Product Modeling & Realization Group (PM&RG)

INNOVATION PROTOTYPING

Innovation prototyping methodology of PM&RG allows designing and experimentation with service ideas of the future. The key elements of the methodology are illustrated in figure 1.

Balanced brokering allows multidisciplinary collaboration of experts such as software, sociology, psychology, usability, business etc. Active work is needed to help the experts understand each other.

Experimentation gives the concrete *justification and proof* for the crucial features of the innovation. Experimentations in pre-product development are focused, systematic and require solid methodology

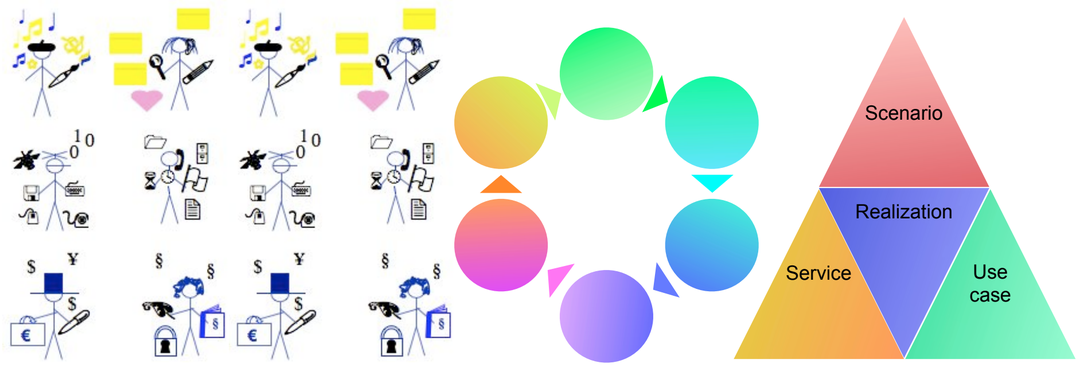


Figure 1: Balanced brokering, experimentation cycle and SSUR modeling.

Design data modeling enforces the design to be analytical and systematic to take advantage of the unexpected and the unintended.

Innovations are nothing miraculous, or created by bouncing unsubstantiated ideas. Instead, innovations are results of systematic and analytical designing.

APPLICATION AREA

The application are of Innovation prototyping methodology is pre-product development of mobile and ubiquitous computing services. Pre-product development is an ongoing process that aims to find well-defined problems for the product development process to solve. Since it is a non-convergent process, it requires concepts and methods that are not applicable in product development.

Ubiquitous computing (ubicomp, jokapaikallinen tietotekniikka) was introduced by Mark Weiser in early 1990's. From his definition and examples, PM&RG has derived three central features of ubicomp. The ubicomp services must be provided in some form even though in ubicomp environment everything changes dynamically all the time when the service is being used

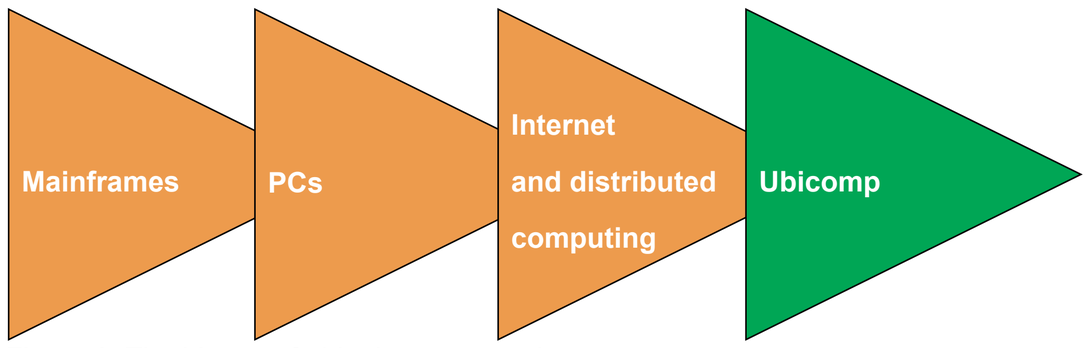


Figure 2: The history of ubiquitous computing

1. Availability and utilization: Ubicomp services can be used everywhere utilizing the facilities available in the environment and in accordance with the situation.
2. Invisible computing: In ubicomp environment the devices and computers disappear and the interaction is done with everyday objects.
3. Concentrating on the task: People will be able to concentrate on the task at hand instead of concentrating on how to operate the computers.

