the evaluation service is executed. See Figure 5.30 for details.

![Inband time synchronisation](image)

**Figure 5.30 Inband time synchronisation.**

Inband time synchronization was used for the case when different computer systems (with attached evaluation systems to them) needed to be synchronized and due to the infrastructural reasons it was not feasible to setup the outband connection for this purpose. That was the case of the evaluation system attached to SoD_6.

The software tool used for time synchronization was Tardis - running as the Windows XP’s background process and automatically synchronizing system clock with provided NTP server(s) on scheduled basis. This program promises synchronization precision of 1ms. The NTP server used for synchronization purpose is *ntp.utwente.nl* (130.89.1.19, alias: *prodnet.civ.utwente.nl*) placed within the UTnet. The update of the system clock has been chosen as frequently as the application allowed (e.g. every 10 minutes) in order to stay synchronized and to avoid the clock drift. For the iPAQ platform there was *ntpd* command used for synchronizing of the clock with the external time source.

**Outband time synchronization**

*Definition:*

*Out-of-band signalling is telecommunication signalling (exchange of information in order to control a telephone call) that is done on a channel that is dedicated for the purpose and separate from the channels used for the telephone call.*

In contrary, the outband time synchronization consists in time source outside the SoD and using dedicated interfaces for synchronization and evaluation service traffics.
Outband time synchronization was used for the case when synchronization over the wireless link has not convenient and it was possible to setup wired connection dedicated for the purpose of synchronization. Outband connection was setup to the third system, which acted as NTP server.

The software tool used for time synchronization was Tardis. Tardis client instances were running on two machines which communicate to the third one, acting as NTP server\footnote{\url{http://www.winguides.com/registry/display.php/1117/}}. The system used was 192.168.0.48 placed within the dedicated LAN (192.168.0.0 / 24 network). The clocks of NTP clients were upgraded automatically in scheduled time intervals of 10 minutes. Frequent synchronization ensured the clock correctness and the clock drift avoidance during the tests execution.

Evaluation system instances behaviour’s synchronization

Coordination of the client and server actions has been dealt with by introducing of the evaluation service initialization phase. In this phase the client instance provides set of workload and system parameters (i.e. management data) to its server peer evaluation system instance. Successful execution of the initialization phase is required before the evaluation service execution can start (i.e. instances start data generation). Management data exchange is done over the dedicated reliable data channel between the client and service instances and it is independent from the data channel over which server and client exchange the workload data. Particularly the client sends the workload and system parameters to the server as indicated at Figure 5.32.

```java
/**
 *  Date of testcycle execution
 */
public static String TEST_DATE = date.toString();
/**
 *  Executed service type
 */
public final static String SERVICE_TYPE = CONFIRMED_SERVICE_TYPE;
/**
 *  Maximum time interval [ms] for evaluation service provision
 */
```
Service type determines the way server and client generate workload to the system. Regarding the workload parameters, client feeds the server with the information on where to start the data generation in the matrix (coordinates) and how many times each SE needs to be repeated (sample size). Moreover the link capacity system parameter and its saturation value (workload parameter) are of importance.

For administration purposes client provides also the test date to the server (measurement function saves then the test date in the file header when saving the sample). Maximum test run time is given; the delivery of evaluation service lasts only for a specified (maximum) time. Also the value of the read timeout on socket is given to the server.

As it was said in the previous section, the automatization of the evaluation service delivery has been implemented in such way that the number of successive independent evaluation service executions (test runs) by one evaluation system instance can follow, forming the test cycle. This option is particularly utilized when multiple instances of paired server-client systems examine SoD simultaneously. At Figure 5.33 the TEST_OPTION_NUMBER value with relation to the number of server-client systems is visualized.

```java
/**
   * TEST_OPTION_NUMBER controls the number of test runs and the test run per
   * test cycle. This attribute can have a value 1..MAX_NUMBER_OF_SERVERS
   * (i.e. maximum number of servers that can be registered to this client).
   *
   * Example use: MAX_NUMBER_OF_SERVERS = 5 and TEST_OPTION_NUMBER = 3.
   * The test cycle will be constructed based on the figure below:
*/
```
The test cycle starts with a run that uses three servers in parallel and end with a run that uses five servers in parallel.

```java
public final static int MAX_NUMBER_OF_SERVERS = 1;
public final static int TEST_OPTION_NUMBER = 1;  // [1...MAX_NUMBER_OF_SERVERS]
```

Figure 5.33  Test cycle management implementation.

The assumption is that all instances of the server-clients peers examine the SoD simultaneously and by means of the same procedures i.e. using the same system and workload parameters. As it was pointed at the beginning of the section, the user of the evaluation system input the workload and system parameters only once to the system, regardless the number of evaluation systems instances. There is only one object (Client object), which has all the knowledge on the system and workload parameters and it distributes the knowledge to all instances of client evaluation systems during the initialization phase (and clients further distribute it to their paired servers). Therefore each new test cycle starts with the initialization phase during which the parameters in the instance of server-client evaluation system are setup.

The implementation of the coordination of the number of client evaluation systems has been supported by means of the java threads. Each independent instance of the client evaluation system is then ServerControl object running as a separate java application thread. Therefore each instance of server-client peered evaluation systems can execute the evaluation service independently, but in synchronized with the other instances running on SoD.

Due to the organization of the evaluation service delivery in test cycles, each Server evaluation system instance gets the matrixTestRunStartTimes parameter from its peered ServerControl (client) evaluation system. In this parameter there are time values at which the server is expected to start the evaluation service delivery execution. Therefore, in order to successfully deliver the evaluation service, both (i.e. peered server and client) system clocks needs to be synchronized. This implies that for a case when multiple instances of peered server-client evaluation systems are delivering service simultaneously, all instance’s clocks need to be synchronized as well.

### 5.2 Measurements Setup and Execution

In this section the details on setup and execution phase of the performance measurements are given. Setup consists of the workload and system parameters combinations and then, based on that, designing of experiments. We have two types of measurements : active and passive. The active measurements are laboratory measurements made in a controlled environment on different networks, while the passive measurements are taken during the actual trials. Sections 5.2.1 and 5.2.2 describe the set-up and result of the active measurements activities, while the answers on “where” “when” and “who” executed the active measurements experiments is given in section 5.2.2. Section 5.2.3 describes the passive measurements’ set up and provides a overview of the data analysis.
5.2.1 Setup

In section 4.7 possible workload and system parameters’ values has been presented. In section 5.1.1 eight SoD instances with different system parameters’ setups were pointed. Next step is to identify, which combinations of these parameters will be used during measurements. Ideally, the measurements should be performed in such way, that all possible combinations of the workload and system parameters (i.e. SoDs) are taken into account. That would give 19200 different experiments\(^4\). Moreover, each experiment needs to be repeated a number of times in order to get statistically sound data for the obtained system performance parameters. Due to resource constraints, only a selected number of combinations of parameters were taken into account, such the most relevant data with the minimum number of experiments could be obtained.

In this section the combinations of the workload and system parameters used during the measurement execution activity, are presented.

The selection criteria of the combinations of the workload and system parameters of interest were:

a) resource constraints (e.g. time, equipment),
b) priority criteria: some of the associations were of higher importance than others.

The resource constraints were as follows:

a) the time constraint: there were 45 days available for measurements and following Jain’s note [Jain1991] the worst case scenario has been anticipated: “Murphy’s law strikes measurements more often than other (evaluation) techniques. If anything can go wrong, it will. As a results, the time required for measurements is the most variable among the three (evaluation) techniques” (i.e. analytical modelling, simulation and measurements). The worst case scenario would mean the repetition of the complete set of measurements, which resulted in estimation of ~23 days available for effective measurements.

b) the personnel constraint: there were two people available for 80% of the time

c) the equipment constraint: there was limited number of equipment available e.g. only one notebook platform, so the scalability test could no be executed with this platform.

Moreover, during the selection of the workload and system parameters combinations, there were priority criteria taken into account. Namely, there was required a benchmark\(^4\) measurement for the confirmed service case (see section 4.7). For the unconfirmed service test, only the most important workload and system parameters were pointed - packet size 524MB, also due to the time constraint.

As the result of the selection of the associations of the SoDs and workload parameters, the table has been obtained as presented Figure 5.34. Cross in the cell means that the combination of the

\(^{41}\) \( (8 \text{ SoDs}) \times (20 \times 20 \text{ matrix } + 8 \times 1 \text{ matrix } \times 10 \text{ saturations}) \times (3 \text{ No of users}) = 19200 \)

\(^{42}\) Benchmark is selected to be run on particular SoD configuration with the goal of fair comparison of the performance of other SoD configurations. Benchmark characteristics are: fair (minimized bias of a specific platform), relevant, easy to measure, easy to explain.
workload and system parameter given by the cell is used in the measurement activity. Evaluation system instance (java program) is ported at each computer system in SoD; the instance of evaluation system called server is ported such it generates the upstream traffic to SUT (i.e. it’s always ported at nb or ipaq computer system), and the client instance generates the SUT downstream traffic (i.e. it’s always ported at pc2 computer system). In case SoD_6, there is additional instance of evaluation system ported in between the V3GNL SUT and UTnet SUT (i.e. at pc2 computer system). This instance of evaluation system is called proxy, due to the fact that for the server it’s seen as a client, and for the client as the server.

<table>
<thead>
<tr>
<th>computer systems</th>
<th>nb, pcl</th>
<th>nb, pcl</th>
<th>nb, pcl</th>
<th>nb, pcl, pc2</th>
<th>ipaq, pcl</th>
<th>nb, pcl</th>
<th>nb, pcl</th>
<th>nb, pcl</th>
<th>nb, pcl</th>
</tr>
</thead>
<tbody>
<tr>
<td>intra comm.</td>
<td>usb</td>
<td>bluetooth</td>
<td>PCMCIA</td>
<td>usb</td>
<td>bluetooth</td>
<td>usb</td>
<td>usb</td>
<td>usb</td>
<td>usb</td>
</tr>
<tr>
<td>UMTS terminal</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>PC Card</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td>Nokia 6650</td>
<td></td>
</tr>
<tr>
<td>APN</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
<td>web. vodafone. nl</td>
<td>utwente.nl</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>buff. sizes: appl.socket</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
<td>64.64</td>
<td>32.64</td>
<td>32.32</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.34 System and associated workload parameters.

The detailed specification of the equipment used for the measurement activity can be found as Appendix 1.

---

43 A observation is a repetition of SE. The number of observations in sample is 500 in order to get statistically sound data.
Based on the eight instances of SoD and identified combination of the workload and system parameters, there were 11 different experiments designed to be executed. Experiments details will be given in the following sections. Experiments naming convention (i.e. TestID) is “T” plus the test number (test number is not necessary the same as an experiment number).

The description of each experiment consisting of the specification of the workload and SoD parameters is given in the table as presented at Figure 5.35. The example values of particular parameters are given in italic.

<table>
<thead>
<tr>
<th>TestID</th>
<th>Experiment number:</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>e.g. 01</td>
</tr>
<tr>
<td>HOPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>e.g. 01</td>
</tr>
</tbody>
</table>

**Test description**

- **ServiceType**: Confirmed Service, 20x20, UMTS-UT/USB, Server/Client, 1 Terminal
- **EvalSys1/EvalSys2 [EvalSys3]**
- **NoOfEvalSys**: 1

**Extra comms**

For the comp. system at the ingress point of V3GNL – it is a wireless communication to access V3GNL i.e. UMTS

**Intra comms**

For the comp. system at the ingress point of V3GNL - it is a wireless communication to access the UMTS terminal e.g. USB, bluetooth or PCMCIA

**Clock sync**

Software used to synchronize server system clock e.g. Tardis 2000 V1.5

Value of clock drift (for server) e.g. 1s / test run

NTP server used e.g. ASUS nb

**Equipment** - computer systems on which instances of evaluation system are ported plus UMTS terminal used

<table>
<thead>
<tr>
<th>Identifier</th>
<th>name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>pc1, e.g. Utip194</td>
<td>e.g. P4, 2.4 GHz, 512 MBmem</td>
<td>Operating system e.g. WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>nb, ipaq, e.g. Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>Operating system e.g. WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>e.g. Nokia 1</td>
<td>e.g. Nokia 6650</td>
<td>e.g. PR4</td>
</tr>
</tbody>
</table>

**Figure 5.35** Example of experiment specification table.

Designed experiments details are given in the following sections.
Experiment 1

<table>
<thead>
<tr>
<th>TestID</th>
<th>01</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 20x20, UMTS-UT/USB, Server/Client, 1 Terminal</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis 2000 V1.5</td>
</tr>
</tbody>
</table>

Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 1</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.36 Specification of the experiment 1.

Experiment 2

<table>
<thead>
<tr>
<th>TestID</th>
<th>04</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT/BT, Server/Client, 1 Terminal</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>Bluetooth 1.0</td>
</tr>
<tr>
<td>Clock sync</td>
<td>ntpdate</td>
</tr>
</tbody>
</table>

Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 20</td>
<td>Enschede-20</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rc8-kernel-1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 8</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.37 Specification of the experiment 2.
## Experiment 3

<table>
<thead>
<tr>
<th>TestID</th>
<th>05</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT/BT, Server/Client, 5 Terminals in area ~10m²</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>Bluetooth 1.0</td>
</tr>
<tr>
<td>Clock sync</td>
<td>ntpdate</td>
</tr>
</tbody>
</table>

### Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 3</td>
<td>Enschede-16</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 8</td>
<td>Enschede-17</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 13</td>
<td>Enschede-18</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 19</td>
<td>Enschede-19</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 20</td>
<td>Enschede-20</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>Terminals</td>
<td>Nokia 1, 2, 3, 8, 9</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.38 Specification of the experiment 3.
### Experiment 4

<table>
<thead>
<tr>
<th>TestID</th>
<th>03</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT/BT, Server/Client, 10 Terminals in area ~10m²</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>Bluetooth 1.0</td>
</tr>
<tr>
<td>Clock sync</td>
<td>ntpdate ntp.utwente.nl</td>
</tr>
</tbody>
</table>

**Equipment:**

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 03</td>
<td>Enschede-3</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 08</td>
<td>Enschede-8</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 13</td>
<td>Enschede-13</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 14</td>
<td>Enschede-14</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 15</td>
<td>Enschede-15</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 16</td>
<td>Enschede-16</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 17</td>
<td>Enschede-17</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 18</td>
<td>Enschede-18</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 19</td>
<td>Enschede-19</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>ServerId 20</td>
<td>Enschede-20</td>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13, CVM J2ME personal profile 1.0</td>
</tr>
<tr>
<td>Terminals</td>
<td>Nokia 1..10</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.39 Specification of the experiment 4.
### Experiment 5

<table>
<thead>
<tr>
<th>TestID</th>
<th>06</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT/BT, Server/Client, 1 Terminal</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>Bluetooth 1.0</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0, BT dongle</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 11</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.40 Specification of the experiment 5.

### Experiment 6

<table>
<thead>
<tr>
<th>TestID</th>
<th>07</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;&gt; Proxy &lt;&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT/USB, Server/Proxy/Client, 1 Terminal</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>ntpdate ntp.utwente.nl</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 1</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.41 Specification of the experiment 6.
Experiment 7

<table>
<thead>
<tr>
<th>TestID</th>
<th>08</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>web.vodafone.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-Vodafone/USB, Server/Client, 1 Terminal</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>ntpdate</td>
</tr>
<tr>
<td></td>
<td>ntp.utwente.nl</td>
</tr>
</tbody>
</table>

Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mp3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 1</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.42 Specification of the experiment 7.

Experiment 8

<table>
<thead>
<tr>
<th>TestID</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Confirmed Service, 8x8, UMTS-UT, Server/Client, 1 PCMCIA Card</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>None</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis</td>
</tr>
<tr>
<td></td>
<td>ASUS notebook</td>
</tr>
</tbody>
</table>

Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mp3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0, BT dongle</td>
</tr>
<tr>
<td>Terminal</td>
<td>PC card</td>
<td>PCMCIA UMTS</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.43 Specification of the experiment 8.
### Experiment 9

<table>
<thead>
<tr>
<th>TestID</th>
<th>09</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Unconfirmed Service (uplink only), 524B, UMTS-UT/USB, Server/Client, 1 Terminal Buffer size: application 64 KB, socket 64 KB</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis 2000 V1.5</td>
</tr>
</tbody>
</table>

#### Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 11</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.44 Specification of the experiment 9.

### Experiment 10

<table>
<thead>
<tr>
<th>TestID</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;-&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Unconfirmed Service (uplink only), 524B, UMTS-UT/USB, Server/Client, 1 Terminal Buffer size: application 32 KB, socket 64 KB</td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis 2000 V1.5</td>
</tr>
</tbody>
</table>

#### Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 11</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.45 Specification of the experiment 10.
Experiment 11

<table>
<thead>
<tr>
<th>TestID</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>APN</td>
<td>utwente.nl</td>
</tr>
<tr>
<td>HOPS</td>
<td>Server &lt;&gt; Client</td>
</tr>
<tr>
<td>Test description</td>
<td>Unconfirmed Service (uplink only), 524B, UMTS-UT/USB, Server/Client, 1 Terminal</td>
</tr>
<tr>
<td>Buffer size: application 32 KB, socket 32 KB</td>
<td></td>
</tr>
<tr>
<td>Extra comms</td>
<td>UMTS</td>
</tr>
<tr>
<td>Intra comms</td>
<td>USB</td>
</tr>
<tr>
<td>Clock sync</td>
<td>Tardis 2000 V1.5</td>
</tr>
</tbody>
</table>

Equipment:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>HW-platform</th>
<th>SW-platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientId 01</td>
<td>Utip194</td>
<td>P4, 2.4 GHz, 512 MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>ServerId 01</td>
<td>Freelander</td>
<td>mP3, 1 GHz, 640MBmem</td>
<td>WindowsXP, JDK 1.3.0</td>
</tr>
<tr>
<td>Terminal</td>
<td>Nokia 11</td>
<td>Nokia 6650</td>
<td>PR4</td>
</tr>
</tbody>
</table>

Figure 5.46 Specification of the experiment 11.

Each experiment starts with the initialization phase, during which all clocks of the computer systems, where evaluation systems are ported, are synchronized. Evaluation service end-user provides the workload and system parameters to the system. This management data is exchanged between the evaluation system instances; client sends management data to the server. The management channel between the peered client-server evaluation system instances is established over the TCP connection port number\(^{44}\) 4444. Data is exchanged over the separate TCP channel on port number 6790.