RESEARCH GROUP ACTIVITIES

Research

Main areas are

- Innovation prototyping methodology: design data modeling, solid experimentation methods and combining complementary expertises.
- Mobile and ubicomp applications: system architectures, challenges of heterogeneous networks and gadgets, designing for real world conditions and users, as well as instrumentation of experimentation settings.

Key enablers for research are coordination, integration and application development in projects such as GO, VHO, MERCoNe and WISEciti.

Education

Main objective is learning how to apply research methods in product development.

- Assignments and courses enforce setting a well defined focus and making a literature survey on methods, latests results and new approaches of research.
- PM&RG guides students in preparing justified and methodologically sound experimentation plan including focused hypothesis.
- Students can use software, gadgets and network facilities for realizing their experimentation settings and innovation prototypes.

Collaboration

Main ways of collaboration are

- Expert networks: PM&RG has a hangaround and special interest groups, and participates in Tekes thematic groups.
- Students of Aalto and beyond: tutoring a student group yearly, and involving B.Sc., M.Sc. and Ph.D. workers in the group activities.
- Wider audience: arranging open events on essential topics that give a wider view on other areas of expertise.

Ultimate goal: Educating engineers who have well-founded respect for and literacy on other disciplines; contributing to the Aalto handicraft connection; and contributing to the engineering ethics.

Modeling and realization

Foundation of research, education and collaboration is on design data modeling and its lifecycle. The ultimate goal is to allow the designers to consider involved expertises concurrently. For this goal, ubicomp paradigm is the best real world application area with the visions reaching far into the future. Design data modeling allows the designers to take advantage of the enablers produced by other expertises and combine the enablers in novel ways to produce innovations. Realizing enablers and systems for experimentations makes it possible to justify and prove the key elements of ubicomp services of the future.

Contact information

Leader:
Mervi Ranta

Coordinator:
Henrik J. Asplund

Email:
pmrg@tkk.fi

WWW:
http://www.cs.hut.fi/~pmrg

Telephone:
+358 9 470 4807

Aalto University
School of Science and Technology

Experimentation cycle and SSUR models in Innovation prototyping methodology

Mervi Ranta and Henrik J. Asplund

Keywords: experimentation cycle, SSUR modelling

Experimentation cycle

Innovation prototyping methodology utilizes experimentation cycle that contains six stages to research ubiquitous computing (ubicomp).

1. **Knowledge acquisition** means finding from literature existing research, including methods appropriate for the field. Also, results from earlier experimentations are considered.
2. **Processing and selection** prepares the gathered material for use in experimentation planning. Also, potential hypotheses, methods and approaches are chosen to set the focus for the experimentation.
3. **Experimentation planning** stage starts with explicating the main hypothesis. The methods for data analysis and gathering are selected to allow proving or disproving the hypothesis. Also, the methods set the requirements for experimentation setting and instrumentation, and therefore allow defining what needs to be implemented.
4. **Realization of the setting** stage produces the concrete experimentation setting, i.e. networks, prototypes, software and other instruments that are needed to carry out the experimentation.
5. **Experimentation** stage means carrying out the experimentation plan. In this stage, the experimentation data is gathered. Experimentation requires careful application of the methods and execution of the plan, in order maintain validity.
6. **Organizing and analysis of the data** means first structuring the data from the experimentation stage, and then applying the chosen analysis methods to produce results that validate or invalidate the hypothesis.

Experimentations in real world environments allow discovering and analysis of complex relationships, implications and interoperability issues. Innovation prototyping methodology is experimentation-driven, due to focusing on common denominators and analyzing the problem, instead of producing product prototypes.

To maintain consistency and coherency through whole experimentation cycle, and to ensure transparency, explication and persistence of information, all the generated information is stored in SSUR models.

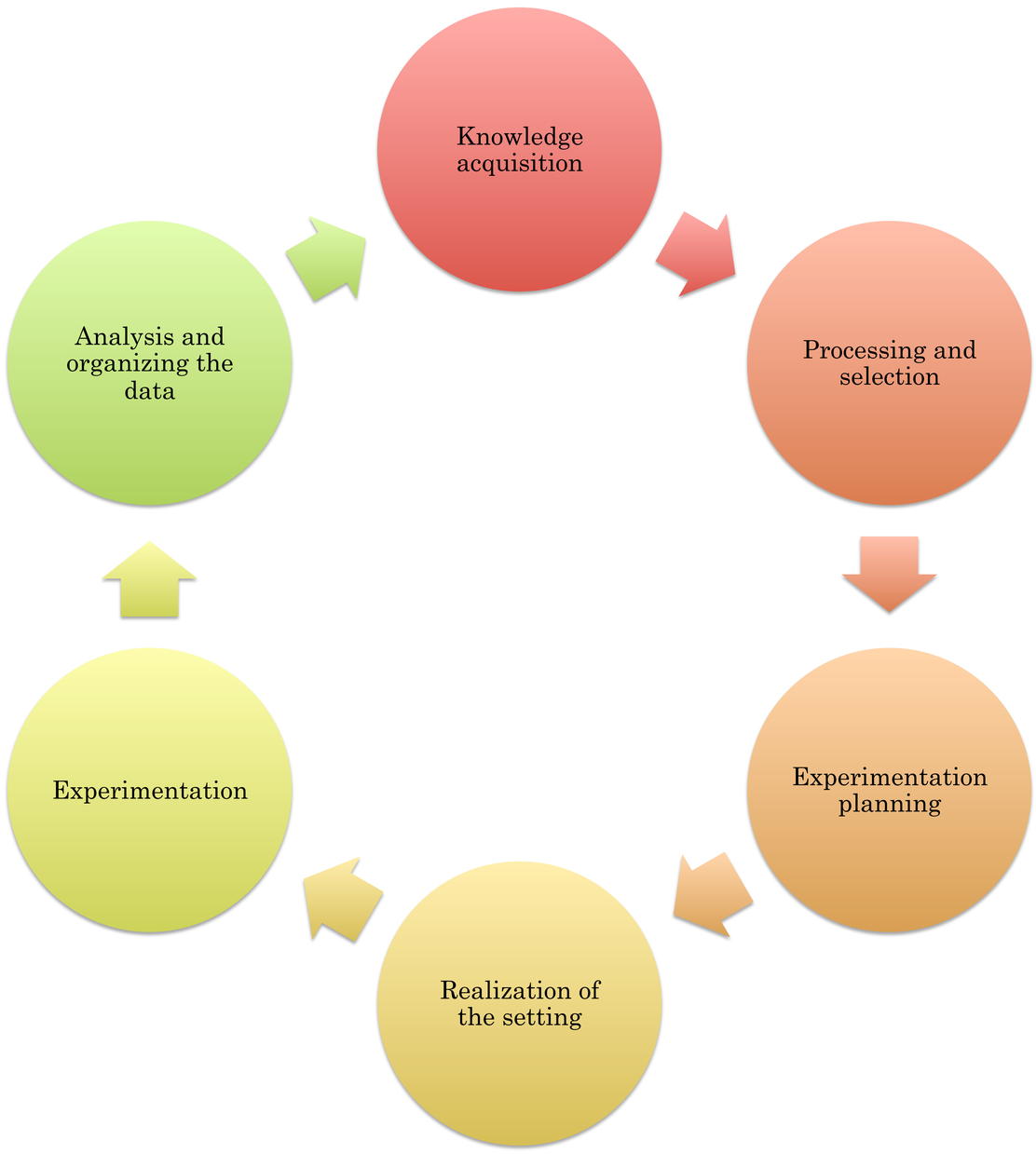


Figure 1 – Experimentation cycle

SSUR models

SSUR consists of four distinct models, i.e., scenario, service, use case and realization. Each model has a different purpose, and they are targeted at analysis and brokering of different aspects needed in Innovation prototyping.