The computer systems themselves and the links between computer systems involved in experiments are kept as silent as possible during the measurements. This means that no processes other than the ones strictly needed for the measurement are running on the systems and no other traffic is permitted on the links. Applications and other users/processes, such as window environment, that may interfere with the measurements are not allowed on the computer systems during the experiments.

### 5.2.2 Execution

All tests were executed in the timeframe of 1\(^{st}\) February 2004 – 31\(^{st}\) march 2004 by authors of this report. Due to the fact that the V3GNL has been free from other users (no commercial

\(^{44}\) In a TCP/IP-based network such as the Internet, it is a number assigned to an application running in the computer. The number is included in the transmitted packets to link the incoming data to the correct service. [WebTech]
network), tests were done without taking into account of the V3GNL load distribution. Every day or night was treated the same.

Entities (equipment) and parties that have been involved in test activities were placed, as indicated at the figures. Server(s) and Client(s) was always placed at the University of Twente, Zilvering building, room 5013. For scalability test, all the servers involved have been placed in one room (5013), which means in the area of ~10m².

The coverage of the Vodafone’s UMTS network was always full. There was no mobility or system/ intersystem (gprs/umts) handover tested. All equipment has been fully charged and constantly under the power.

In this section the execution results of experiments are given. Below the experiment description there is table with indication which SEs have been executed successfully. In most cases not all planned SEs tests were executed successfully. That was due to the time constraint. The legend of SEs execution is given at Figure 5.47.
<table>
<thead>
<tr>
<th>500</th>
<th>174</th>
<th>349</th>
<th>524</th>
<th>...</th>
<th>35632</th>
<th>41920</th>
<th>48208</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>0,0</td>
<td>0,1</td>
<td>0,2</td>
<td>...</td>
<td>0,17</td>
<td>0,18</td>
<td>0,19</td>
</tr>
<tr>
<td>349</td>
<td>1,0</td>
<td>1,1</td>
<td>2</td>
<td>...</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>524</td>
<td>2,0</td>
<td>1,1</td>
<td>2</td>
<td>...</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7336</td>
<td>17,0</td>
<td>1</td>
<td>2</td>
<td>...</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>7860</td>
<td>18,0</td>
<td>1</td>
<td>2</td>
<td>...</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>8122</td>
<td>19,0</td>
<td>1</td>
<td>2</td>
<td>...</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
</tbody>
</table>

NOP: not planned, not executed
1: planned, not executed
2: out-band time synchronisation
3: in-band time synchronisation
4: execution failed

Figure 5.47 Legend for test run matrix.
### Experiment 1

#### a) clock synchronization data

Clock sync | Tardis 2000 V1.5 | 0.712 seconds/day | ASUS notebook  
--- | --- | --- | ---  
174 | 0.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
349 | 1.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
524 | 2.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
786 | 3.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
1048 | 4.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
1310 | 5.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
1572 | 6.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
2096 | 7.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
2620 | 8.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
3144 | 9.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
3668 | 10.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
4192 | 11.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
4716 | 12.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
5240 | 13.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
5764 | 14.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
6288 | 15.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
6812 | 16.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0 
7336 | 17.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
7860 | 18.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0  
8122 | 19.0 | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 | 17.0 | 18.0 | 19.0

#### b) executed SEs
Experiment 2

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>ntpdate</th>
<th>0.01 seconds/testrun</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>524</td>
<td>1.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>1048</td>
<td>2.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>1572</td>
<td>3.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>2096</td>
<td>4.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>4192</td>
<td>5.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>6288</td>
<td>6.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>7860</td>
<td>7.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

b) executed SEs

Figure 5.48  Test run matrix for experiment 1.

Figure 5.49  Test run matrix for experiment 2.
Experiment 3

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>ntpdate</th>
<th>0,026 seconds/testrun</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
</table>

b) executed SEs

Note: Terminal Nokia 1 was swapped by Nokia 4 on February 2, 2004 due to repetitive communication errors in test SEs

![Matrix](image)

Figure 5.50  Test run matrix for experiment 3.
### Experiment 4

**a) clock synchronization data**

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>ntpdate</th>
<th>0.010 seconds/testrun</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>524</td>
<td>1.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>1048</td>
<td>2.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>1572</td>
<td>3.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>2096</td>
<td>4.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>4192</td>
<td>5.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>6288</td>
<td>6.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>7860</td>
<td>7.0</td>
<td>1.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**b) executed SEs**

![Test run matrix for experiment 4.](image)

Figure 5.51 Test run matrix for experiment 4.
Experiment 5

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>Tardis 2000 V1.5</th>
<th>2.89 s/day clock drift</th>
<th>ASUS notebook</th>
</tr>
</thead>
</table>

b) executed SEs

```
 500 174 524 1572 2096 6288 8384 14672 16768
 174 0.0 1 2 3 4 5 6 7
 524 1.0 1 2 3 4 5 6 7
1048 2.0 1 2 3 4 5 6 7
1572 3.0 1 2 3 4 5 6 NOP
2096 4.0 1 2 3 4 5 NOP NOP
4192 5.0 1 2 3 4 5 NOP NOP
6288 6.0 1 2 3 4 5 NOP NOP
7860 7.0 1 2 3 4 5 NOP NOP
```

Figure 5.52 Test run matrix for experiment 5.
Experiment 6

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>ntpdate</th>
<th>11.669 s/day clock drift</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
</table>

b) executed SEs

Figure 5.53 Test run matrix for experiment 6.
Experiment 7

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>ntpdate</th>
<th>1.9 s/day clock drift</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b) executed SEs

Due to the time constraint only two packet sizes has been executed, therefore the measurement results are considered as indicative.

Figure 5.54 Test run matrix for experiment 7.
Experiment 8

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>Tardis 2000 V1.5</th>
<th>-1.892 s/day clock drift</th>
<th>ASUS notebook</th>
</tr>
</thead>
</table>

b) executed SEs

```
<table>
<thead>
<tr>
<th>500</th>
<th>174</th>
<th>1572</th>
<th>2096</th>
<th>7860</th>
<th>8384</th>
<th>16244</th>
<th>16768</th>
<th>48732</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>524</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>1048</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>1572</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>2096</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>4192</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>6288</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>7860</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
```

Figure 5.55 Test run matrix for experiment 8.
Experiments 9, 10, 11

a) clock synchronization data

<table>
<thead>
<tr>
<th>Clock sync</th>
<th>Tardis 2000 V1.5</th>
<th>2.089 seconds/day</th>
<th>ntp.utwente.nl</th>
</tr>
</thead>
</table>

b) executed SEs

Number of observations: 30 (time constraint)

![Test run matrix for experiment 9,10, and 11.]

5.2.3 Results of the EDR data analysis

One of the mechanisms used for collection of evaluation data was automatic data capture. A software module was designed and implemented as part of the BANware. This module logs usage statistics during use of the BANs.

The logs, which were stored locally on the MBUs, were uploaded at intervals by the trial owners according to a defined Upload Protocol. The logs are stored on a server at the University of Twente. The logs are referred to as EDR (Evaluation Data Record) logs. An example of an EDR log file is shown below:
EDR log files contain comma separated values and each record consists of 7 columns. The headers of these columns are also written into the EDR log and are separated by a semi-colon. The interpretation of each column is explained below:

1) **Event** – This indicates why an entry was written into the log file. Possible values are: **Start** (Start button on MBU application was pressed), **Intermediate** (this is written every 20 seconds during the measurement session to indicate that the session is ongoing), **Stop** (Stop button on MBU application was pressed) and **Exit** (MBU application was exited).

2) **Time** – Moment in time that this event took place. Note that this is the time of the iPAQ, which usually is completely out of sync with reality. It may only be used as a relative timer.

3) **Bytes Sent** – The number of bytes sent during this measurement session.

4) **Reason** – Indicates the reason why this event was generated. Possible values: **User** (The end-user pressed a button on the screen), **Sending** (The MBU is still sending data), **System** (A system event took place, without user intervention).

5) **Battery Level MBU** – Indicates the % of battery power of the main battery of the iPAQ.

6) **Battery Level Sensor Front End** – Indicates battery power of the Mobi. Note that this cannot be measured. It will always be N/A.

7) **Battery Level Sensor GPRS Modem** – Indicates battery power of the GPRS jacket. Note that this cannot be measured. It will always be N/A.

The EDR upload protocol is reproduced below.
**MH-EDR upload protocol**

**Instructions to trial owners:**

At each trial site: the trial owner should appoint a person responsible for upload of EDR data from all BANs at that site. Upload is a very simple procedure but MUST BE DONE or there will be no evaluation data from the BANs.

If possible please ensure as soon as possible that the system date/times are set correctly on the IPAQs before measurement sessions.

**STEP 1. TEST UPLOAD**

As soon as possible: For each BAN in the trials at your site

- Use the ‘EDR upload’ application on IPAQ
- Notify WPM5 and Scientific coordinator by email (quoting the BAN IDs)

**STEP 2. INTERMEDIATE UPLOAD**

Regularly (at least once per week)

For each BAN in the trials at your site

- Use the ‘EDR upload’ application on IPAQ

**STEP 3. FINAL UPLOAD**

<date to be agreed with WPM5 – as soon as possible after the trials have finished>

For each BAN in the trials at your site

- Use the ‘EDR upload’ application on IPAQ
- Notify WPM5 and Scientific coordinator by email (quoting BAN IDs)

**EDR UPLOAD**

Note 1: FOR WP4/5: You need to ask the trial owners to nominate a responsible person at each site. That person will be responsible for ensuring the trial owners EDR upload procedure is followed for each BAN and at the right time. Final upload should be as soon as possible after trials are finished for that BAN.

**EDR TRACKING**

Meanwhile to keep track, WP4/5 need an inventory of all BANs in use in the trials by BAN ID so you can make sure all data is uploaded and keep track of things using the following matrix:

<table>
<thead>
<tr>
<th>BANID</th>
<th>type</th>
<th>Country Sw/NL/D/S</th>
<th>trial</th>
<th>EDR upload responsible person</th>
<th>test upload planned date / completed</th>
<th>final upload planned date / completed</th>
</tr>
</thead>
</table>
Following the final upload of EDR logs, the logs had to be cleaned (to clean corrupted data and to fix timestamp anomalies, for example) and analysed. Analysis software for the EDR logs was designed and implemented using MySQL and PHP scripting.

In the following sections we present the results of analysis of the EDR logs collected during the MobiHealth trials. The results comprise descriptive statistics on the BAN sessions by country, by trial and by network type (GPRS or UMTS). Successful upload of EDRs was not achieved for all BANs (reasons as yet unknown, but communications failure is a possible cause). The results presented below therefore show only the subset of trial BANs for which successful registering and upload of EDR logs occurred.

**NUMBER OF BANS LOGGED AND NUMBER OF SESSIONS LOGGED**

EDR logs were uploaded to the server for a total of 30 of the trial BANs, and record 393 BAN sessions. 226 of the sessions were over GPRS and 167 over UMTS. Table 1 below shows the EDR logs by number of BANs and number of sessions by trial and for GPRS and UMTS.

<table>
<thead>
<tr>
<th>Country</th>
<th># BANS</th>
<th># Sessions</th>
<th># GPRS</th>
<th># UMTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>0</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>3</td>
<td>52</td>
<td>24</td>
<td>28</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>1</td>
<td>45</td>
<td>45</td>
<td>0</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>3</td>
<td>45</td>
<td>14</td>
<td>31</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>9</td>
<td>113</td>
<td>77</td>
<td>36</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>2</td>
<td>20</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>2</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Sweden/Resp</td>
<td>2</td>
<td>55</td>
<td></td>
<td>55</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>30</strong></td>
<td><strong>393</strong></td>
<td><strong>226</strong></td>
<td><strong>167</strong></td>
</tr>
</tbody>
</table>

**DURATION OF SESSIONS**

Duration of session was calculated according to the timestamps in the logs. Table 2 shows the durations of sessions (only for non-empty sessions, ie sessions wherein some data was successfully transmitted to the BeSys).

<table>
<thead>
<tr>
<th>Country</th>
<th>Duration (in minutes)</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>309.25</td>
<td>30.93</td>
<td>1.75</td>
<td>122.27</td>
<td>43.45</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>5136.15</td>
<td>160.5</td>
<td>0</td>
<td>1200.17</td>
<td>276.63</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>172.35</td>
<td>21.54</td>
<td>0.57</td>
<td>92.02</td>
<td>28.76</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>1582.47</td>
<td>98.9</td>
<td>0.68</td>
<td>894.03</td>
<td>213</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>922.25</td>
<td>16.47</td>
<td>0</td>
<td>204.95</td>
<td>30.95</td>
</tr>
</tbody>
</table>
Table 3 shows durations for UMTS sessions.

<table>
<thead>
<tr>
<th>Country</th>
<th>Duration (in minutes)</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>2601.67</td>
<td>153.04</td>
<td>0</td>
<td>731.82</td>
<td>204.02</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>1333.02</td>
<td>88.87</td>
<td>0.68</td>
<td>894.03</td>
<td>216.29</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>573.32</td>
<td>20.48</td>
<td>0.63</td>
<td>204.95</td>
<td>39.95</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>36.73</td>
<td>6.12</td>
<td>3.52</td>
<td>15.47</td>
<td>4.21</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>166.1</td>
<td>55.37</td>
<td>54.92</td>
<td>55.6</td>
<td>0.32</td>
</tr>
<tr>
<td>Sweden/Resp</td>
<td>3739.5</td>
<td>133.55</td>
<td>0</td>
<td>2531.63</td>
<td>467.34</td>
</tr>
<tr>
<td>Total</td>
<td>8450.33</td>
<td>87.12</td>
<td>0</td>
<td>2531.63</td>
<td>2848.49</td>
</tr>
</tbody>
</table>

Table 4 shows durations for GPRS sessions.

<table>
<thead>
<tr>
<th>Country</th>
<th>Duration (in minutes)</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>309.25</td>
<td>30.93</td>
<td>1.75</td>
<td>122.27</td>
<td>43.45</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>2534.48</td>
<td>168.97</td>
<td>0.92</td>
<td>1200.17</td>
<td>340.5</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>172.35</td>
<td>21.54</td>
<td>0.57</td>
<td>92.02</td>
<td>28.76</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>249.45</td>
<td>249.45</td>
<td>249.45</td>
<td>249.45</td>
<td>0</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>348.93</td>
<td>12.46</td>
<td>0</td>
<td>67.27</td>
<td>16.94</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>29.95</td>
<td>7.49</td>
<td>7.15</td>
<td>7.93</td>
<td>0.3</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/Resp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>3942.25</td>
<td>46.93</td>
<td>0</td>
<td>1200.17</td>
<td>1582.72</td>
</tr>
</tbody>
</table>
AMOUNT OF DATA TRANSMITTED

Tables 5-7 show the amount of data (in MBs) transmitted by sessions.

**Table 5** Bytes transmitted per (non-empty) session, by trial in MB

<table>
<thead>
<tr>
<th>Country</th>
<th>Total Bytes transmitted</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>21.49</td>
<td>1.95</td>
<td>0.19</td>
<td>8.19</td>
<td>2.25</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>263.04</td>
<td>7.52</td>
<td>0.06</td>
<td>94.79</td>
<td>16.94</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>23.69</td>
<td>2.96</td>
<td>0.1</td>
<td>9.82</td>
<td>3.29</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>57.52</td>
<td>3.2</td>
<td>0.04</td>
<td>8.18</td>
<td>2.62</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>180.77</td>
<td>3.06</td>
<td>0.03</td>
<td>60.29</td>
<td>8.59</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>12.51</td>
<td>1.14</td>
<td>0.05</td>
<td>2.23</td>
<td>0.72</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>21.88</td>
<td>7.29</td>
<td>1.51</td>
<td>12.33</td>
<td>4.45</td>
</tr>
<tr>
<td>Sweden/Resp</td>
<td>208.3</td>
<td>7.18</td>
<td>0.09</td>
<td>43.4</td>
<td>12.25</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>851.16</td>
<td>4.43</td>
<td>0.03</td>
<td>94.79</td>
<td>10.28</td>
</tr>
</tbody>
</table>

**Table 6** Bytes transmitted per (non-empty) session, by trial by UMTS in MB

<table>
<thead>
<tr>
<th>Country</th>
<th>Total Bytes transmitted</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>151.22</td>
<td>7.56</td>
<td>0.19</td>
<td>36.71</td>
<td>9.48</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>52.97</td>
<td>3.12</td>
<td>0.04</td>
<td>8.18</td>
<td>2.68</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>136.99</td>
<td>4.89</td>
<td>0.03</td>
<td>60.29</td>
<td>11.84</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>4.44</td>
<td>0.63</td>
<td>0.05</td>
<td>1.03</td>
<td>0.33</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>21.88</td>
<td>7.29</td>
<td>1.51</td>
<td>12.33</td>
<td>4.45</td>
</tr>
<tr>
<td>Sweden/Resp</td>
<td>208.3</td>
<td>7.18</td>
<td>0.09</td>
<td>43.4</td>
<td>12.25</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>575.79</td>
<td>5.54</td>
<td>0.03</td>
<td>60.29</td>
<td>10.15</td>
</tr>
</tbody>
</table>

**Table 7** Bytes transmitted per (non-empty) session, by trial by GPRS in MB

<table>
<thead>
<tr>
<th>Country</th>
<th>Total Bytes transmitted</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/Cardio</td>
<td>21.49</td>
<td>1.95</td>
<td>0.19</td>
<td>8.19</td>
<td>2.25</td>
</tr>
<tr>
<td>Netherlands/Preg</td>
<td>111.82</td>
<td>7.45</td>
<td>0.06</td>
<td>94.79</td>
<td>23.45</td>
</tr>
<tr>
<td>Netherlands/Trauma</td>
<td>23.69</td>
<td>2.96</td>
<td>0.1</td>
<td>9.82</td>
<td>3.29</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>4.55</td>
<td>4.55</td>
<td>4.55</td>
<td>4.55</td>
<td>0</td>
</tr>
<tr>
<td>Spain/outdoors</td>
<td>43.78</td>
<td>1.41</td>
<td>0.03</td>
<td>14.92</td>
<td>2.85</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>8.06</td>
<td>1.79</td>
<td>0</td>
<td>2.23</td>
<td>0.16</td>
</tr>
</tbody>
</table>
Table 8 shows the calculated transfer rate by country (and hence by operators: Germany (D2) Netherlands (Vodafone), Sweden (Telia) and Spain (Telefonica Moviles). Transfer rates are show in kiloBytes per second (kB/s).

Table 8 transfer rate by country.