1. **Scenario** explicates the real life manifestations of the service. Scenarios contain information on situations in which the users utilize the service. Scenarios are written in colloquial language and in narrative form.
2. **Service** describes the profit model, stakeholders and contracts needed to provide a ubicomp service to consumers.
3. **Use case** describes the interfaces between the constituents of the service, which can be humans, companies, systems etc. Ubicomp services are typically not monolithic entities, but rather systems of systems. Use cases explicate what information is transferred between the constituents, instead of describing a protocol or the transmission process. Use cases ultimately explicate the semantics of the transferred information. Use cases act as handles to the enablers that allow a ubicomp service to be created.
4. **Realization** model explicates the experimentation setting, selected data gathering, organization and analysis methods and the hypothesis.

Ubicomp requires advanced technology, but the point is in how the user perceives the ubicomp service. Ubicomp does not itself define any specific set of technologies that can be utilized in the creation of the service.

In Innovation prototyping methodology, a service is designed just to prove common denominators of the ubicomp services and properties of the enablers. Designing and carrying out valid experimentations is the goal of the whole methodology. Instead of creating just one product, Innovation prototyping produces knowledge that can be used to create completely new types of services and understanding all the consequences of ubicomp.

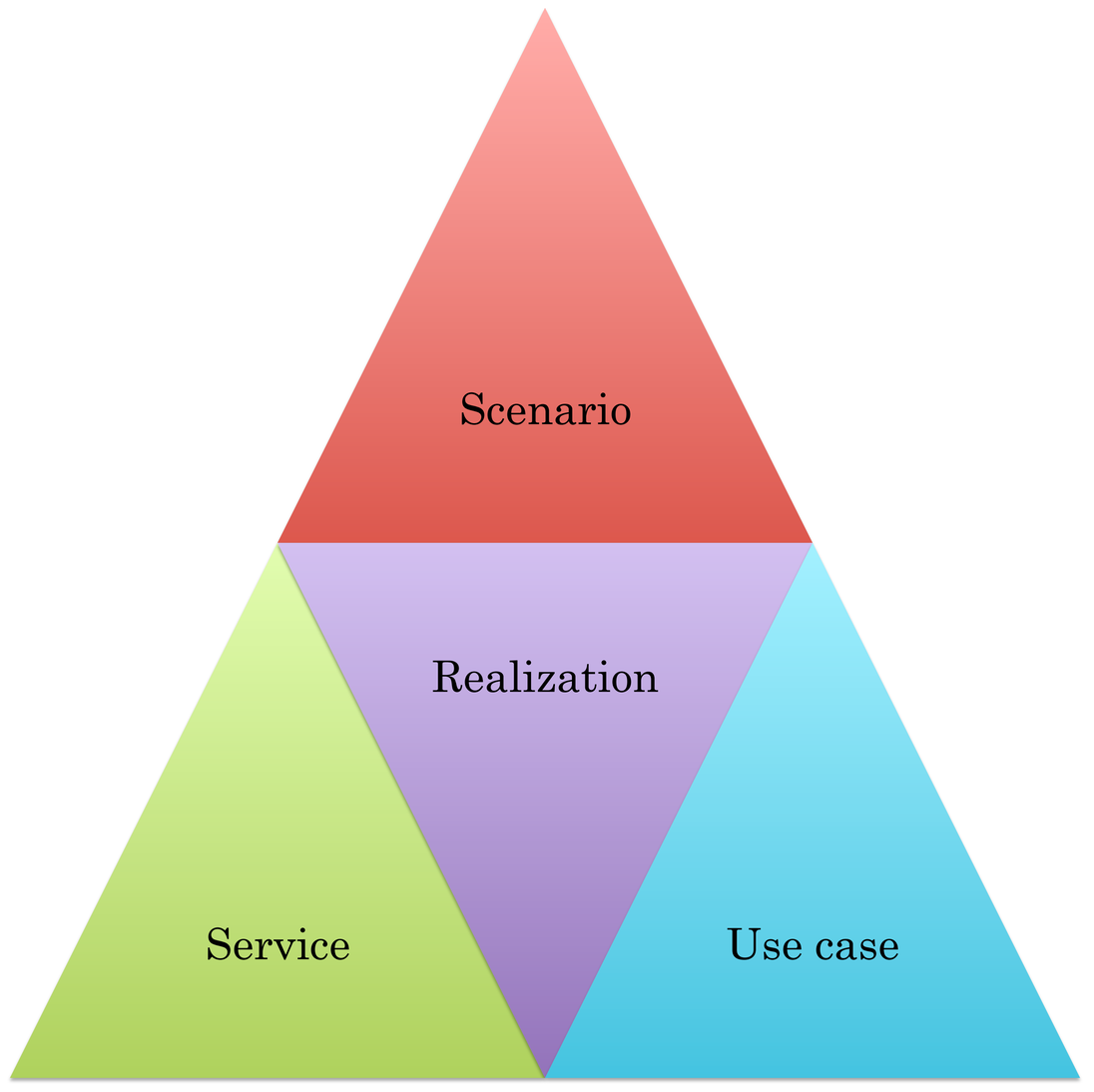


Figure 2 – SSUR models

Product Modelling & Realisation Group (PM&RG)

Email: pmrg@tkk.fi
WWW pages: http://www.cs.hut.fi/~pmrg
Telephone: +358 9 451 4807

Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

Ubiquitous computing according to Mark Weiser

Eetu Pilli-Sihvola

Keywords: ubiquitous computing, ubicomp, calm technology, prototypes

Central concepts

Ubiquitous computing makes hundreds of computers available in our everyday environment while keeping them effectively invisible to the user. It turns computers from being the focus of our attention to being a tool that assists us. [1][2]