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>1.31</td>
<td>0.33</td>
<td>2.48</td>
<td>0.81</td>
</tr>
<tr>
<td>Netherlands</td>
<td>0.97</td>
<td>0.01</td>
<td>3.24</td>
<td>0.93</td>
</tr>
<tr>
<td>Spain</td>
<td>1.13</td>
<td>0.01</td>
<td>3.26</td>
<td>0.96</td>
</tr>
<tr>
<td>Sweden</td>
<td>1.52</td>
<td>0.14</td>
<td>5.09</td>
<td>0.93</td>
</tr>
<tr>
<td>Total</td>
<td>1.3</td>
<td>0.01</td>
<td>5.09</td>
<td>0.99</td>
</tr>
</tbody>
</table>

Table 9 Transfer rate (non-empty) session, by country by UMTS

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Netherlands</td>
<td>0.67</td>
<td>0.01</td>
<td>1.29</td>
<td>0.44</td>
</tr>
<tr>
<td>Spain</td>
<td>1.5</td>
<td>0.01</td>
<td>3.26</td>
<td>0.95</td>
</tr>
<tr>
<td>Sweden</td>
<td>1.35</td>
<td>0.14</td>
<td>5.09</td>
<td>0.83</td>
</tr>
<tr>
<td>Total</td>
<td>1.31</td>
<td>0.01</td>
<td>5.09</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 10 Transfer rate (non-empty) session, by country by GPRS

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>1.31</td>
<td>0.33</td>
<td>2.48</td>
<td>0.81</td>
</tr>
<tr>
<td>Netherlands</td>
<td>1.16</td>
<td>0.01</td>
<td>3.24</td>
<td>1.1</td>
</tr>
<tr>
<td>Spain</td>
<td>0.38</td>
<td>0.02</td>
<td>1.01</td>
<td>0.31</td>
</tr>
<tr>
<td>Sweden</td>
<td>2.98</td>
<td>2.78</td>
<td>3.12</td>
<td>0.13</td>
</tr>
<tr>
<td>Total</td>
<td>1.28</td>
<td>0.01</td>
<td>3.24</td>
<td>1.1</td>
</tr>
</tbody>
</table>
Tables 11-13 show a further breakdown by individual trials.

**Table 11 Transfer rate (non-empty) session, by country by trial**

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/CARDIO</td>
<td>1.31</td>
<td>0.33</td>
<td>2.48</td>
<td>0.81</td>
</tr>
<tr>
<td>Netherlands/PREG</td>
<td>0.61</td>
<td>0.01</td>
<td>1.31</td>
<td>0.45</td>
</tr>
<tr>
<td>Netherlands/TRAUMA</td>
<td>2.31</td>
<td>0.11</td>
<td>3.24</td>
<td>1.03</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>0.79</td>
<td>0.01</td>
<td>1.54</td>
<td>0.49</td>
</tr>
<tr>
<td>Spain/Outdoors</td>
<td>1.25</td>
<td>0.02</td>
<td>3.26</td>
<td>1.04</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>2.15</td>
<td>0.66</td>
<td>3.12</td>
<td>0.85</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>1.47</td>
<td>0.3</td>
<td>2.49</td>
<td>0.9</td>
</tr>
<tr>
<td>Sweden/RESP</td>
<td>1.28</td>
<td>0.14</td>
<td>5.09</td>
<td>0.84</td>
</tr>
<tr>
<td>Total</td>
<td>1.3</td>
<td>0.01</td>
<td>5.09</td>
<td>0.99</td>
</tr>
</tbody>
</table>

**Table 12 Transfer rate (non-empty) session, by country by UMTS by trial**

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/CARDIO</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Netherlands/PREG</td>
<td>0.67</td>
<td>0.01</td>
<td>1.29</td>
<td>0.44</td>
</tr>
<tr>
<td>Netherlands/TRAUMA</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>0.83</td>
<td>0.01</td>
<td>1.54</td>
<td>0.48</td>
</tr>
<tr>
<td>Spain/Outdoors</td>
<td>1.86</td>
<td>0.44</td>
<td>3.26</td>
<td>0.94</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>1.6</td>
<td>0.66</td>
<td>2.39</td>
<td>0.66</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>1.47</td>
<td>0.3</td>
<td>2.49</td>
<td>0.9</td>
</tr>
<tr>
<td>Sweden/RESP</td>
<td>1.28</td>
<td>0.14</td>
<td>5.09</td>
<td>0.84</td>
</tr>
<tr>
<td>Total</td>
<td>1.31</td>
<td>0.01</td>
<td>5.09</td>
<td>0.89</td>
</tr>
</tbody>
</table>

**Table 13 Transfer rate (non-empty) session, by country by GPRS by trial**

<table>
<thead>
<tr>
<th>Country</th>
<th>Ave</th>
<th>Min</th>
<th>MAX</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany/CARDIO</td>
<td>1.31</td>
<td>0.33</td>
<td>2.48</td>
<td>0.81</td>
</tr>
<tr>
<td>Netherlands/PREG</td>
<td>0.55</td>
<td>0.01</td>
<td>1.31</td>
<td>0.45</td>
</tr>
<tr>
<td>Netherlands/TRAUMA</td>
<td>2.31</td>
<td>0.11</td>
<td>3.24</td>
<td>1.03</td>
</tr>
<tr>
<td>Spain/Nurse</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>Spain/Outdoors</td>
<td>0.39</td>
<td>0.02</td>
<td>1.01</td>
<td>0.31</td>
</tr>
<tr>
<td>Sweden/Remote</td>
<td>2.98</td>
<td>2.78</td>
<td>3.12</td>
<td>0.13</td>
</tr>
<tr>
<td>Sweden/Lighthouse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sweden/RA</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
CLASSIFICATION OF SESSION TYPES

Table 14 show the reason for end of session, where User STOp implies normal termination and System stop incidated interruption of communication between BAN and BeSys.

Table 14 Reason for end of session

<table>
<thead>
<tr>
<th>Reason Reason</th>
<th>Total</th>
<th>Of which, GPRS</th>
<th>Of which, UMTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>STOP (USER) (ie normal termination)</td>
<td>304</td>
<td>181</td>
<td>123</td>
</tr>
<tr>
<td>STOP (SYSTEM) (ie interruption of BAN-BESYS comms)</td>
<td>89</td>
<td>45</td>
<td>44</td>
</tr>
</tbody>
</table>

Further analysis of abnormal sessions yields eight classes of abnormal session, shown in Table 11 below. Table 15 also shows the incidence of normal and abnormal sessions in the logged data.

Table 15 Normal and Abnormal sessions

<table>
<thead>
<tr>
<th>Type of session (N=normal, AB=abnormal)</th>
<th>event sequence</th>
<th>interpretation</th>
<th>XREF external sources</th>
<th>NUMBER SESSIONS</th>
<th>Of which, GPRS</th>
<th>Of which, UMTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>START -&gt; INTERMEDIATE * (1...n) -&gt; STOP(USER) -&gt; EXIT</td>
<td>normal session</td>
<td></td>
<td>170</td>
<td>108</td>
<td>62</td>
</tr>
<tr>
<td>AB1</td>
<td>START -&gt; STOP(SYSTEM) -&gt; EXIT</td>
<td>comms failure: modem (AB1a) network (AB1b)</td>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>AB2</td>
<td>............ STOP(USER) -&gt; INTERMEDIATE * (1...n) ............</td>
<td>possibly flushing data, implies insufficient bandwidth</td>
<td></td>
<td>132</td>
<td>67</td>
<td>65</td>
</tr>
<tr>
<td>AB3</td>
<td>Any incomplete prefix of normal session</td>
<td>User error AB3a</td>
<td>iPaq battery failure AB3b</td>
<td>31</td>
<td>14</td>
<td>17</td>
</tr>
<tr>
<td>AB4</td>
<td>START -&gt; STOP -&gt; EXIT</td>
<td>Failure in bandwidth allocation</td>
<td>Cross check with BDR. Indication: No</td>
<td>47</td>
<td>32</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Event sequence</td>
<td>Description</td>
<td>Mobi failure:</td>
<td>Corresponding BDR session</td>
<td>Notes</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----------------</td>
<td>-------------</td>
<td>--------------</td>
<td>---------------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>AB5</td>
<td>INTERMEDIATE -&gt; STOP</td>
<td>Mobi failure: comms (AB5a) battery (AB5b)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AB6</td>
<td>No session present in EDR, where paper log shows session</td>
<td>Network failure</td>
<td>Paper logs from trial owners</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>AB7</td>
<td>No session present in EDR, where paper log shows session</td>
<td>Clock synch anomaly (aka timestamp anomaly)</td>
<td>Cleaned from logs</td>
<td>12</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

**Notes:****

Event set = {START, STOP, INTERMEDIATE, EXIT} (event UPLOAD not explicit in logs)
STOP is classified into User or System.
STOP (SYSTEM) indicates termination of BAN-BeSys communication
EXIT indicates termination of BAN app

Notation used:
- X * (1…n) one or more repetitions of X
- X -> Y Subsequence of session
- X -> Y event X precedes event Y with no intervening events
- Timestamp (X) date-timestamp of the event X
- a | b | c alternative interpretations

**GUI is supposed to enforce traces {N + AB1}**

**BATTERY LIFE**

The logs record battery level of MBU only. The battery Levels of the Sensors Front End and GPRS Modems were not accessible to the system. Table 16 shows the incidence of low battery levels over GPRS and UMTS sessions.

**Table 16: Low battery level**

<table>
<thead>
<tr>
<th>Number of sessions where battery level was lower than 5%</th>
<th>Of which GPRS</th>
<th>Of which UMTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>7</td>
<td>3</td>
</tr>
</tbody>
</table>
5.3 Measurements Evaluation

In this section the details of the measurement results evaluation are presented. The executed performance measurements gave the set of raw data as an outcome. Outcomes are random variables, and they can be different each time the measurements are executed. Due to this fact the statistical analysis (e.g. variability) of gathered data is of importance.

The evaluation process has been structured as follows. Firstly the (raw) measurement results were collected and (roughly) validated; the statistical data based on the data obtained from each individual experiment were generated. Specialized analysis tools and programs exist for the purpose of data analysis e.g. Matlab, but due to the specific nature of the measurement data, that needed to be analyzed, the module of the automatic data processing was developed as the java application. Java application delivers particular functionality: statistical analysis of the raw data and raw data visualization. Particularly, the functionality of application is as follows:

1. measurement data files retrieval from the client / server / proxy evaluation systems
2. automatic match of the corresponding client / server / proxy measurement data files
3. based on the data retrieved and matched, capability of automatic calculation of: the upstream, downstream and round-trip delay performance measure the mean and standard deviation of the delay(s) the delay(s)’s obtained accuracy for a given confidence interval
4. automatic raw data visualisation
5. capability of storage (for later retrieval) of visualized raw data plus its statistical analysis

The visualized raw data (plus its statistical data), gathered for all executed measurements, are presented in Volume 2 of this report.

Based on the raw data graphs and the statistical data carried out by java application, measurements have been evaluated. This first-level evaluation was for a purpose of taking the decision to keep the data or re-execute the measurements has been taken. It occurred that the worst case scenario mentioned in section 5.2.1 (i.e. “If anything can go wrong, it will…”) has been accomplished and the whole set of measurements needed to be redone. Due to the time constraint only the majority, not all of the measurements were redone. Particularly problems with clock synchronization / clock drift, and the unconfirmed test case procedure execution have caused the measurements repetition. Observations done during the measurements evaluation, problems and some proposed solutions are highlighted in the next sections.

Number of observations

After the first round of measurements the empirical number of 500 observations was verified. The empirical value was determined such that sample measurement data is statistically sound. There were dedicated measurements taken in experiment 1, SE packet size: 524B, 524B for 500 and 2500 observations in order to see the influence of increasing the number of observations on the accuracy of results. Figure 5.57 shows that there is no significant improvement in statistical data accuracy for 2500 observations comparing to 500, so the number of 500 remained for the
rest of the measurements. Moreover, for the big observation number, the problem with time synchronization of the system occurs.

Figure 5.57  500 and 2500 observations and their statistical characteristics.

Inband time synchronisation

Recall 5.1.3, the inband time synchronization consists in synchronizing of the systems clocks with the time source inside the SoD and over the same interface, over which the SoD is stimulated.

There were no problems encountered with the time synchronization or clock drifts when the systems clock synchronization has been executed between the measurement activities (i.e. system is idle). When synchronization was done while traffic generation, there was a significant clock drift on both systems observed (see example Tardis log file at Figure 5.58).
The reason of experienced problem with clock instability was that the synchronization over the wireless link has been negatively influenced due to unpredictable behaviour of this communication link – especially when link has been asymmetrical and heavily loaded. Moreover, when using Tardis on the notebook the high variance of the clock drift has been noticed (see example Tardis log file at Figure 5.59). This can be explained as the effect of the high fluctuations of temperature – inside the notebook and also external temperature (heating system in Zilvering building is operational from 8.00-18.00).

Outband time synchronization

Recall 5.1.3, the outband time synchronization consists in synchronizing of the systems clocks with the time source outside the SoD and over the dedicated interface. There were no problems encountered with the time synchronization or clock drifts when the synchronization has been done in between measurement activities (i.e. system is idle). When synchronization was done while traffic generation, there was a significant clock drift on both systems observed (see example Tardis log file at Figure 5.60).
The outband synchronization has been negatively influenced due to unpredictable behaviour of the synchronized systems – especially when synchronization was done when systems were heavily loaded. This could be caused by the different processes’ priorities running under the Windows OS\(^{45}\). The priority of the running evaluation system instance had could be higher than the priority of the Tardis process, but it was not possible to change by the measurements executors. The partial solution for this problem was found. The clocks of computers systems involved in measurements were upgraded automatically at scheduled basis every 2 hours instead of 10 minutes proposed before. The idea was to synchronize systems in between measurement activities; such synchronization correctness would be ensured. Once again problems were encountered. Test results were negatively influenced by clock drift. Synchronization time interval of 2 hours has been too less frequent.

Problem encountered with computer systems clocks were the main reason of repetition of the whole set of measurements. Summarizing, the outband time synchronization was more accurate than the inband one. At the Figure 5.61 the visualized raw data is given, where measurements “a” were taken while inband synchronization method has been practiced, and “b” – while the outband one. The improvement of the synchronization method is seen in better clock accuracy and less drift observed.

a) inband clocks synchronization

b) outband clock synchronization
During the unconfirmed service type execution while delivery of evaluation service, there was a problem with a high probability of the service delivery failure, encountered. This has to do with the way the unconfirmed test type has been executed. Namely, first for a particular link saturation factor, the maximum rate, at which the packets are generated, has been calculated. Then when the test has been started, the packets were generated to the SoD with calculated maximum rate (Figure 5.62), which caused the high probability of the terminal failure (i.e. V3GNL disconnection). The maximum packet rate is calculated with respect to the maximum available bandwidth, with the protocol overhead taken into account. The number of slowstart packets is given accordingly to the formulas at Figure 5.62.

Output buffer queue overrun happened when the workload generator generated SE’s (i.e. packets) long enough to overrun the UMTS terminal’s output queue. Based on following system observations and knowledge on the network behaviour (bearer switch) it has been concluded, that the calculated rate, at which packets are offered to SoD is too high comparing to the rate system (i.e. terminal) is able to handle. When data has been generated with a maximum rate, it was buffered by the terminal, before it could be handled by the network (i.e. before the network switches from common to dedicated bearer).

There was a need of introducing the slow start mechanism at the java application level, such each data transfer (with maximum rate) phase would be preceded by the phase with small rates (started from 1 to maximum). Slowstart phase functioned as an initialization phase for the actual performance measurements. There is one very important slowstart parameter, which needs to be mentioned, namely the SLOWSTART_BACKOFFTIME parameter. After executing the slowstart by generating the number of packets with the rate from 1 to the maximum-1, the workload generator stops for the SLOWSTART_BACKOFFTIME [ms] before the generation of

---

46 to flow over [Webster]
packets with the maximum rate. This is done in order to keep the probability of UMTS terminal output queue overrun as low as possible; in order to empty the buffer from the slowstart packets. But the SLOWSTART_BACKOFFTIME [ms] is low enough not to lose the UMTS connectivity and assigned bearer in the system.

```
/*
   * protocol layer overhead for PPP based communication
   */
private static final int TCP_OVERHEAD = 32;
private static final int IP_OVERHEAD = 20;
private static final int PPP_OVERHEAD = 8;

maxPacketRate = Server.UP_LINK_CAP / (packetSize + protocolOverhead)
```

During the small rate phase, the network can switch from the common bearer to the dedicated one such the data with the maximum rate can be handled. This allows avoiding the overloading of the Nokia’s UMTS terminal send buffer and its eventual failure. Moreover, for the case when the slow start is not implemented, the experienced behaviour of the network is unpredictable; high variance of delay can be intercepted, such it is shown at the example data at Figure 5.63, where the first observation had been implemented with the slow start scenario, and the behaviour of the network is experienced as stable, while the second observation had been implemented without the slow start, and long delays are seen while starting data generation at the maximum rate.
Figure 5.63  Slow start mechanism’s influence at experienced delays.
6 Performance Modelling and Evaluation

The focus of this chapter is on the modelling and evaluation methodologies phase presented in sections 3.1 and 3.2, and covers steps 8 an 9 of measurement methodology (Figure 6.1 (a)) and 7, 8 and 9 of modelling methodology (Figure 6.1 (b)) and therefore the essence of the evaluation methodology.

<table>
<thead>
<tr>
<th>Measurement Methodology Phase 3.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. State the Goals and System Definition</td>
</tr>
<tr>
<td>2. List Services and their Outcomes</td>
</tr>
<tr>
<td>3. Select Performance Criteria (i.e. Metrics):</td>
</tr>
<tr>
<td>4. List System and Workload Parameters</td>
</tr>
<tr>
<td>5. Select Factors and their Levels</td>
</tr>
<tr>
<td>6. Select the Workload</td>
</tr>
<tr>
<td>7. Design and Execute the Experiments</td>
</tr>
<tr>
<td>8. Analyze, Evaluate and Interpret the Data</td>
</tr>
<tr>
<td>9. Present the Results</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modelling Methodology.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. State the Goals and System Definition</td>
</tr>
<tr>
<td>2. List Services and their Outcomes</td>
</tr>
<tr>
<td>3. Select Performance Criteria (i.e. Metrics)</td>
</tr>
<tr>
<td>4. List System and Workload Parameters</td>
</tr>
<tr>
<td>5. Select Factors and their Levels</td>
</tr>
<tr>
<td>6. Select the Workload</td>
</tr>
<tr>
<td>7. Create Model</td>
</tr>
<tr>
<td>8. Parameterize the Model</td>
</tr>
<tr>
<td>9. Validate and Verify the Model</td>
</tr>
</tbody>
</table>

In this chapter the results of the performance evaluation of the MobiHealth transport system are presented. The interpretation of the experiments’ numerical outcomes in order to get an answer to the initially posted research questions is done. This activity commences of the modelling of the transport system based on the obtained measurement results. Firstly, the decision on the abstract representation of the system is taken i.e. choice on the model which yields delay metrics (identified in chapter 4) is taken. Development of the high-level abstract model contains the description of the system in the modelling language; description of delay-related relations in the system. Model is parameterized and then validated based on the measurement results.
After the modelling, performance evaluation of the MobiHealth transport system based on the measurement results and developed model, follows. Evaluation consists of extensive analysis of the system with respect to the different aspects of the system. Firstly the system’s delay characteristics are studied, and then the bottleneck analysis with regards to delay follows. The scalability of the system with respect to the number of concurrent users is also examined.