Virtual reality makes the computer invisible by taking over human senses and providing a reality removed from the everyday physical world. [2]
Calm technology refers to technology that engages and moves between the center of our attention and the periphery. It helps react to aberrations around us and tune out things that don't require immediate attention. [3]
The periphery describes the things that we are aware of without paying specific attention to them. Things in the periphery can move quickly to the center of our attention and back again. [3]

Surveyed articles

[1] Weiser, M. "The Computer for the 21st Century". ACM SIGMOBILE Mobile Computing and Communications Review, 1999. Vol. 3:3. p. 3-11. ISSN 1559-1662.

[2] Weiser, M. "Some computer science issues in ubiquitous computing". Communications of the ACM, 1993. Vol. 36:7. p. 75-84. ISSN 0001-0782.

[3] Weiser, M. & Seely Brown, J. "The Coming Age of Calm Technology". Xerox PARC, 1996. [Cited 21.10.2008]. Available from: http://nano.xerox.com/hypertext/weiser/acmfuture2endnote.htm.

Contents

Mark Weiser is in many instances referred to as "the father of ubiquitous computing". He first used the term in 1988 while working for the Xerox Palo Alto Research Center.

•The most influential technologies fade into the background.
•Computers of the 1990s are the center of our focus rather than a tool that assists us.
•To be effective computers must become a part of our normal life so that no special attention needs to be paid to them.
•***Ubicomp*** makes hundreds of computers available in our everyday environment while keeping them practically invisible to the user.

•***Virtual reality*** is practically the opposite of ubicomp as its premise is to fool the user by leaving the everyday physical world behind.

•***Calm technology*** engages and moves between the center of our attention and *the periphery*.
•***The periphery*** describes the things that we are aware of without paying specific attention to them.
•Things in the periphery can move quickly to the center of our attention and back again (e.g. driving).
•Helps pay attention to more things and react to things that don't seem right.