*Note: In this chapter the terms “upstream” and “uplink” have the same meaning, the same is valid for the terms “downstream and “downlink”.

### 6.1 Transport System Model

The main objective of this section is to present the development of simple high-level model that captures the performance characteristics of the of the MobiHealth transport system. The analytical model and particularly the queuing model yielded the speed-related performance characteristics of the system defined in chapter 4. Moreover, this model has been derived based on the measurement results. Since it was enable to place the measurement probes along the data path in the network, measurements were limited to the end-to-end performance metrics (e.g. delay and throughput). As a consequence, the fitting of the measurements to the relative simple model that captured the end-to-end transport performance characteristics has been uncertain.

The low-level details of the transport system were ignored (due to the lack of access to the system insights), although future versions of the model may benefit from such refinements. For the more comprehensive performance analysis purpose probes inside the transport system would be needed.

Modelling question:

Can we model the speed-related performance characteristics of the transport system for different workload parameters?

There was a “black-box” view (see section 4) on the system as it is shown on Figure 6.2. Transport service is assumed to be available and reliable. Transport requests arrive to the system at its ingress point, are served with a particular speed and departure from the system at its egress point at the same order.
Performance criteria and parameters

Recall section 4.5.2., there is a primary speed criterion-related system performance parameter defined such: delay; which is time taken for the transport service to be completed i.e. system response time in modelling language. Derived performance parameters are: “jitter” - delay variability i.e. system response variability and “goodput” - the rate at which the service can be performed i.e. number of completed transport requests in time. Regarding the required level of detail, parameters’ average values and variances (only for the primary ones) were assumed to be of the vital importance while developing the system model.

System and workload parameters

There were two kinds of parameters which had influenced the transport system’s performance: system parameters and system usage (i.e. workload) parameters. The model has been conducted for the system and workload parameters, as indicated in the experiments 1 and 9 (see section 5.2.1). Results from the experiment 1 (confirmed service type) indicate the quantitative behaviour of the system while there is no contention, while the results from experiment 9 (unconfirmed service type) give the system behaviour when contention occur.

Queuing Model

Transport system was modelled as the queuing system consisting of multiple-server queue. A high-level diagram of the transport system model is presented on Figure 6.3. Packets of fixed size arrive at ingress point of the transport system with a fixed frequency, they are transported through the system and they departure from the system at the egress point with a particular frequency. The transport request completion time is seen as the system response time, which equals at minimum the packet service time.
Identification of instances

In this section the results on the analytical analysis of the transport system are given. The system’s queuing model is fully characterized by the following:

1. **Arrival process**: packets (of given (fixed) size) arrive to the transport periodically at the fixed rate, which means that the arrival process is deterministic.
2. **Service time distribution**: service time is a random variable which assumed to be IID (Independent and Identically Distributed) and generally distributed. Service time corresponds to a random delay through the network.
3. **Number of servers**: there are \( m \) \((m \geq 1)\) servers in the transport system.
4. **Buffer size**: is assumed a finite/fixed buffer size (i.e. waiting positions) for arriving packets into the system.
5. **Population size**: there is assumed to be infinite because the arrival rate is fixed. There is infinite total number of potential packets who can ever come to the transport system (i.e. open queuing network is considered). This is the default value of the population size parameter.
6. **Service discipline**: the order at which packets are served at the serving entity is First Come First Served (FCFS). This is the default value of the service discipline parameter.

According to the Kendall’s notation the queue representing the transport system is a type of: \( D/G/m/B/\infty/FCFS \), which means: a deterministic packet inter arrival time, general (arbitrary) service time, \( m \) serving entities, finite buffer size \((B)\), infinite population size and FCFS serving discipline. Moreover, for the default model parameters’ values it’s not necessary to indicate them in Kendall’s notation, thus our queue notation can be reduced to: \( D/G/m/B \).

The key variables used in the analysis of the developed queuing model are as presented at the Figure 6.4 and they are:

\[ \text{Error! Objects cannot be created from editing field codes.} \] : arrival rate
\[ \text{Error! Objects cannot be created from editing field codes.} \] : interarrival time, which is time between two successive arrivals, \( = \frac{1}{\text{Error! Objects cannot be created from editing field codes.}} \)
\[ s \] : service time which is transport delay of one packet in the transport system, characterized by the mean and variance; it is a random variable which represents the service time or transport delay of one packet.
\[ \frac{1}{\bar{s}} \]: service rate, 
\[ \rho \]: system utilization; traffic offered to the system, \( \rho < 1 \) for stable system 
\[ n \]: stationary number of packets in the system in the steady-state (i.e. number of packets transported and those waiting in the queue) 
\[ n_q \]: number of packets waiting for transport in the input (arrival) buffer 
\[ n_s \]: number of packets in transport (\( n_s \geq 1 \)) 
\[ r \]: system response time (i.e. packet waiting time and transport delay; \( r = w + s \)) 
\[ w \]: waiting time, that is, the time interval between packet arrival time and the instant its transport begins

\[ \lambda \mu \tau \]

\[ n_q \]

\[ \text{Figure 6.4 Random variables used in analysis a queuing system.} \]

**Parameterisation of the transport system queuing model**

In this section there is parameterization process of the model presented. Transport system has been identified as the \( D/G/m/B \) queuing model. The complication is that the \( m \) and \( B \) are unknown. The question is if is possible to derive these parameters from the measurement results. In order to do that, particular assumptions needs to be taken for a model:

1) **Generally distributed arrival process**: packets (of given (fixed) size) arrive to the transport periodically at the fixed rate, which means that the arrival process is deterministic, which is just a particular case of the general arrival. Thus, for simplicity reason, the generally distributed arrival process is assumed.

2) **Infinite buffer size**: it is assumed to be infinite buffer size (i.e. waiting positions) for arriving packets into the system. Traffic offered to the system is sufficiently low comparing to the system buffer size, so the buffer size is assumed to be infinite.
Moreover, even if the buffer is finite, the packets are not lost, thus assumption of infinity is valid. This is the default value of the buffer size parameter.

Based on the assumptions, transport system queuing model can be parameterized as $G/G/m/\text{inf}/\text{inf}/\text{FCFS}$, which in short is called $G/G/m$ model. Moreover, this particular queuing model cannot be parameterized with details, therefore, following the literature [Klei1976], only the bounds (e.g. for the waiting time in the system) on system behaviour are given. The upper bound and lower of the waiting time in the $G/G/m$ queue:

$$
\frac{\lambda \cdot \frac{\sigma_r^2}{m^2} - \overline{s} \cdot (2 - \rho)}{2 \cdot (1 - \rho)} - \frac{(m-1) \cdot \sigma_r^2}{2 \cdot m \cdot \overline{s}} \leq W \leq \frac{\lambda \cdot \left(\frac{\sigma_r^2}{m^2} + \frac{(m-1) \cdot (\overline{s})^2}{m^2}\right)}{2 \cdot (1 - \rho)}
$$

Figure 6.5 Packet waiting time bounds.

Where:
- $\sigma_r^2$ : variance of the interarrival time
- $\overline{s}$ : mean service time
- $m$ : variance of the service time
- $\rho$ : server utilization; traffic offered to the system, $\frac{\overline{s}}{\overline{r} \cdot m} = \frac{\lambda \cdot \overline{s}}{m}$

Parameterisation of the model consists in finding the number of servers ($m$) in the transport system based on the measurement results. Parameterization is done based on the results for experiments: 1 and 9 (see section 5.2.1).
The following assumptions were taken into account while parameterization of the model:

1) only upstream SUT behaviour

2) only one fixed packet size i.e. 524 [B] (4192 [b])

3) only discrete system contention levels for which there is a job flow balance in the system; i.e. number of arrivals is being equal to the number of departures from the system

Measurement results are summarized in the following table (Figure 6.7) For the arrival rate of the packets up to 12, the system behaves identically regardless the number of packets being served.
Mean packet service time $\bar{s}$ [s] | Mean service rate per packet $47$ [kbps] | Packet arrival rate | Mean service rate $48$ [kbps] | System utilization
\[ \rho = \frac{\lambda \cdot \bar{s}}{m} \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>3</th>
<th>12,28</th>
<th>0,552 / m</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>28,65</td>
<td>1,288 / m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>32,75</td>
<td>1,472 / m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>36,84</td>
<td>1,656 / m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11</td>
<td>45,03</td>
<td>2,024 / m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>49,125</td>
<td>2,208 / m</td>
</tr>
</tbody>
</table>

the condition on job flow balance does not hold:
\[ \text{Error! Objects cannot be created from editing field codes.} \]
\[ \geq 14 \]
increasing system response time

Figure 6.7 Measurement results used for a model parameterization.

It is concluded that for the $\leq 12$ the system behaves stable ($\rho < 1$), and there is no queue observed. That implies that there are at least $m \geq 3$ servers in the system. For the $= 13$ there are no measurement results available. For the $\geq 14$ the system behaves stable, but the building of the queue is seen. The system response time is increasing (i.e. waiting time is increasing) until the point where the application is stopped from sending data to the transport system by the TCP congestion avoidance algorithm. In this case send data see the maximum response time (maximum waiting time plus service time). The developed model is not taking into account this behaviour of the system, but the further model refinements can benefit from that.

In order to support the reasoning of multiple-server queue behaviour of the transport system, the waiting time, based on the formula given on Figure 6.5, has been calculated for the cases when arrival rate $\leq 12$.

---

$47$ Mean Service Rate per Packet = $\frac{4192}{\bar{s} * 1024}$ [kbps]

$48$ Mean Service Rate = $\frac{4192 \cdot \lambda}{1 \cdot 1024}$ [kbps]
The minimum waiting time gave for all cases negative result, which is a trivial conclusion and is not presented in the table. Obtained from the formulas maximum waiting time, which is less than 0,5% of the service time value, confirm the observation that there is hardly any queuing in the network. Therefore the conclusion is, that the proposed G/G/3 model can be applied to the transport system. The validation of the model has been not possible due to the fact that there was no other data, than for packet size 524B, gathered from experiment 9.

**Conclusion**

The question given at the beginning of this section has been answered positively. Namely it is possible to model the speed-related performance characteristics of the transport system for different workload parameters (particularly regarding the different packets rate). However presented G/G/3 model reflects the transport system behaviour for the case when there is no contention in the upstream direction (low packet rate). Therefore, there could be of interest to analyze the behaviour of the system for the case when it gets contented. Moreover, the behaviour of the system for other than 524B packet sizes could be analyzed in order to validate the model. Furthermore, the downstream direction analysis needs to be done.
6.2 Delay Analysis

The delay analysis of the SUT aims to fulfil the first and second objective described in section 4.1. Recall these objectives:

1) To characterize the behaviour of this system as a MobiHealth transport service.
2) To determine the maximum BANip packet size and packet rate for a specified (maximum) delivery time.

In chapter 4 and 5 we explained that workload generator SE’s are used to simulate BANip packet sizes. For the delay analysis of the SUT we use the results obtained from experiment 1\(^{49}\); the “benchmark” for all other performance evaluation activities. The first and second objective are combined and incorporated into one question that focuses on the goal of the delay analysis:

What is the influence of the “user confirmed” SE’s size on the delay behaviour of the SUT “uplink” and “downlink”?

The SUT delay analysis activities are divided into four steps:

1) Presentation of “conclusive” graphs (section 6.2.1).
2) Description on how these graphs are derived (section 6.2.2).
3) Quantitative analysis of the conclusive graphs (section 6.2.3).

The influence of system parameters on the delay behaviour of the SUT is described in section 6.2.4.

6.2.1 Conclusive Graphs

The workload generated for the SUT consists of “user confirmed” SE’s with different SP sizes for “uplink” (SP_u) and “downlink” (SP_d), effectively disclosing the specific SUT behaviour of these links. The SE’s generated have a fixed SP_u size (SP_u\(_{\text{fixed}}\)) and variable SP_d sizes (SP_d\(_{\text{variable}}\)). The SP_u\(_{\text{fixed}}\) sizes range from 174 up to 8122 bytes (rows of the 20x20 matrix) and the SP_d\(_{\text{variable}}\) sizes range from 174 up to 48208 bytes (columns of the 20x20 matrix). The “nuts and bolts” on the generation of the different SE’s are explained in section 5.1.2.

Figure 6.9 presents the influence of the “user confirmed” SE’s size on the delay behaviour of the SUT’s “uplink”. On the horizontal axis the values of the SP_d\(_{\text{variable}}\) component are presented. The vertical axis presents the statistical average of the primary performance parameter \(\text{delay}\)\(^{50}\) (section 4.5.2) for a SP_u\(_{\text{fixed}}\) size in relation to a SP_d\(_{\text{variable}}\) size.

\(^{49}\) Experiment 1 uses “user confirmed” SE’s as workload; however the BANip application protocol is based on a “user unconfirmed” service. We consider a “user confirmed” service as a combination of two “unconfirmed services”, one for “upstream” and one for “downstream” oriented communication services.

\(^{50}\) Statistical random variable.
In Figure 6.10 the influence of the “user confirmed” SE’s size on the delay behaviour of the SUT’s “downlink” is presented. On the horizontal axis the values of the SP_d variable component are presented. The vertical axis presents the statistical average of the primary performance parameter \(delay\) for a SP_d variable size in relation to a SP_u fixed size.
From a MobiHealth system perspective the SUT “uplink” behaviour is more important than the “downlink” behaviour (recall the producer/consumer flow of data in section 2.1). Furthermore, Figure 6.9 and 6.10 showed that the behaviour of the SUT “downlink” is less predictable than the behaviour of the “uplink” due to the changes in bearers. Hence, we focus for the jitter (derived performance parameter from delay) analysis on the SUT “uplink” only. Figure 6.11 presents the SUT “uplink” delay and jitter behaviour for a variable size SP_u and a fixed size (524 bytes) SP_d. The jitter is indicated as vertical red lines for each delay measurement point.

![Graph showing SUT “uplink” delay and jitter behaviour for SP_u variable and SP_d524.](image)

**6.2.2 Derivation of Conclusive Graphs**

The execution of Experiment 1 is described in section 5.2.1. The successfully executed cells in the 20x20 matrix are presented in green colour in Figure 5.48 and represent 500 observations (equivalent to one sample) of a SE execution. The first 5 observations are removed from the sample to eliminate the initial changing behaviour of the links.\(^{51}\) The remaining observations are used in the statistical calculations for the average delay. The changes in behaviour will have therefore an impact on the variance of the delay (i.e. jitter) for both “uplink” and “downlink”.

**Graphs**

For all individual samples “raw data” graphs were generated (Volume 2, T01 graphs directory). Statistical calculations were performed to obtain the average (i.e. mean), standard deviation and accuracy for a 95% confidence interval for the random variable delay. Figure 6.12-15 present

\(^{51}\) Change from the common bearer to the first dedicated bearer.
four “raw data” graph examples that will be used to explain the delay behaviour of the SUT and the high variance of the average delay.

Figure 6.12 SUT “uplink” converging behaviour.

Figure 6.13 SUT “up/downlink” accurate behaviour.

Figure 6.14 SUT “downlink” altering behaviour.

Figure 6.15 SUT “downlink” converging behaviour.

Figure 6.12 presents “uplink” converging behaviour of the SUT for a SE that consists of SP_u174 and SP_d174 (both SP’s have a size of 174 bytes). After ~250 observations the SUT changes from the common bearer to dedicated bearer 1. The “uplink” behaves accurate (i.e. low variance of the average delay) after it changed to dedicated bearer 1. The change has an impact on the variance of the delay for both “uplink” and “downlink”. In Figure 6.13 both links are converged to dedicated bearer 1. They display predictable delay behaviour and as a result have a low variance and high accuracy. Figure 6.14 presents “downlink” altering behaviour. The “downlink” is changed to dedicated bearer 2 (observations 10 to ~120), changed back to dedicated bearer 1 (observations ~120 to ~240) and changed back again to dedicated bearer 2 (observations ~240 to 500). If however the “downlink” is changed to a dedicated bearer, it behaves accurate. In Figure 6.15 “downlink” converging behaviour is presented; the link is changing from dedicated bearer 1
(observations 10 to ~80) to 2 (observations ~80 to ~360) and converges to 3 (observations ~360 to 500).

*Note:* SUT bearer changes are “proofed” by calculation of the link “goodput” in terms of net obtained bits per second and correlation of the “goodput” to the theoretical throughput of a bearer (recall section 4.3.2, Figure 4.8).

**Statistical data**

Statistical data of all samples are collected in a *StatDataFile*. A section of this file is presented in Figure 6.16 (D1 = “uplink” delay, D2 = “downlink” delay, SE_D = SE delay, Ac = accuracy, Ul = “uplink”, Dl = “downlink”). Row “174.174” is the data used to create the “raw data” graph in Figure 6.12. The conclusive graph in Figure 6.9 is generated from column 2 (SP_d) and column 3 (avg D1). Figure 6.10 is generated from column 2 (SP_d) and column 4 (avg D2). Both figures are using the average *delay* number regardless of the variance.

Figure 6.11 is generated from rows containing SP_d sizes 524\(^{52}\) and the accuracy numbers presented in column 9 (Ac D1). The *jitter* is however presented as a vertical (red) line with a length indicating delay values between a certain minimum and maximum value. The minimum and maximum values are calculated with the accuracy number presented in column 9. For example: The average “uplink” delay for SP_u174-SP_d524 is 124 ms and the accuracy of this measurement is 48%. The jitter ranges from 124*(1-0.48) = 65ms to 124*(1+0.48) = 184ms.

<table>
<thead>
<tr>
<th>SP_u</th>
<th>SP_d</th>
<th>avg D1</th>
<th>avg D2</th>
<th>avg SE_D</th>
<th>std SE_D</th>
<th>std D1</th>
<th>std D2</th>
<th>avg Ac</th>
<th>Ac D1</th>
<th>Ac D2</th>
<th>Ac SE_D</th>
<th>Ul_goodput</th>
<th>Dl_goodput</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>174</td>
<td>430</td>
<td>112</td>
<td>543</td>
<td>311</td>
<td>185</td>
<td>361</td>
<td>142</td>
<td>323</td>
<td>130</td>
<td>3</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>174</td>
<td>349</td>
<td>133</td>
<td>110</td>
<td>243</td>
<td>21</td>
<td>28</td>
<td>33</td>
<td>30</td>
<td>49</td>
<td>27</td>
<td>10</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>174</td>
<td>524</td>
<td>124</td>
<td>143</td>
<td>267</td>
<td>30</td>
<td>59</td>
<td>66</td>
<td>48</td>
<td>81</td>
<td>48</td>
<td>11</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>174</td>
<td>1048</td>
<td>124</td>
<td>201</td>
<td>325</td>
<td>21</td>
<td>45</td>
<td>50</td>
<td>32</td>
<td>44</td>
<td>30</td>
<td>11</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>174</td>
<td>2096</td>
<td>125</td>
<td>385</td>
<td>510</td>
<td>21</td>
<td>244</td>
<td>246</td>
<td>34</td>
<td>124</td>
<td>94</td>
<td>11</td>
<td>44</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.16  Example of statistical data.

The statistical data in the *StatDataFile* was also used to generate graphs presenting the SUT “uplink” and “downlink” delay behaviour for SE’s consisting of a fixed sized SP_u and variable sized SP_d (Volume 2, T01 conclusive graphs directory). Figure 6.17 presents an example for SE’s consisting of SP_u174 and SP_d\(_{\text{variable}}\):

- **red line (bottom):** SP_u delay analysis for SP_u174 and SP_d\(_{\text{variable}}\).
- **green line (mid):** SP_d delay analysis for SP_u174 and SP_d\(_{\text{variable}}\).
- **blue line (top):** SE delay analysis for SP_u174 and SP_d\(_{\text{variable}}\).

\(^{52}\) The size of the SP_d is not influencing the SUT uplink behaviour (section 6.2.3 – downlink delay analysis).
6.2.3 Quantitative Analysis

The quantitative analysis of the SUT is divided into two sections: uplink and downlink.

**uplink delay analysis**

The quantitative delay analysis is based on Figure 6.11 and lead to the following conclusions:

1) **SP_d size does not influence the SUT “uplink” delay behaviour for a fixed SP_u sizes, except for SP_d size 174: after ~250 observations the SUT changed from the common bearer to the first bearer).**

   Based on a quantitative analysis of 360 raw data graphs, we conclude that the size (of SE) and rate of data offered to the “uplink” and “downlink” cause the changing behaviour of the links. This changing behaviour is considered to be normal if it converges to a dedicated bearer and this bearer is going to be used for the remaining part of a data transfer (Figure 6.12, 6.13 and 6.15). In 99.8% of the executed performance measurements this behaviour is validated. Only 0.2% showed an altering behaviour of the downlink (Figure 6.14).

2) **SUT “uplink” delay behaviour is increasing linearly with the SP_u size (Figure 6.11).**

3) **SUT “uplink” jitter is significant for all sizes of SP_u’s because of bearer changes and “rare” events (i.e. 2% observations) that seem to have no frequency in their behaviour (Figure 6.12-15).**
downlink delay analysis

The quantitative analysis based on Figure 6.10 lead to the following conclusions:

1) SP_d size does influence the SUT “downlink” delay behaviour for a fixed SP_u sizes.
2) SUT “downlink” delay behaviour is not linearly increasing with the SP_u size, except for SP_d sizes larger then 23056 bytes.
3) SUT “downlink” delay behaviour has remarkable events for three SP_d sizes: 2096 (E1), 10480 (E2) and 23056 bytes (E3).

   E1: SP_d sizes between 174 and 2096 bytes are independently of SP_u size served by the SUT’s first dedicated bearer.
   E2: SP_d sizes between 2096 and 10480 bytes are independently of SP_u size served by the SUT’s first or second dedicated bearer.
   E3: SP_d sizes between 10480 and 23056 bytes are independently of SP_u size served by the SUT’s second or third dedicated bearer, and SP_d sizes larger then 23056 bytes are independently of SP_u size served by the SUT’s third dedicated bearer.

The bearer changing moment for the downlink seems to be related to the volume of data offered (size x rate). Analysis of the raw data did not clearly indicate the lower and upper boundaries of the volume.

6.2.4 Influence of System Parameters

The influence of the SUT structure and SoD structure system parameters defined in section 4.7 (Figure 4.22) are analysed for the primary performance parameter delay. For every SoD/SUT instance one system parameter of influence is investigated. The “benchmark” (experiment 1) will be used as a reference for all other measurements. Due to the importance of the SUT’s “uplink” behaviour, the scale of this measurement activity and the obtained large volume of raw data (available in Volume 2) we focus only on the SUT’s “uplink” and “downlink” behaviour for one particular SEs: SP_u524, SP_dvariable.

We pose the following five questions:

1) What is the influence of changing the intra communication system from USB to Bluetooth?
2) What is the influence of changing the UMTS terminal from USB-Nokia 6650 to PCMCIA-PCcard?

Note: We abstract from the USB and PCMCIA intra communication systems behaviour and consider these systems as a part of the UMTS terminals.
3) What is the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth?

4) What is the influence of making the Internet component “present” in the SoD structure?

5) What is the influence of changing the SoD transport service and the application protocol buffer size?

The answers to the five questions will be based on the analysis of relevant measurements data. Conclusive graphs will be generated based on relevant statistical data from the measurements\(^{53}\). Furthermore, “raw data” graphs from Volume 2 are used to explain the SUT’s “uplink” and “downlink” behaviour.

**Question 1:** What is the influence of changing the intra communication system\(^{54}\) from USB to Bluetooth?

Measurement results from experiment 1 (bm) and 5 (E5) are used to answer this question. Figure 6.18 and 6.19 present combined graphs for four experiments. The red “bm” line and the magenta “E5” line are relevant for answering this question.

*Note: The lines in Figure 6.18 and Figure 6.19 must be interpreted as “trend” lines passing through the measurement points.*

**conclusive graphs**

The red lines in Figure 6.18 and Figure 6.19 show the SUT behaviour for a USB intra communication system, and the magenta lines show the behaviour for the Bluetooth alternative. The vertical axis represents the “uplink” average delay for a SP_u size of 524 bytes. The horizontal line represents the different sizes of the SP_d.

\(^{53}\) Approach is similar to section 6.2.2, therefore we omitted a description.

\(^{54}\) The connection between the computer system and the Nokia 6650 UMTS terminal is based on USB.
uplink quantitative analysis

The difference between USB and Bluetooth seems to be significant for SP_d sizes of 174, 524 and 1572 bytes. Differences in the range of 20 to 40 ms are considered timing accuracy tolerances due to the instrumentation of the measurements (section 5.1.3). Hence, there is no
significant difference between USB and Bluetooth for other sizes. Figure 6.20, 6.21, 6.22 and 6.23 are used to explain the observed difference.

Figure 6.20 (SP_u524,SP_d174) shows changes in the “uplink” bearer at ~50 and ~150 observations. The same applies to Figure 6.21 (SP_u524,SP_d524), but the changes are at ~60 and ~230 observations. According to the “benchmark” (Figure 6.23, SP_u524,SP_d174), the average delay is 190.0 ms. The SUT “downlink” is not changing, which leads to the conclusion that no time synchronisation event could cause the “uplink” behaviour (section 5.3 “Problems encountered with inband time synchronisation”). The cause for the changing events is not clear, but we assume that the SUT encountered resources problems as a consequence of background load55.

Figure 6.22 (SP_u524,SP_d1572) shows the “uplink” operating with a high average delay (360.0 ms). According to Figure 6.23 the average delay is 190.0 ms. Analysis of the Volume 2 T06 graphs directory series 06_S01_524.*, reveals that the “uplink” average delay in Figure 6.22 is ~143.0 ms higher then the calculated average (225.0 ms) over eight SP_u524 samples. Hence, the SP_u524, SP_d1572 sample may be not accurate.

Answer 1.1: The influence of changing the intra communication system from USB to Bluetooth is negligible for SUT “uplink” behaviour; i.e. the behaviour of the USB and Bluetooth intra communication system is similar.

downlink quantitative analysis

The quantitative analysis based on Figure 6.10 lead to a conclusion that the SUT “downlink” delay behaviour has remarkable events for three SP_d sizes: 2096 (E1), 10480 (E2) and 23056 bytes (E3). These events are shown as down wards “arrows” with an “Event x” label in Figure 6.19. Recall the description of the events:

- **Event 1:** SP_d sizes between 174 and 2096 bytes are independently of SP_u size served by the SUT’s first dedicated bearer.
- **Event 2:** SP_d sizes between 2096 and 10480 bytes are independently of SP_u size served by the SUT’s first or second dedicated bearer.
- **Event 3:** SP_d sizes between 10480 and 23056 bytes are independently of SP_u size served by the SUT’s second or third dedicated bearer, and SP_d sizes larger then 23056 bytes are independently of SP_u size served by the SUT’s third dedicated bearer.

Up to the first event, the difference between USB and Bluetooth seems to increase with SP_d sizes. If we consider differences in the range of 20 to 40 ms as timing accuracy tolerances due to the instrumentation of the measurements (section 5.1.3), then only the difference (~120ms) for SP_d size 2096 bytes is significant.

---

55 The SUT was in commercial operation during this measurement activity. Background load could be caused by other service users.
The analysis of the measurement data for the second event is focused on SP_d size 8384 bytes. The “benchmark” experiment shows an average delay of 717.0 ms and for E5 experiment an average of 957.0 ms. Goodput calculations indicate that both measurement points are related to the second dedicated SUT bearer (“benchmark” ~94.0 Kbps and E5 ~70.0 Kbps).

A SP_d size of 14672 bytes is the focal point for the analysis of the measurement data for the third event. For the “benchmark” experiment the average delay and “goodput” are 661.0 ms and ~178.0 Kbps and for the E5 experiment respectively 1423.0 ms and ~83.0 Kbps. The “goodput” numbers indicate that the “benchmark” was using the third dedicated SUT bearer, while the E5 experiment was still using the second dedicated SUT bearer. This is according to the description of “Event 3” normal behaviour. However, the E5 measurement data for SP_d size 16768 bytes (1617 ms and ~83.0 Kbps) indicates the use of the same SUT bearer as for SP_d size 14672 bytes. We interpret these results as if the Bluetooth intra communication system’s maximum “goodput” rate is limited to ~83.0 Kbps56.

Based on the analysis for the three different events, we conclude that there is a significant difference between USB and Bluetooth for SP_d sizes larger then 524 bytes.

Answer 1.2: The influence of changing the intra communication system from USB to Bluetooth is significant for SUT “downlink” behaviour when using packet sizes larger then 524 bytes.

---

56 On our request Vodafone informed us that the maximum throughput of the Nokia 6650 Bluetooth interface is limited to 155 Kbps.
Question 2: What is the influence of changing the UMTS terminal from Nokia 6650 to PCcard?

Measurement results from experiment 1 (bm) and 8 (E8) are used to answer this question. Figure 6.18 and 6.19 present combined graphs for these experiments. The red “bm” line (Nokia 6650) and the purple “E8” line (PCcard) are relevant for answering this question.

uplink quantitative analysis

The differences between the Nokia 6650 and PCcard UMTS terminals seem to be negligible for all SP_d sizes. Differences in the range of 20 to 40 ms are considered timing accuracy tolerances due to the instrumentation of the measurements (section 5.1.3).
Answer 2.1: The influence of changing the UMTS terminal from USB-Nokia 6650 to PCMCIA-PCcard is negligible for SUT “uplink” behaviour; i.e. the behaviour of the USB-Nokia 6650 to PCMCIA-PCcard is similar.

downlink quantitative analysis

The events shown in Figure 6.19 are used to analyse the SUT’s “downlink”. Up to the first event, the difference between the Nokia 6650 terminal and the PCcard is negligible if we consider differences in the range of 20 to 40 ms as timing accuracy tolerances.

The analysis of the measurement data for the second event is focused on SP_d size 8384 bytes. The “benchmark” experiment shows an average delay of 717.0 ms and for E8 experiment the average is 1147.0 ms. Goodput calculations indicate that the “benchmark” measurement point is related to the second dedicated SUT bearer (“goodput” = ~94.0 Kbps). The E8 measurement point is however related to the first dedicated SUT bearer (“goodput” = ~59.0 Kbps). The description of the second event indicates that SP_d sizes between 2096 and 10480 bytes are independently of the SP_u size served by the SUT’s first or second dedicated bearer. Hence, the difference between the two measurement points can be caused by the SUT not changing the E8 workload to the second dedicated bearer.

A SP_d size of 16768 bytes is the focal point for the analysis of the measurement data for the third event. For the “benchmark” experiment the average delay and “goodput” are 1200.0 ms and ~112.0 Kbps and for the E8 experiment respectively 878.0 ms and ~153.0 Kbps. The “goodput” numbers indicate that the “benchmark” was using the second dedicated SUT bearer, while the E8 experiment was using the third dedicated SUT bearer. This is according to the description of “Event 3” normal behaviour.

Based on the analysis for the three different events, we conclude that there is no difference between Nokia 6650 and PCcard UMTS terminal for SP_d sizes smaller then 2096 bytes. The fact that the Nokia 6650 terminal is faster then the PCcard for SP_d size 8384 bytes, and the fact that the Nokia 6650 terminal is slower then the PCcard for SP_d size 16768 bytes may indicate that the SUT is responsible for the differences. Hence, we assume that the differences between both terminals are not significant. The assumption is based on the SUT “downlink” behaviour for variable SP_d sizes (Figure 6.10).

Answer 2.2: The influence of changing the UMTS terminal from USB-Nokia 6650 to PCMCIA-PCcard is negligible for SUT “downlink” behaviour when using packet sizes smaller then 2096 bytes, and assumed to be negligible for larger packet sizes.

Question 3: What is the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth?
Measurement results from experiment 2 (E2) and 5 (E5) are used to answer this question. Figure 6.18 and 6.19 present combined graphs for these experiments. The blue “E2” line (notebook) and the magenta “E5” line (iPaq) are relevant for answering this question.

**uplink quantitative analysis**

The differences between the notebook and the iPaq seem to be negligible for all SP_d sizes except for 174 and 1572 bytes. We consider differences in the range of 20 to 40 ms as timing accuracy tolerances. Figure 6.24 (SP_u524,SP_d174) shows similar “uplink” behaviour as in Figure 6.20. This behaviour was assumed to be caused by the fact that the SUT encountered resources problems as a consequence of background load. For the SP_u size of 1572 bytes we concluded in the uplink quantitative analysis related to the question “What is the influence of changing the intra communication system from USB to Bluetooth?” that the SP_u524,SP_d1572 observation may be not accurate.

*Answer 3.1: the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth is negligible for SUT “uplink” behaviour; i.e. the behaviour of the notebook and iPaq is similar.*

**downlink quantitative analysis**

The events shown in Figure 6.19 are used to analyse the SUT’s “downlink”. Up to the first event, the difference between the notebook (E5) and the iPaq (E2) is negligible if we consider differences in the range of 20 to 40 ms as timing accuracy tolerances.

The analysis of the measurement data for the second event is focused on SP_d size 8384 bytes. The E2 experiment shows an average delay of 1395.0 ms and for E5 experiment the average is 957.0 ms. Goodput calculations indicate that the E2 measurement point is related to the first
dedicated SUT bearer (“goodput” = ~48.0 Kbps). The E5 measurement point is however related to the second dedicated SUT bearer (“goodput” = ~70.0 Kbps). The description of the second event indicates that SP_d sizes between 2096 and 10480 bytes are independently of the SP_u size served by the SUT’s first or second dedicated bearer. Hence, the difference between the two measurement points can be caused by the SUT not changing the E2 workload to the second dedicated bearer.

For the SP_d sizes larger than 8384 bytes, the differences between the notebook (E5) and the iPaq (E2) become negligible again.

Based on the analysis for the three different events, we conclude that there is no difference between notebook and iPaq for SP_d sizes other than 8384 bytes. The fact that the notebook is faster than the iPaq for SP_d size 8384 bytes may indicate that the SUT is responsible for the difference. Hence, we assume that the differences between both computer systems are not significant. The assumption is based on the SUT “downlink” behaviour for an 8384 bytes SP_d size with a variable SP_u size (Figure 6.10).

**Answer 3.2:** the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth is negligible for SUT “downlink” behaviour for all packet sizes under the assumption that the SUT is responsible for the observed difference for packet sizes of 8384 bytes.

**Question 4:** What is the influence of making the Internet component “present” in the SoD structure?

Measurement results from experiment 1 (bm) and 7 (E7) are used to answer this question. Figure 6.18 and 6.19 present combined graphs for these experiments. The red “bm” line (“Internet not present”) and the green “E7” line (“Internet present”) are relevant for answering this question.

**Note:** Due to the fact that the number of different SP_d sizes used in this experiment is low (section 5.2.1) we consider the results as indicative.

**uplink quantitative analysis**

The differences between “Internet not present” and “Internet present” seem to be negligible for SP_d sizes 174 and 1572 bytes. We consider differences in the range of 20 to 40 ms as timing accuracy tolerances.

**Answer 4.1:** the influence of making the Internet component “present” in the SoD structure is negligible for SUT “uplink” behaviour when using packet sizes between 174 and 1572 bytes.

**downlink quantitative analysis**
The differences between “Internet not present” and “Internet present” are negligible for SP_d sizes 174 and 1572 bytes.

**Answer 4.2:** the influence of making the Internet component “present” in the SoD structure is negligible for SUT “downlink” behaviour when using packet sizes between 174 and 1572 bytes

**Question 5:** What is the influence of changing the SoD transport service and the application protocol buffer size?

Measurement results from experiments 9 (E9), 10 (E10) and 11 (E11) are used. These experiments specified a “user unconfirmed” test for the “uplink” of the SUT with fixed SP_u size of 524 bytes. In the analysis towards the answer to this question, we only focus on a link saturation factor of 2. From an application protocol designer’s perspective, it is important to understand the behaviour of the SUT for “user unconfirmed” application protocols that saturate the SUT’s “uplink”.