Weiser designed and built three types of ***prototypes***:

Tabs
- ➢Tiny computers (analogous to e.g. Post-it notes)
- ➢Active badges are certain types of tabs (location information)
- ➢Hundreds per each person in an office

Pads
- ➢Analogous to scrap paper
- ➢Tens per each person in an office

Boards
- ➢Wall-sized interactive surface (analogous to e.g. office whiteboard)
- ➢One or two per each person in an office

Conclusions

Ubicomp classification	Description
Mobile/Hotspot	The surveyed articles describe mobile users in the workplace and in home environment and the services these environments can offer. Challenges include user identification, power consumption and wireless communication. At home the outside surface of the refrigerator door, for example, can serve as an electronic bulletin board and provide information and reminders of various things. At work different kinds of electronic pads, tabs and boards can be used similarly to store and exchange information.
Object/Environment	The surveyed articles describe computing that is both embedded into objects and distributed to elements of an environment. Computing is embedded into objects that can store and display information (electronic notepads, smaller electronic tabs that correspond to Post-It notes, electronic boards that are the equivalent of office whiteboards), that can be used in identifying people (active badges) or locating people and objects (any object that knows its location). Environment is enhanced to support these applications (location information, identification, etc.).
Context	The surveyed articles describe the use of identity information as a means to implement access control in the workplace, i.e. employees are only allowed access to areas and resources they need in their work.
User/Industry	The surveyed articles describe equipment and applications for both end-users and business purposes. End-users could benefit from electronic bulletin boards at home whereas identity information (active badges) could be beneficial for various kinds of businesses.
System/Enabler	The surveyed articles focus mainly on enablers, such as location information, identification and shared drawing. They also describe a system of electronic devices that can store and display information and work seamlessly together using wireless communication .

Aalto University
School of Science
and Technology

Decoupling application architecture design and network layers

Conclusions on WISEciti project / PM&RG

Introduction

When mobile and ubiquitois computing applications are considered, it is important to take into account the constraints that rise from the energy and networking requirements, also in designing the software architectures and the information consumption of a system. Another important issue is the fact that all the information sources or sinks are not available all the time, but connectivity may be lost. Coping with these issues in every application would be time consuming and a source for innumerate amount of errors. Figure 1 shows the proposed architecture including three APIs that will be needed at least in the future.

VHO API

The application has to be able to affect network interfaces and functions of network layers. Also, the application has to be able to receive information on network parameters and state, no matter which specific technologies and implementations are on the network layers. The logic and ontologies of application programming are very different from how the network operates, and therefore it is necessary that there is an API that translates the information and commands from application to different network layers, and vice versa.

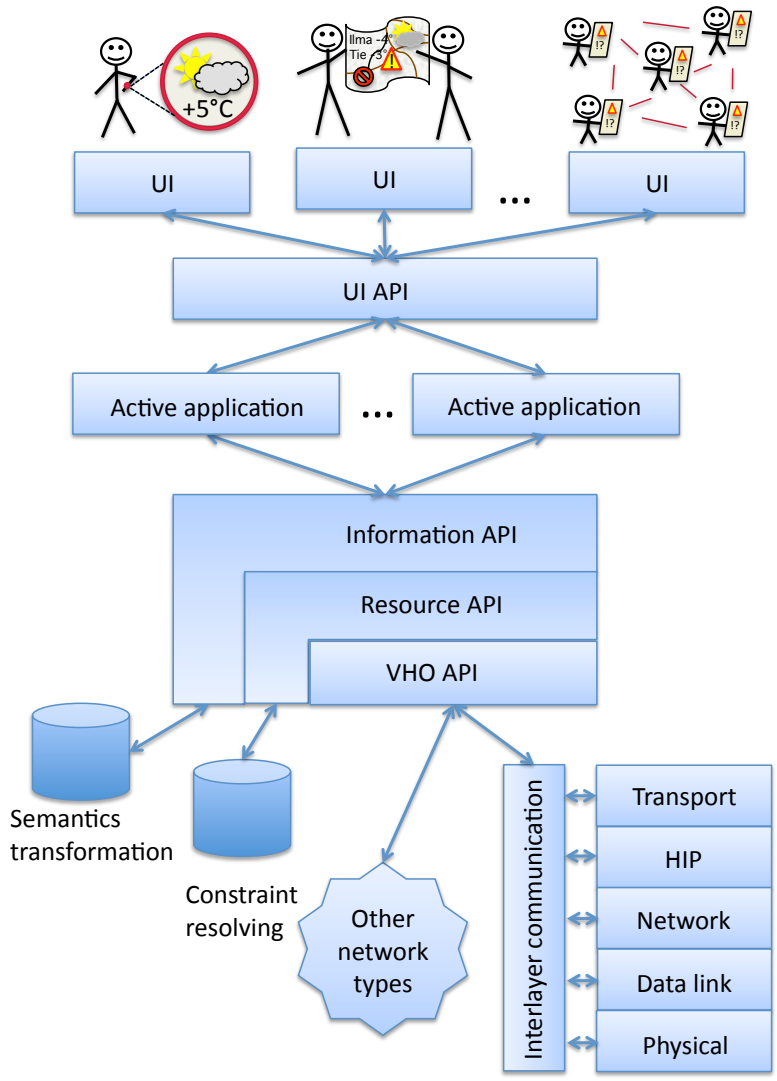


Figure 1: Proposition for the future architecture of mobile and ubicomp applications

Resource API

Energy consumption is a tough issue in mobile and ubiquitous computing. Therefore, a resource API is needed so that unnecessary information transfers and and processing can be avoided, or specific network technologies can be selected based on the contents, to avoid consuming too much battery. Resource API handles caching and information lifecycle issues. Resource API bridges the gap between application logic, the information consumption of the application, and the energy consumption of the device.

Information API

In order to be able to access information, the application needs to be able to address it in a manner that does not define specific networking technology or protocol. Information API is targeted to allow for addressing information based on its semantics, instead of its location. In ubiquitous computing, location of the information is volatile, and making naïve assumption that information is always fetched from the network is not an acceptable solution, due to the constraints caused by available battery power and limited connectivity.

PM&RG research group
Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

eMail: pmrg@cs.hut.fi
WWW: http://www.cs.hut.fi/~pmrg
Visiting address: Konemiehentie 2, Espoo, Finland

Aalto University
School of Science
and Technology

Analyzing and modeling mobile and ubiquitous community services

Conclusions on WISEciti project / PM&RG

SSUR models

SSUR (Scenario, Service, Use case and Realization models) are used in innovation prototyping methodology. A service model explicates the system that appears implicitly in the scenarios and the use cases. It puts the system in the context and shows how the system can be an enabler for a system of systems.