**conclusive graph**

Figure 6.25 presents a conclusive graph for this experiment. The red line denotes experiment E11 with a transport service and application buffer size of 32 Kbytes. Experiment E10 is denoted by the green line. The transport service and application buffer sizes are 64 Kbytes and 32 Kbytes. The blue line denotes experiment E9 with a transport service and application buffer size of 64 Kbytes. The vertical axis represents the “uplink” average delay for a SP_u size of 524 bytes. The horizontal line represents a particular packet number in a sample.

**uplink quantitative analysis**

Two important phases can be identified for all experiments: slowstart and saturation phase. We consider the “uplink” delay behaviour during the slowstart phase to be irrelevant for the analysis of the SUT’s behaviour. The slowstart phase functions as an initialisation phase for the actual performance measurements.

The slowstart phase starts at packet number 1 and ends at number 406.

*Note:* In section 5.3 we introduced the formulae to calculate the “SlowstartNoPackets”. For experiments 9, 10 and 11, the SUT “uplink” capacity is 64 Kbps (8192 Bps), the saturation factor is 2 and the SP_u size is 524 bytes. Given is a protocol overhead is 60 bytes.

\[
SP_u_{rate} = \left[ \frac{\text{link}_\text{cap} \times \text{saturation}_\text{factor}}{\text{SP}_u_{size} + \text{protocol}_\text{overhead}} \right]
\]
\[ SP_{\text{u rate}} = \left[ \frac{8192 \times 2}{524 + 60} \right] = 28.0; n = 28 \]

\[ \text{SlowstartNoPackets} = \sum_{i=1}^{n} i = \frac{n*(n+1)}{2} = \frac{28 \times 29}{2} = 406 \]

The “saw tooth” behaviour shown in all the lines is caused by the workload generator. After generating \( SP_{\text{u rate}} \) SP_u’s, it stops for “SLOWSTART_BACKOFFTIME” ms to keep the probability of UMTS terminal output queue overrun as low as possible during the slowstart phase.

The saturation phase starts after packet number 406. From this point onwards the SUT’s “uplink” is offered packets with twice its link packet rate. Figure 6.25 shows that the behaviour of the SoD is influenced significantly by the size of the transport service buffer. This means that the analysis for E9 and E10 are similar.

The average delay for E11 increases linearly up to “K1” (packet no. 427). The transport service buffer is filled with a rate of 14 SP_u’s (+ protocol overhead) per second. The E11 line shows that 13 SP_u’s after the end of the slowstart (427-406) the point “K1” is reached.

Note: The SUT link rate equals the \( SP_{\text{u rate}} / 2 \). The buffer fill rate is \( SP_{\text{u rate}} - \) SUT link rate = 14. The maximum number of SP_u’s (+ protocol overhead) in a 32 Kbyte buffer is 56. The fact that only 13 SP_u’s could be buffered, means that SP_u’s from the slowstart phase were still waiting in the buffer.

After “K1” the average delay remains constant for the rest of the observations. The transport service is implemented by the TCP protocol which uses a flow control service\(^{57}\) offering a speed regulating effect to the TCP service users. The “speed regulation” is translated into a regulated transport delay.

The average delay for E9 and E10 increase linearly up to “K2” (packet no. 515). After “K2” there is a “overshoot” phase which may indicate the presence of additional buffer capacity in the SoD. We focus on the behaviour as depicted by the black line that crosses point “K2”. The transport service buffer is filled with a rate of 14 SP_u’s (+ protocol overhead) per second. The E9 and E10 line show that 109 SP_u’s after the end of the slowstart (515-406) the point “K2” is reached.

Note: The SUT link rate equals the \( SP_{\text{u rate}} / 2 \). The buffer fill rate is \( SP_{\text{u rate}} - \) SUT link rate = 14. The maximum number of SP_u’s (+ protocol overhead) in a 64 Kbyte buffer is 112. The fact that only 109 SP_u’s could be buffered, means that SP_u’s from the slowstart phase were still waiting in the buffer.

\(^{57}\) TCP uses a flow control service to its service users to eliminate the possibility of the sender overflowing the receiver’s buffer. Flow control is a speed matching service, matching the rate at which the sender is sending to the rate at which the receiving service user is reading [Kuro2001].
After “K2” the average delay remains constant for the rest of the observations. The TCP protocol flow control service regulates the speed as in the E11 experiment. Hence, the “speed regulation” is translated into a regulated transport delay.

Figure 6.25  E9, E10, E11 SP_u524 and saturation factor 2.0.

Answer 5: The influence of changing the SoD transport service and the application protocol buffer size is only significant for the transport service. Increasing the buffer size of the transport service results in higher average delays.

6.3 Bottleneck Analysis

In section 6.1 the analytical analysis of the transport system has been conducted which led to the derivation of the high-level queuing model in the system as a tool for a system performance analysis. Moreover, in section 6.2 the delay analysis of the system has been conducted. However, these analyses didn’t lead to the conclusion on the system bottleneck with respect to the goodput performance parameter (defined in chapter 4). Therefore the question on the transport sub-system which is a bottleneck, i.e. a key limiting factor in achieving higher goodput, is raised. The bottleneck analysis is based on the delay analysis and on the derived queuing model of the system.

Definition:

Bottleneck is lessening of throughput. It often refers to networks that are overloaded, which is caused by the inability of the hardware and transmission lines to support the traffic. [WebTech].
The bottleneck analysis activities are divided into four steps:

1. Presentation of “conclusive” graphs (section 6.3.1) for the “benchmark” data
2. Description on how these graphs were derived (section 6.3.2)
3. Quantitative analysis of the conclusive graphs (section 6.3.3)
4. Conclusions on influence of system parameters on bottleneck (section 6.3.4)

6.3.1 Conclusive Graphs

In this section the SUT goodput and efficiency analysis’ results for the benchmark case for the uplink and downlink SUT direction are presented. The goodput and efficiency performance parameters are derived from the delay performance parameter analyzed in section 6.2.

Figure 6.26 presents the influence of the “user confirmed” SE’s size on the goodput behaviour of the SUT’s “uplink”. On the horizontal axis the values of the SP_d variable component are presented. The vertical axis presents the statistical average of the derived performance parameter goodput (section 4.5.2) for a SP_u fixed size in relation to a SP_d variable size. Two red horizontal lines indicate the system capacity boundaries for the common (16 Kbps) and dedicated bearer (64 Kbps) in the upstream direction.

In Figure 6.27 the influence of the “user confirmed” SE’s size on the goodput behaviour of the SUT’s “downlink” is presented. On the horizontal axis the values of the SP_d variable component are presented. The vertical axis presents the statistical average of the derived performance parameter goodput for a SP_d variable size in relation to a SP_u fixed size. Three red horizontal lines...
indicate the system capacity boundaries for the common (16 Kbps) and dedicated bearer (64 Kbps, 128 Kbps) in the downstream direction.

As it has been explained in section 6.2, the SUT “uplink” characteristics are of importance from the MobiHealth system perspective. Furthermore, Figure 6.26 and 6.27 showed that the goodput behaviour of the SUT “downlink” is less predictable then the behaviour of the “uplink” due to the changes in bearers. Hence, we focus on the efficiency (derived performance parameter from goodput) analysis on the SUT “uplink” only, as it was for the delay and jitter analysis (Figure 6.11). Figure 6.28 presents the SUT “uplink” goodput and efficiency behaviour for a variable size SP_u and a fixed size (524 bytes) SP_d. The efficiency is indicated on the right vertical axis as percentage of the nominal (maximum) uplink capacity (which is 64 Kbps) for a dedicated bearer.
6.3.2 Derivation of Conclusive Graphs

The statistical data used in the delay analysis, was used to obtain a derived performance parameter from the SUT: “goodput” and then efficiency. The “goodput” is defined as:

\[
\text{SUT}_{\text{goodput}} = \text{net SUT throughput measured at application layer (in Kbps)}
\]

\[
\text{SUT}_u_{\text{goodput}} = \frac{\text{SP}_u_{\text{length}}}{\text{SP}_u_{\text{delay}}}
\]

\[
\text{SUT}_d_{\text{goodput}} = \frac{\text{SP}_d_{\text{length}}}{\text{SP}_d_{\text{delay}}}
\]

units of measure: [Kbps] = [b] / [ms]

Efficiency is the ratio of maximum achievable goodput (i.e. usable capacity) to nominal capacity and it is defined as:

\[
\text{SUT}_u_{\text{efficiency}} = \left( \frac{\text{SUT}_u_{\text{goodput}}}{\text{SUT}_u_{\text{capacity}}} \right) \times 100
\]

\[
\text{SUT}_d_{\text{efficiency}} = \left( \frac{\text{SUT}_d_{\text{goodput}}}{\text{SUT}_d_{\text{capacity}}} \right) \times 100
\]

units of measure: [%] = [Kbps] / [Kbps]
SUT\textsubscript{capacity} is assumed to be different for uplink: 16Kbps or 64Kbps (\text{SUT\_u\textsubscript{capacity}}) and downlink: 16Kbps, 64 Kbps, 128 Kbps and 384 Kbps (\text{SUT\_d\textsubscript{capacity}})

From the upstream delay analysis Figure 6.9 has been generated, from which the upstream goodput characteristic presented at Figure 6.26 follows. Therefore Figure 6.26 presents the relation between SP\_u “goodput” and a variable SP\_d size for fixed SP\_u sizes. Two red lines placed at figure indicate the nominal uplink capacity of the SUT for the common (16 Kbps) and dedicated bearer (64 Kbps).

Analogously to the upstream analysis, from the downstream delay analysis Figure 6.10 has been generated, from which the downstream goodput characteristic presented at Figure 6.27 follows. Therefore Figure 6.27 presents the relation between SP\_d “goodput” and a variable SP\_d size for fixed SP\_u sizes. Three red lines placed at figure indicate the SUT’s nominal uplink capacity boundaries for the common and dedicated bearers.

Figure 6.28 is generated based on the SUT “uplink” delay analysis for the variable size SP\_u and a fixed (524B) SP\_d. The motivation for the choice of this particular case (uplink direction plus one SP\_d size) has been presented in section 6.2.1, where the corresponding Figure 6.11 is presented. Therefore the Figure 6.28 presents the SUT “uplink” goodput and efficiency behaviour for a variable size SP\_u and a fixed size (524 bytes) SP\_d. The goodput is indicated on the left vertical axis, while the efficiency is indicated on the right vertical axis. Efficiency has been calculated with respect to the uplink SUT\textsubscript{goodput} for the dedicated bearer, which is 64Kbps; from the delay analysis it’s known that while executing SEs with SP\_d 524B the dedicated bearer has been used.

### 6.3.3 Quantitative Analysis

The quantitative analysis of the SUT is divided into two sections: uplink and downlink.

**uplink “goodput” analysis**

The quantitative delay analysis is based on Figures: 6.26, 6.28 and on the knowledge obtained from the system model and lead to the following conclusions:

1) SP\_d size \textit{does not} influence the SUT “uplink” goodput behaviour for a fixed SP\_u sizes, except for SP\_d size 174: for the SP\_u\textsubscript{174} and SP\_d\textsubscript{174} the common bearer is used, while for the SP\_d higher than 174 the SUT changes from the common bearer to the first bearer, such the goodput seen by the SP\_u\textsubscript{174} is higher (see the discussion in section 6.2.3).

2) For SP\_u sizes other than 174 bytes the SUT’s dedicated bearer 1 is always used; i.e. all data lines are above the lower red line.
3) The uplink “goodput” depends on the SP_u size. There is not a linear relation between the “goodput” and the SP_u size as proofed by Figure 6.28. The relation is exponential; such there is a limit on the maximum achievable goodput.

4) The maximum uplink “goodput” is obtained with SP_u sizes larger than 6812 bytes and equals 54 Kbps. The same maximum uplink “goodput” number has been also derived while modelling the system (section 6.1). For SP_u size 524B arrived with rate \( \geq 14 \) [pack/sec] to the system, the departure rate was only 13, which indicated that the system is maximally saturated and cannot handle the input. Summarizing the discussion, the system efficiency equals at most 84%. This efficiency is seen from the application level, which could mean that there is 16% of the communication overhead.

**downlink “goodput” analysis**

The quantitative analysis based on Figure 6.27 lead to the following conclusions:

1) SP_d size does influence the SUT “downlink” goodput behaviour for a fixed SP_u sizes.

2) SUT “downlink” goodput behaviour is not linearly increasing with the SP_u size.

3) As it has been observed in case of delay behaviour, the SUT “downlink” goodput behaviour has also remarkable events for three SP_d sizes: 2096 (E1), 10480 (E2) and 23056 bytes (E3). These events are of importance, especially from the efficiency point of view.

   **E1:** SP_d sizes between 174 and 2096 bytes are independently of SP_u size served by the SUT’s first dedicated bearer.
   
   If we assume that an event for which SP_d has a size 4192B is an event for which we get the maximum achievable goodput (which is the 54 Kbps), then the maximum efficiency of the system for this bearer is 84%, which conform the number obtained in the uplink goodput analysis. Communication overhead is 16%.

   **E2:** SP_d sizes between 2096 and 10480 bytes are independently of SP_u size served by the SUT’s first or second dedicated bearer.
   
   If we assume that an event for which SP_d has a size 10480B is an event for which we get the maximum achievable goodput in the second bearer (which is the 106 Kbps), then the maximum efficiency of the system for this bearer is 83%, which conform the number obtained in the uplink goodput analysis. Communication overhead is again 17%.

   **E3:** SP_d sizes between 10480 and 23056 bytes are independently of SP_u size served by the SUT’s second or third dedicated bearer, and SP_d sizes larger
then 23056 bytes are independently of SP_u size served by the SUT’s third
dedicated bearer.
The discussion regarding the system efficiency gives the following conclusions.
Particularly, if we assume that an event for which SP_d has a size 35632B is an
event for which we get the maximum achievable goodput (which is the 289
Kbps), and then the maximum efficiency of the system for this bearer is only
75%. There are two possible conclusions that lead from the discussion on this
number. First is that the communication overhead is higher than in previous
cases and equals 25%. The other conclusion is that for these big packets
(35632B, 48208B) the network has not been saturated, such the maximum
achievable goodput seen is still lower, than it could be. This could be a subject
of the future research.

As it has been said in delay analysis, the bearer changing moment for the downlink
seems to be related to the volume of data offered (size x rate) but goodput analysis
did not unequivocally indicated the lower and upper boundaries of the volume.

### 6.3.4 Influence of System Parameters

The influence of the SUT structure and SoD structure system parameters are analysed for the
derived performance parameter goodput in the same manner as for delay performance parameter
in section 6.2.4. As in previous analysis the focus is on SEs: SP_u524, SP_d_variable.

We pose the following five questions:

1) What is the influence of changing the intra communication system from USB to
Bluetooth?

2) What is the influence of changing the UMTS terminal from USB-Nokia 6650 to
PCMCIA-PCcard?

   *Note: We abstract from the USB and PCMCIA intra communication systems
   behaviour and consider these systems as a part of the UMTS terminals.*

3) What is the influence of changing the computer system from notebook-Bluetooth to
iPaq-Bluetooth?

4) What is the influence of making the Internet component “present” in the SoD
structure?

5) What is the influence of changing the SoD transport service and the application
protocol buffer size?

The answers to the five questions will be based, as in case of delay analysis, on the analysis of
relevant measurements data. Conclusive graphs are generated based on relevant statistical data
from the measurements, particularly based on the Figures 6.18 and 6.19. Furthermore, “raw data” graphs from Volume 2 are used to explain the SUT’s “uplink” and “downlink” behaviour.

**Question 1:** What is the influence of changing the intra communication system\(^{58}\) from USB to Bluetooth?

Similarly to the delay analysis, measurement results from experiment 1 (bm) and 5 (E5) are used to answer this question. Figure 6.29 and 6.30 present combined graphs for four experiments. The red “bm” line and the magenta “E5” line are relevant for answering this question.

*Note: The lines in Figure 6.29 and Figure 6.30 must be interpreted as “trend” lines passing through the measurement points.*

**conclusive graphs**

The thin red line in Figure 6.29 and Figure 6.30 show the SUT goodput behaviour for a USB intra communication system, and the magenta lines show the goodput behaviour for the Bluetooth alternative. The vertical axis represents the “uplink” average goodput for a SP_u size 524 bytes. The horizontal line represents the different sizes of the SP_d.

---

\(^{58}\) The connection between the computer system and the Nokia 6650 UMTS terminal is based on USB.
uplink quantitative analysis

As in case of experienced delay, the difference between goodput achieved in case of USB and Bluetooth seems to be significant for SP_d sizes of 174, 524 and 1572 bytes. As it has been
proven in the delay analysis, this difference can be caused by the inaccuracy of measurements. Moreover, there is no significant difference between USB and Bluetooth for other sizes.

**Answer 1.1:** The influence of changing the intra communication system from USB to Bluetooth is negligible for SUT “uplink” goodput behaviour; i.e. the goodput behaviour of the USB and Bluetooth intra communication system is similar.

downlink quantitative analysis

The goodput quantitative analysis based on Figure 6.27 leaded to a conclusion that the SUT “downlink” goodput behaviour has remarkable events for three SP_d sizes: 2096 (E1), 10480 (E2) and 23056 bytes (E3). Up to the first event, the difference between USB and Bluetooth seems to increase with SP_d sizes. At the first event both measurement points are related to the first dedicated SUT bearer. As it has been indicated in delay analysis, the second event’s measurement data analysis gives obtained goodput number as related to the second dedicated SUT bearer (for both experiments). Regarding the third event, as it could be seen from the Figure and as the analysis proves, the Bluetooth never gets more goodput that 83Kbps. That proves that Bluetooth is a goodput bottleneck.

Based on the analysis for the three different events, we conclude that there is a significant difference between goodput obtained for USB and Bluetooth for SP_d sizes larger then 524 bytes.

**Answer 1.2:** The influence of changing the intra communication system from USB to Bluetooth is significant for SUT “downlink” goodput behaviour when using packet sizes larger then 524 bytes. The Bluetooth is a bottleneck; the maximum obtained goodput is 83 Kbps.

**Question 2:** What is the influence on goodput of changing the UMTS terminal from Nokia 6650 to PCcard?

Measurement results from experiment 1 (bm) and 8 (E8) are used to answer this question. Figure 6.29 and 6.30 present combined graphs for these experiments. The thin red “bm” line (Nokia 6650) and the purple “E8” line (PCcard) are relevant for answering this question.

uplink quantitative analysis

As it could be seen from the figure, there is no significant difference in obtained goodput for both technologies.

**Answer 2.1:** The influence of changing the UMTS terminal from USB-Nokia 6650 to PCMCIA-PCcard is negligible for SUT “uplink” goodput behaviour; i.e. the goodput behaviour of the USB-Nokia 6650 to PCMCIA-PCcard is similar.
downlink quantitative analysis

As in case of the previous downlink analysis, remarkable events for three SP_d sizes are considered. Up to the first event (SP_d 2096B) the USB and PCcard give similar goodput behaviour. For the second event, the SP_d 8384B has been taken into account, as measurement results are available for this point. It occurs that “benchmark” measurement point is related then to the second dedicated SUT bearer ("goodput" = ~94.0 Kbps), while for E8 the first bearer was in use (“goodput” = ~59.0 Kbps), while for the third event, the measurement data indicate that SUT has been using the second dedicated SUT bearer and the PCcard has been using the third dedicated SUT bearer. As it has been said in case of SUT delay behaviour analysis, the SUT is responsible for a difference in goodput behaviour experienced in case of USB and PCcard, therefore it is assumed that the differences between both terminals are not significant.

*Answer 2.2: The influence of changing the UMTS terminal from USB-Nokia 6650 to PCMCIA-PCcard is negligible for SUT “downlink” goodput behaviour when using packet sizes smaller than 2096 bytes, and assumed to be negligible for larger packet sizes.*

*Question 3: What is the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth?*

Measurement results from experiment 2 (E2) and 5 (E5) are used to answer this question. Figure 6.29 and 6.30 present combined graphs for these experiments. The blue “E2” line (notebook) and the magenta “E5” line (iPaq) are relevant for answering this question.

uplink quantitative analysis

The differences between the notebook and the iPaq seem to be negligible for all SP_d sizes except for 174 and 1572 bytes. For these sizes the analysis in Question1 has been conducted that leaded to the conclusion that the difference between notebook-Bluetooth (E5) and notebbok-USB (“bm”) is assumed to be negligible, therefore the difference between the notebook-Bluetooth and iPaq-Bluetooth is assumed to be negligible.

*Answer 3.1: the influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth is negligible for SUT “uplink” goodput behaviour; i.e. the behaviour of the notebook and iPaq is similar.*

downlink quantitative analysis

Based on the Figure 6.30, it’s concluded that there is no significant difference in the goodput obtained in both experiments. Moreover, it’s concluded that the Bottleneck maximum goodput is ~83Kbps.
Answer 3.2: The influence of changing the computer system from notebook-Bluetooth to iPaq-Bluetooth is negligible for SUT “downlink” goodput behaviour for all packet sizes. In both the Bluetooth is a bottleneck; the maximum obtained goodput is 83 Kbps.

Question 4: What is the influence of making the Internet component “present” in the SoD structure?

Measurement results from experiment 1 (bm) and 7 (E7) are used to answer this question. Figure 6.29 and 6.30 present combined graphs for these experiments. The red “bm” line (“Internet not present”) and the green “E7” line (“Internet present”) are relevant for answering this question.

Note: Due to the fact that the number of different SP_d sizes used in this experiment is low (section 5.2.1) we consider the results as indicative.

uplink quantitative analysis

The differences between “Internet not present” and “Internet present” seem to be negligible for SP_d sizes 174 and 1572 bytes.

Answer 4.1: the influence of making the Internet component “present” in the SoD structure is negligible for SUT “uplink” goodput behaviour when using packet sizes between 174 and 1572 bytes

downlink quantitative analysis

The differences between “Internet not present” and “Internet present” are negligible for SP_d sizes 174 and 1572 bytes.

Answer 4.2: the influence of making the Internet component “present” in the SoD structure is negligible for SUT “downlink” goodput behaviour when using packet sizes between 174 and 1572 bytes

Question 5: What is the influence of changing the SoD transport service and the application protocol buffer size?

Measurement results from experiments 9 (E9), 10 (E10) and 11 (E11) are used. These experiments specified a “user unconfirmed” test for the “uplink” of the SUT with fixed SP_u size of 524 bytes. In the analysis towards the answer to this question, we only focus on a link saturation factor of 2. From an application protocol designer’s perspective, it is important to understand the behaviour of the SUT for “user unconfirmed” application protocols that saturate the SUT’s “uplink”.

conclusive graph
Figure 6.31 presents a conclusive graph for this experiment. The red line denotes experiment E11 with a transport service and application buffer size of 32 Kbytes. Experiment E10 is denoted by the green line. The transport service and application buffer sizes are 64 Kbytes and 32 Kbytes. The blue line denotes experiment E9 with a transport service and application buffer size of 64 Kbytes. The vertical axis represents the “uplink” average delay for a SP\_u size of 524 bytes. The horizontal line represents a particular packet number in a sample. The goodput behaviour is calculated based on the experienced delay behaviour.

![Figure 6.31 E9, E10, E11 SP\_u 524 and saturation factor 2.0.](image)

In section 6.1 we described the relation between the system response time, service time and waiting time:

$$ \bar{r} = \bar{s} + \bar{w} $$

We assume the service time as constant in this analysis, therefore the formula above is rewritten into:

$$ \bar{r} = \bar{c} + \bar{w} $$

The system’s response time is a function of waiting time. As we proofed in section 6.2.4 Question 5, the SoD “uplink” delay behaviour is influenced significantly by the size of the transport service buffer. Therefore, the system’s waiting time (i.e. average delay) is associated with the buffer size; large buffer sizes result in long waiting times (compare Figure 6.31 lines E10 and E11). Hence, the system’s response time is a function of the transport service buffer size:

$$ \bar{r} = \bar{c} + f(Buff_{size}) $$
The “goodput” of the SoD for E10 equals 13 Kbps and for E11: 20 Kbps and is calculated by:

\[ \text{SoD}_{\text{goodput}} = \frac{SP_u \_ \text{length}}{SP_u \_ \text{delay}} \]

The average system response time for a particular SP_u size equals the SP_u transmission delay. This leads to the formula:

\[ \text{SoD}_{\text{goodput}} = \frac{SP_u \_ \text{length}}{r} = \frac{SP_u \_ \text{length}}{C + f(Buff\_\text{size})} \]

The formula above proofs that the SoD “goodput” is a function of the transport service buffer size. The efficiency of the SoD as a transport system is derived by the division of “goodput” by “throughput”. Hence, the efficiency is a function of the transport service buffer size. The formula below supports this analysis:

\[ \text{Efficiency}_{\text{SoD}} = \frac{\text{SoD}_{\text{goodput}}}{\text{SoD}_{\text{throughput}}} = \frac{SP_u \_ \text{length}}{\text{SoD}_{\text{throughput}} \ast (C + f(Buff\_\text{size}))} \]

Figure 6.32 shows the SoD efficiency (i.e. influence) for the two different transport service buffer sizes.

<table>
<thead>
<tr>
<th>SoD efficiency</th>
<th>buffer size</th>
</tr>
</thead>
<tbody>
<tr>
<td>32%</td>
<td>32 Kbyte</td>
</tr>
<tr>
<td>21%</td>
<td>64 Kbyte</td>
</tr>
</tbody>
</table>

Figure 6.32 SoD efficiency versus transport service buffer size.

We assume that the SoD efficiency is correlated to the SUT’s efficiency for the behaviour after the “K1” and “K2” points; i.e. the average SoD delay remains constant. The influence of the non-SUT part (i.e. transport service) becomes less significant as the buffers reach their maximum capacity. The flow control service of TCP will control the rate at which its service user generates new SP_u’s. The SUT is influencing this process by successfully transporting SP_u’s from a sender to a receiver, and as a consequence releasing buffer capacity in the non-SUT part. Hence, the behaviour the SoD’s “uplink” border is directly associated with the SUT “uplink” behaviour.

Answer 5: The influence of changing the SoD transport service and the application protocol buffer size is only significant for the transport service. Increasing the buffer size of the transport service results in lower average goodput.

6.4 Scalability Analysis

The performance evaluation of the MobiHealth transport service has been done so far for the case when there is only one service user. However of interest is to know what the behaviour of the
system serving multiple users is. Therefore the system scalability analysis has been conducted with respect to the delay and goodput performance measures. For this purpose the results of scalability experiments 2, 3 and 4 were used, where the measurements for 1, 5 and 10 instances of services has been conducted. The systems’ parameters’ setup has been the same for all experiments. The raw data set belonging to experiments 2, 3 and 4 are placed in Volume2.

The SUT scalability analysis activities are divided into three steps:

1) Presentation of “conclusive” delay and goodput graphs (section 6.4.1).
2) Description on how these graphs are derived (section 6.4.2).
3) Quantitative analysis of the conclusive graphs (section 6.4.3).

### 6.4.1 Conclusive Graphs

Conclusive graphs: Figures 6.33, 6.34, 6.35 and 6.36 present the influence of the “user confirmed” SE’s size on the SUT’s statistical average of the primary performance parameter delay (6.40 and 6.41) and it’s accuracy, and statistical average of the derived performance parameter goodput (6.42 and 6.43) behaviours for one instance of the server which has been executing the service standalone (1) or competing with other instances of the service (5 and 10 instances of the service). These conclusive graphs have been drawn for SE with SP_u variable size and SP_d174. This choice of the packet size has been made with respect to the BANip protocol.

Figure 6.33 presents the influence of the “user confirmed” SE’s size on the SUT uplink delay behaviour for the service running in scalability experiments. On the horizontal axis the values of the SP_u variable component are presented. The left vertical axis presents the average uplink delay for a SP_d174 size in relation to a SP_u variable size. The right vertical axis presents the statistical accuracy of the obtained delay. Three lines represent the data obtained in scalability experiments.
Figure 6.33  SUT “uplink” delay behaviour and its accuracy for SP_u_variable and SP_d174 when 1, 5, 10 service instances.

Figure 6.34 presents the influence of the “user confirmed” SE’s size on the downlink delay behaviour of the SUT for the service running in scalability experiments. On the horizontal axis the values of the SP_u_variable component are presented. The left vertical axis presents the average downlink delay for a SP_d174 size in relation to a SP_u_variable size. The right vertical axis presents the statistical accuracy of the obtained delay.
Figure 6.34  SUT “downlink” delay behaviour and its accuracy for SP_d174 and SP_u variable when 1, 5, 10 service instances.

Figure 6.35 presents the influence of the “user confirmed” SE’s size on the uplink goodput behaviour of the SUT for the service running in scalability experiments. On the horizontal axis the values of the SP_u variable component are presented. The vertical axis presents the average of the goodput for a SP_d174 size in relation to a SP_u variable size. The red lines present the capacity boundaries of the bearers.
Figure 6.35  SUT “uplink” goodput behaviour for SP_d174 and SP_u_variable when 1, 5, 10 service instances.

Figure 6.36 presents the influence of the “user confirmed” SE’s size on the downlink goodput behaviour of the SUT for the service running in scalability experiments. On the horizontal axis the values of the SP_u_variable component are presented. The left vertical axis presents the average of the goodput for a SP_d174 size in relation to a SP_u_variable size. The red lines present the capacity boundaries of the bearers.
6.4.2 Derivation of Conclusive Graphs

The conclusive graphs have been drawn for one instance of the server which has been using the service alone, then when it has been competing with the other: 5 and then 10 other service users.

delay SUT behaviour

The delay behaviour conclusive graphs have been derived in the same way, as is has been presented in section 6.2.2. Firstly, the raw data from experiments 2, 3 and 4 has been collected individually for each service instance (see Volume 2). Then for each service instance the average delay and its accuracy have been calculated for each individual SE (in StatDataFile). The conclusive graphs have been drawn for one instance of the server which has been participating in these experiments and for SEs with size SP_d 174B. As it has been pointed in section 6.2.2 even if some delay data had been obtained with a low accuracy (in relation to the 95% conf interval), it has been used for further analysis.

goodput SUT behaviour

The goodput conclusive graphs have been derived in the same way, as is has been presented in section 6.3.2. The goodput analysis has been done based on the same data set, as used for delay analysis.
6.4.3 Quantitative Analysis

For the simplicity of analysis description, the following abbreviations are introduced:

- \( c_1, c_5, c_{10} \) cases where the system is used by a single user (\( c_1 \)), by five concurrent users (\( c_5 \)) and 10 concurrent users (\( c_{10} \))

- \( d_1, g_1 \) meaning the average delay (\( d_1 \)) and goodput (\( g_1 \)) behaviour experienced by a single user in case \( c_1 \)

- \( d_5, g_5 \) meaning the average delay (\( d_5 \)) and goodput (\( g_5 \)) behaviour experienced by the service user in case \( c_5 \)

- \( d_{10}, g_{10} \) meaning the average delay (\( d_{10} \)) and goodput (\( g_{10} \)) behaviour experienced by the service user in case \( c_{10} \)

**uplink delay analysis**

The quantitative uplink delay analysis is based on Figure 6.33 and lead to the following conclusions:

1) the SUT uplink average delay behaviour experienced by the service user while concurring with other users is less predictable while the number of other users increase; the variance of the delay increase.

2) the difference between \( d_1 \) and \( d_5, d_{10} \) is significant for the SE: SP_u 174B and SP_d 174B, due to the fact that for \( c_1 \), there was no switch to the bearer 1 observed, as it has been in case of \( c_5 \) and \( c_{10} \) (see Figure 6.37). Due to this fact the accuracy of obtained SP_u174 delay for \( c_5 \) and \( c_{10} \) is very low.
3) For packets SP_u bigger than 174B, there is a significant difference between d1 and d5 (or d10). D1 is always lower than d5 or d10. The difference between d5 and d10 is constant (around 100ms).

4) The difference between d1 and d5 (or d10) is increasing for the packet SP_u increasing (and bigger than 174B). For SP_u 1048B the difference between d1 and d5 (or d10) is 100ms (200ms), while for 7860B is 700ms (for both: d5 and d10).

5) The “jitter” of the obtained delay is comparable for c1 and c5 (for SP_u bigger than 174B), while for c10 seems to be unpredictable especially for a big packets. The high “jitter” has to do with the uplink bearer switching behaviour of the system; between the common bearer and bearer 1. This behaviour is unpredictable (see Figure 6.38) and occurs with the higher probability for bigger packet sizes and for increasing number of concurrent users (c5 and c10).
The quantitative downlink delay analysis is based on Figure 6.34 and lead to the following conclusions:

1) When taking into consideration the accuracy of measurements: +/- 40 ms, it’s fair to say that the SUT downlink average delay behaviour for SP_d 174B experienced by the service user while concurring with other users is the same regardless the number of other users.

2) The “jitter” of the delay is unpredictable for all cases for SP_u and even less predictable for bigger SP_u sizes and higher number of concurrent users. The downlink “jitter” has to do with the highly probable downlink bearer switching behaviour of the system; between the common bearer and bearer 1 (see uplink analysis).

uplink goodput analysis

Figure 6.38  E2, E3 and E4 uplink bearer switching.
The quantitative uplink goodput analysis is based on Figure 6.35 and lead to the following conclusions:

1) the SUT uplink average goodput behaviour experienced by the service user while concurring with other users is lower while the number of other users increase.

2) the difference between g1 and g5, g10 is significant for the SE: SP_u 174B and SP_d 174B, due to the fact that for c1, there was no switch from the common bearer to the bearer 1 observed, as it has been in case c5 and c10. The efficiency of the system usage for this case is higher for c5 and c10 than for c1 and it equals around 10%. This low efficiency has to do with the small packet sizes, that don’t utilize the system.

3) For packets SP_u bigger than 174B, there is a significant difference between g1 and g5 (or g10). The difference between g5 and g10 is small. The efficiency of the system usage for c1 is always higher than for c5 or c10, but it is never higher than 50%.

4) The difference between g1 and g5 (or g10) is increasing for the packet SP_u bigger than 174B and up to 1572B. For SP_u 1572B the difference between d1 and d5 (or d10) is the biggest and equals 10Kbps. As it could be seen on Figure 6.38 for this packet size in c1 system has assigned bearer 1, while for c5 and c10 the system has assigned common bearer or bearer 1 with the high probability of switching between these bearers. The difference in efficiency obtained in between c1 and c5 or c10, is the highest for the SP_u 1572B and equals 20%.

4) The difference between g1 and g5 (or g10) is decreasing for the packet SP_u bigger than 1572B and increasing. For big packet sizes in c1, c5 and c10 system has assigned common bearer or bearer 1 with the higher probability of switching between these bearers, which result in low experienced goodput for all cases. Therefore the efficiency of the system is lower for bigger packet sizes and equals ~40%.

downlink goodput analysis

The quantitative downlink goodput analysis is based on Figure 6.36 and lead to the following conclusions:

1) When taking into consideration the accuracy of measurements, it’s fair to say that the SUT downlink average goodput behaviour for SP_d 174B experienced by the service user while concurring with other users is the same regardless the number of other users.

2) It could be concluded that for all cases: c1, c5 and c10 there was a common bearer assigned in the system. This is not completely true, as it has been discussed in the uplink goodput section; system switches between the common bearer and bearer 1.
3) The goodput experienced by the user is lower than 16Kbps regardless the assigned bearer in the system. This is because SP_d is small.

4) The efficiency of the system usage is low; between 10 and 20 % for all cases. This has to do with the packet size which does not much utilize the system.
7 Digest of results

This chapter presents the results of the performance evaluation activity of the V3GNL network (Vodafone’s (pre) commercial UMTS network in the Netherlands) as a MobiHealth transport system supporting the BANip application protocol. Due to the inverted data “producer – consumer” paradigm, the main point of the evaluation will the “uplink” behaviour of the network. We do however provide conclusive information about the “downlink” behaviour. We decided to base the performance evaluation of the V3GNL network on measurements to have real-world data available for the assessment of V3GNL network. An analytical (queuing) model was developed and parameterised by means of the measurement data to support the assessment activity.

The measurement activity is considered a complex and time consuming task, therefore a methodological approach is needed to successfully execute this activity. The methodology selected is based on [Jain1991] [Hoek1997] and used in our measurements activity. The methodology’s stepwise approach made it possible to execute the measurements in an organized and efficient manner; the approach worked for us! We developed an evaluation (i.e measurements) system that was able to generate a priori defined sequences of packets, offer them to the V3GNL network and measure the delay for the “uplink” and “downlink” communication links of the network.

The design and execution of the experiments followed by a first (quick) evaluation of the measurement data indicated significant problems with the time synchronisation of the different components of the evaluation system. We concluded that inband time synchronisation of the components over UMTS wireless links with changing characteristics (e.g. roundtrip delays) is not usable. Outband time synchronisation delivered better time synchronisation of the evaluation system components, but due to the instrumentation of the measurements only a minimum tolerance of 40ms was possible.

The performance criterion selected was delay, providing a timely behaviour of the V3GNL network while transporting packets of different sizes and rates. Once the delay characteristics of the network are known, other performance parameters can be derived: jitter and “goodput”. The conclusions the delay, jitter and “goodput” of the V3GNL network are presented in relation to the “uplink” and “downlink”. We also investigated the influence of certain parameters to the behaviour of the V3GNL network.

V3GNL uplink

1) For a “user confirmed” application protocol, the size of a packet transported over the V3GNL “downlink” does not influence the V3GNL “uplink” delay behaviour related to the transportation of a specified packet size; except for the “uplink” packet size of 174 bytes.
2) The V3GNL “uplink” delay behaviour is increasing linearly with the packet size offered to this link.

3) The size and rate of a packet offered to the “uplink” and “downlink” cause changing behaviour of the links. This changing behaviour is considered to be normal if it converges to a dedicated bearer and this bearer is going to be used for the remaining part of a data transfer. In 99.8% of the executed performance measurements this behaviour is validated. Only 0.2% showed an altering behaviour of the downlink.

### jitter

The V3GNL “uplink” jitter is significant for all packet sizes offered to this link because of bearer changes and “rare” events that seem to have no frequency in their behaviour.

### “goodput”

1) For “uplink” packet sizes other than 174 bytes the V3GNL’s dedicated bearer 1 is always used.

2) The uplink “goodput” depends on the packet size. There is no linear relation between the “goodput” and the packet size. The relation is exponential; therefore there is a limit on the maximum achievable “goodput”.

3) The maximum “uplink” “goodput” of 54 Kbps is obtained for packet sizes larger than 6812 bytes. The V3GNL analytical model showed that for packet sizes of 524 bytes arriving at a rate \( \geq 14 \) packets per second to the V3GNL network, the departure rate was only 13. This indicated that the network is maximally saturated and cannot handle the input. The maximum system efficiency is calculated at 84%. This efficiency is seen from the application level, which implies a 16% communication overhead. We assume the same efficiency for other packet sizes offered to the “uplink”.

### V3GNL downlink

### delay

1) For a “user confirmed” application protocol, the size of a packet transported over the V3GNL “downlink” does influence the V3GNL “downlink” delay behaviour independently of a specified packet size transported over the “uplink”.

2) The V3GNL “downlink” delay behaviour is not linearly increasing with the packet size, except for packet sizes larger than 23056 bytes.
3) The V3GNL “downlink” delay behaviour has remarkable events for three packet sizes: 2096 (E1), 10480 (E2) and 23056 bytes (E3).

E1: packet sizes between 174 and 2096 bytes are packet size independently served by the V3GNL first dedicated bearer.
E2: packet sizes between 2096 and 10480 bytes are packet size independently served by the V3GNL’s first or second dedicated bearer.
E3: packet sizes between 10480 and 23056 bytes are packet size independently served by the V3GNL’s second or third dedicated bearer, and packet sizes larger then 23056 bytes are independently of packet size served by the V3GNL’s third dedicated bearer.

The bearer changing moment for the downlink seems to be related to the volume of data offered (size x rate). Analysis of the raw data did not clearly indicate the lower and upper boundaries of the volume.

\textit{jitter}

The V3GNL “downlink” jitter is significant for all packet sizes offered to this link because of bearer changes and “rare” events that seem to have no frequency in their behaviour.

\textit{“goodput”}

1) The V3GNL “downlink” “goodput” behaviour is \textit{not} linearly increasing with the packet sizes used on the “uplink”.

2) For the V3GNL “downlink” “goodput” behaviour we use the same remarkable events as specified in the previous \textit{delay} section.

E1: packet sizes between 174 and 2096 bytes are packet size independently served by the V3GNL first dedicated bearer. For this event the V3GNL’s maximum achievable “goodput” is the 45 Kbps, which leads to an efficiency of 70%.
E2: packet sizes between 2096 and 10480 bytes are packet size independently served by the V3GNL’s first or second dedicated bearer. For the first bearer the maximum “goodput” and efficiency is obtained for a packet size of 4192 bytes: respectively 54 Kbps and 84%. Communication overhead is 16%. For a packet size of 10480 bytes, the maximum “goodput” and efficiency is obtained for the second bearer: respectively 106 Kbps and 83%. Communication overhead is again 17%.
E3: packet sizes between 10480 and 23056 bytes are packet size independently served by the V3GNL’s second or third dedicated bearer, and packet sizes larger then 23056 bytes are independently of packet size served by the V3GNL’s third dedicated bearer. If we assume that for a packet size of 35632 bytes the network provides the maximum achievable “goodput” of 289 Kbps, then the maximum efficiency of the third bearer is 75%. There are two possible considerations: 1) the communication overhead is higher than in previous cases
(25%), and 2) the V3GNL network is not saturated for packets with sizes of 35632 and 48208 bytes, therefore the maximum achievable “goodput” is lower. This could be a subject of the future research.

As similar to the delay analysis, the bearer changing moment for the “downlink” seems to be related to the volume of data offered (size x rate) but the “goodput” analysis did not unequivocally indicated the lower and upper boundaries of the volume.

V3GNL system parameters of influence

We defined “system parameters of influence” that in principal could influence the behaviour of the V3GNL network. The decision was made to consider the UMTS terminal and the communication system used between the terminal and a computer system as part of the V3GNL network. The communication system was denote as “intra communication system” and is comparable to the MobiHealth intra BAN communication system. The conclusions are therefore (partially) usable for the MobiHealth intra BAN communication system performance evaluation.

intra communication system

Changing the intra communication system from USB to Bluetooth (same UMTS terminal) had a negligible influence for the V3GNL “uplink” behaviour; i.e. the behaviour of both intra communication systems is similar. The influence of the change is significant for the V3GNL “downlink” behaviour when using packet sizes larger then 524 bytes. The Bluetooth instantiation of the intra communication system was identified as a bottleneck. The maximum obtained “goodput” for “downlink” communication was 83 Kbps.

changing UMTS terminals

The influence of changing the UMTS terminal from Nokia 6650 to Vodafone’s Mobile Connect PCcard is negligible for the V3GNL “uplink” behaviour; i.e. the behaviour of both terminals is similar. The influence of changing terminals is negligible the V3GNL “downlink” behaviour when using packet sizes smaller then 2096 bytes, and assumed to be negligible for larger packet sizes.

changing computer systems

Changing the computer system from a notebook to iPaq (using Bluetooth as intra communication system) is negligible for the V3GNL “uplink” behaviour; i.e. the behaviour of both computer systems is similar. The influence of changing computer systems is negligible for the V3GNL “downlink” behaviour for all packet sizes under the assumption that the V3GNL network is responsible for the observed difference for packet sizes of 8384 bytes (no switch to second bearer).

influence of the Internet
The influence of making the Internet component “present” in the SoD structure is negligible for the V3GNL “uplink” and “downlink” behaviour when using packet sizes between 174 and 1572 bytes.

*application and transport service buffer sizes*

The influence of changing the transport service (i.e. TCP) and the application protocol buffer size is only significant for the transport service if the application buffer size is larger or equal to the transport service buffer size. In the scenario when the V3GNL network becomes overloaded, increasing the buffer size of the transport service results in higher average delays: 32 and 64 Kbytes buffer sizes will have average delays of respectively 4820 ms and 8722 ms. The average delays remain constant until the load on the V3GNL network reduces (due to TCP flow control behaviour). The “goodput” and efficiency obtained for 32 and 64 Kbytes transport service buffers are respectively: 20 Kbps and 32%, and 13 Kbps and 21%.
8 Recommendations and overall conclusions

The MobiHealth project proved that the current UMTS technology is up to date for the creation and deployment of new mobile health services. Nevertheless there are many barriers and technological details that need to be resolved before stable and viable services can be introduced in the market. In this section we present a description of the issues and problems identified during the project, coming from all players in the mobile health services domain: i.e. hospitals, operators and technology companies. We do not mention the patients and the doctors, because it is not for them to adapt to new technologies but rather the new technologies must be adapted to their needs. Of course we must note that some of the issues described in this document are somewhat dated, meaning that they were valid at the moment of writing. The field being so dynamic, certain issues may have been resolved (we hope!) a few months after the writing of the report.

8.1 Hospitals

The main problems identified in the introduction of new mobile health services based on Internet technologies in hospitals relate to the changes required to support these new services, in both technological level and work practices level. From a technology point of view, the introduction of new mobile services requires a modern ICT infrastructure with secure connections to the Internet. However, many hospitals today do not have this kind of ICT infrastructure. Some hospital IT departments didn’t even accept the use of standard HTTPS (i.e. HTTP over secure sockets) to retrieve vital sign information. As a result any new mobile health service will need to be introduced as a stand alone service, not connected to the hospital ICT infrastructure.

A second technology related problem concerns the precaution measures taken by the majority of the hospitals in the use of wireless communication devices inside the hospitals. Due to lack of serious studies regarding the interference of wireless communication devices, like telephones, with medical equipment, most hospitals prohibit the use of any such device within their premises. This has as an immediate consequence that any wireless medical system, like MobiHealth, will have difficulties to be officially authorised to function inside the hospital. What is interesting however is the fact that some operators promote the use of GPRS/UMTS as a wireless communication network for use inside the hospital, as an alternative to WLAN. The argument being that one way or another GPRS/UMTS signals are present in the hospital and thus why not take advantage of it. This however does not resolve the question of the use of Bluetooth. Although Bluetooth technology makes use of the same ISM frequency band as WLAN, we believe that an in depth study is needed that will allow the identification of possible interference problems and possibly the creation of medical-non-interference label/accreditation for the new wireless medical devices.

A final problem for hospitals relates to the changes that will be required in the work environment and the work practices of medical personnel. Before the introduction of new mobile health services can take place, an in-depth study should be undertaken to identify how these new services can be introduced and what is really needed. For example in the case of trauma care, when the paramedic wears a video camera sending the video to the hospital, his role is totally different: he becomes a cameraman who will be possibly receiving instructions from e.g. a trauma surgeon at the hospital where to point the camera. How this will influence his work, needs
to be evaluated. Nevertheless, in this specific example we can anticipate other solutions, like tele monitoring robots that are controlled from the hospital. In general, the changes in the workplace will need to be evaluated on a case per case basis to avoid an effective degradation of the currently offered services, due to time and effort losses in mastering the technology, or due to flooding medical professionals with possibly a lot of irrelevant information.

8.2 Operators

The analysis of the collected measurement data during the trials allowed us to identify a number of problems that need to be resolved by the operators in order to allow the development of robust mobile health services. In the following we summarize the most important results and recommendations regarding the current status of UMTS networks.

Data rate
An important problem of the UMTS network is the difficulty for the user to establish high uplink and downlink bandwidth for variable and fixed data rate transmissions.

The UMTS network allocates different dedicated bearers for uplink and downlink channels depending on the data volume generated (uplink) or requested (downlink) by the user. When a user is generating or requesting low data volumes, the network allocates the common bearer (lowest bandwidth). On the other hand, if the user is generating or requesting high data volumes the network will allocate a dedicated bearer; uplink bearer 1 (64 Kbps) and downlink bearer 1 (64 Kbps), bearer 2 (128 Kbps) and bearer 3 (384 Kbps).

When the user is silent for a couple of seconds the allocated bearer is switched to the common bearer. If the user now wishes to transmit or receive a high volume of data again, he will not have immediately available the required highest bearer (i.e. highest bandwidth). The UMTS network will have to switch the uplink incrementally from the common bearer to bearer 1, and the downlink from the common bearer via 1 and 2 to the highest bearer 3. As a result the application risks loosing data due to buffer overflow due to the inability to transmit data at the required bandwidth. Thus variable data rate transmission is more complicated than other wireless or wired networks.

On the other hand fixed data rate transmissions also face some problems. It takes (to much) time for the UMTS network to switch from the common bearer to a dedicated bearer, data has to be buffered at the sending end (e.g. MobiHealth BAN). Buffering data will result in a variable delay and buffer overflow due to capacity limitation may result in temporal application termination or data loss.

The exact conditions and policy for bearer allocation should be made available by the operators to the application designers so that an optimal method for the use of the available bandwidth can be designed.

Delays
During the trials we observed that the uplink delays in the UMTS network increased linearly with the size of packets. However the observed delay variation (jitter) was very high due to bearer switching. The implication of the high jitter is that buffering of data is required to compensate for the delays. This is in line with the use of IPv4, which does not provide any Quality of Service
(QoS), but it will not be acceptable in the future IPv6 environment where a far better QoS will be required. Further fine tuning of the network will be required.

Handovers
The operators associated with the MobiHealth project have implemented UMTS soft handovers (from one cell to another). During the trial execution phase we discovered in general no significant impact on the behaviour of the MobiHealth application. However there were some occasions of connection loss during horizontal handover. The reasons of this connection loss were not clear to us, and we were not able to reproduce the problem, which seemed to be random. We presume that it was due to transient problems (we were running on a pre-commercial network) which will be (or are) already solved.

A GPRS to UMTS (and vice versa) hard handover scenario was not supported by all the operators. In our trials we had the opportunity to test (in an ad hoc manner) UMTS to GPRS handover on the Swedish Telia network using dual mode terminals. Although the handover worked correctly (i.e. the IP context remained the same), we observed a high delay during the handover process and a temporary interruption of the communications. The delay observed was between 10 and 20 seconds. This created problems in the MobiHealth application since during this time the data needed to be buffered leading in many cases to buffer overflow and data loss. In addition we were not able to find out (neither our contacts in Telia were able to tell us) when and under what conditions the handover between UMTS and GPRS takes place. We thus would like the operators to define the handover policy (when and how) and to reduce the handover delays. The information regarding the handover (when, what bandwidth will be next available etc) should also be available to the application designers.

Bandwidth
A major issue in our trials was the available UMTS bandwidth. For the time the available bandwidth of the UMTS network is far below the “dream” 2 Mbps, the operators do not yet support this bandwidth. Nevertheless, in the Netherlands we measured a steady bandwidth for downlink of 384 Kbps (netto: 270Kbps due to overhead), and 64 Kbps (netto: 57 Kbps) for uplink. These figures were stable and were tested also with moving terminals (up to 60 Km/h). However the traffic model of UMTS networks should be reviewed by the operators and industry so that it takes into consideration the fact that end users can also be producers of information and not only consumers (inverted producer – consumer paradigm). This will have implications in the bandwidth allocation and the design of terminals, all of which do not allow, for the time being, high data transmission from the user.

IP address allocation
Different operators have different policies regarding IP address allocation of the mobile devices. Some allocated public IP addresses, thus making them visible directly from the Internet, while others use private addresses making the mobile devices invisible from the Internet. Both solutions have advantages and disadvantages, depending on the application. We believe that the operators should allow the application providers to choose which model they want to use for their applications and not impose the one or the other model.

Communication costs
A major issue in the development of new medical services will be the communication costs. From our trials we have observed that continuous monitoring of vital signals will generate in the order of magnitude of 10 MB per day per user. With the existing cost policies the overall communication costs over a period of just one month will make the application cost prohibitive. We expect that the operators will introduce a different cost model for continuous transmission applications, like for example a flat charge for unlimited data and usage (as is the case today of some operators offering flat cost unlimited use for GSM communications).

8.3 Technology suppliers

Communication information and control
One of the problems we observed in our trials was the inability to obtain information on available bandwidth during the operation of the system. Although the PCcard UMTS terminals support a proprietary interface that provides information on the bandwidth available, this interface can only be accessed when the card is not under operation. We would thus like the card manufactures to provide a standard API for communication control and information.
Some of the functionalities we would like to access are, dynamic available bandwidth information, the actually used bearer, bearer changes, and control of UMTS to GPRS (and vice versa) handover. Clearly some of these functionalities can only become available if the operators are able to provide the related information and control.

Power supply for the terminals
A major problem in the mobile medical applications is the limited power supply. A UMTS terminal (e.g. Nokia telephone) transmitting data continuously will empty its battery in less than 2 hours (at best). More research in alternative power sources needs to be conducted.

Bluetooth IntraBan communication
A bandwidth limitation for the intra-BAN communication comes from the limited bandwidth of the Bluetooth interface supported by the Nokia UMTS telephones. Although the USB connection of the telephone supports high bandwidth, the Bluetooth connection has a bandwidth limited to 115 Kbps, much lower than the 384 Kbps of the UMTS downlink. We hope that in the future the Bluetooth connection will support higher bandwidth, so that it is not the communication bottleneck.

The processing power of the Bluetooth hardware in the front-end device (a.k.a. Mobi) also proved a bottleneck. Only a maximum throughput of 160kbps of the theoretical maximum of 720kpbs could be achieved. Although this has been sufficient for the trials executed in the MobiHealth context, with other trials that require more sensors (for example additional ExG signals) this will soon become a bottleneck.

Standardization
Within the next few years the mobile health services will start becoming available at a large scale. However, to promote the development of value added services and to facilitate the interoperation of these services, several standards need to be designed and applied. Standards for medical systems being operational in a MobiHealth BAN need be developed, as well as the
standards for representation and transmission of vital signal measurements (of the many that exist today) might need to be revised or tailored for mobile applications.
References


[Have1998]


[TechWeb] [http://www.techweb.com/encyclopedia]

[Viss2000] Vissers C.A., Ferreira Pires L., Quartel D.A.C., van Sinderen M.J., The architectural design of distributed systems, University of Twente, November 2000

[Webster] [http://www.m-w.com/cgi-bin/dictionary]
Appendix 1  System Parameters of Influence

<table>
<thead>
<tr>
<th>System</th>
<th>Operating System</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>notebook</td>
<td>Microsoft Windows XP Professional, Version 2002, SP1</td>
<td>Pentium III, mobile CPU 1GHz</td>
<td>640 MB</td>
</tr>
<tr>
<td>pc1</td>
<td>Microsoft Windows XP Professional, Version 2002, SP1</td>
<td>Pentium 4, 2.4 GHz</td>
<td>512 MB</td>
</tr>
<tr>
<td>pc2</td>
<td>Microsoft Windows XP Professional, Version 2002, SP1</td>
<td>Pentium III, 600 MHz</td>
<td>128 MB</td>
</tr>
<tr>
<td>iPaq 3870</td>
<td>Linux 2.4.19-rmk6-pxa1-hh13</td>
<td></td>
<td>64 MB</td>
</tr>
</tbody>
</table>

Table 1  Computer systems.

Appendix 2  V3GNL Leased line infrastructure
VPN: CORP_Utwente
APN: utwente.nl
Roaming: No
Date: 23-jul-03
Version: 03

Utwente assigned IP-Address