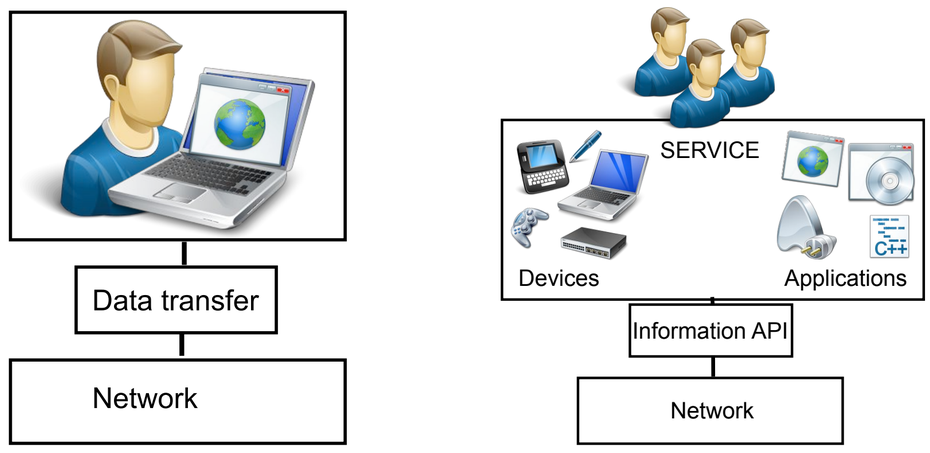


Figure 1: User includes device, application and the human user

User: consumer / industry

These models characterize where the ubicomp service is targeted at. Some systems are created for end-users, i.e., ordinary consumers. They naturally differ clearly from the systems created for industrial environment. User groups and communities are analysed with (e.g.) communication models, group and community models, and mobility models.

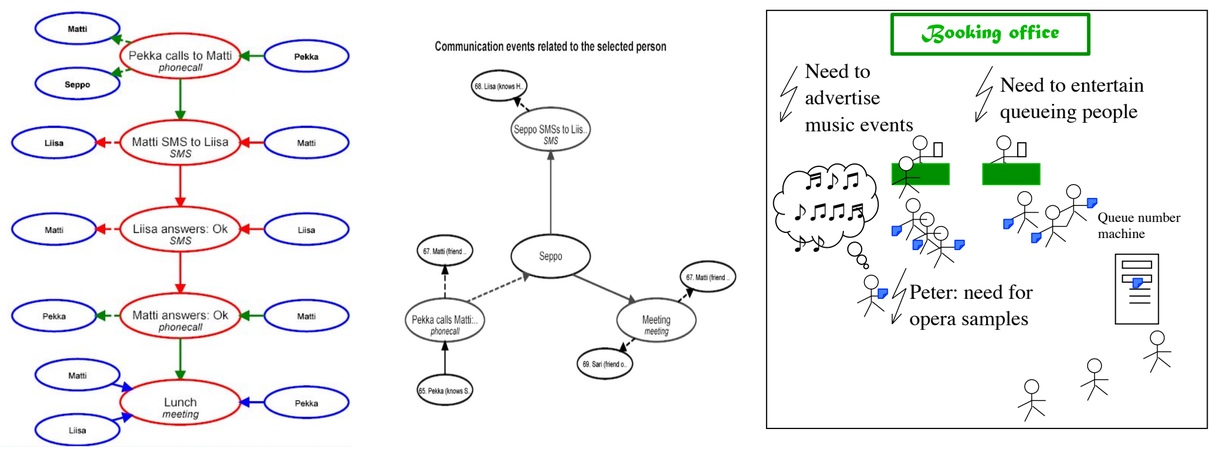
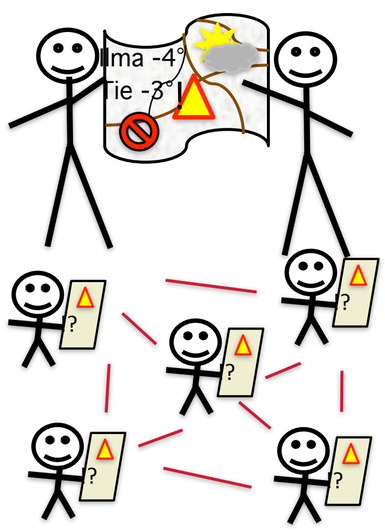


Figure 2: Communication, community and mobility model

Divergent service

Designing future mobile and ubicomp services is about planning a dynamic service provision chain, that assembles a suitable combination of available devices, (active) applications and access networks. Divergent service as an approach turns this challenge into an advantage by exploiting the chance for variation.



Object/environment
- Is the computing embedded in objects or distributed to the environment?

System/enabler
- Is the service a complete system, or an enabler to another system?

Mobile/hotspot
- Are the facilities or the user moving, or is the service available in one place?

Stakeholders

Profit model allows for analysis of contracts of stakeholders:
- Exchanging content, profit, network access etc.
- Which stakeholders provide the needed constituents of the service?

Mobile and ubicomp services are complex. Therefore the constituents of service provision must be analyzed.

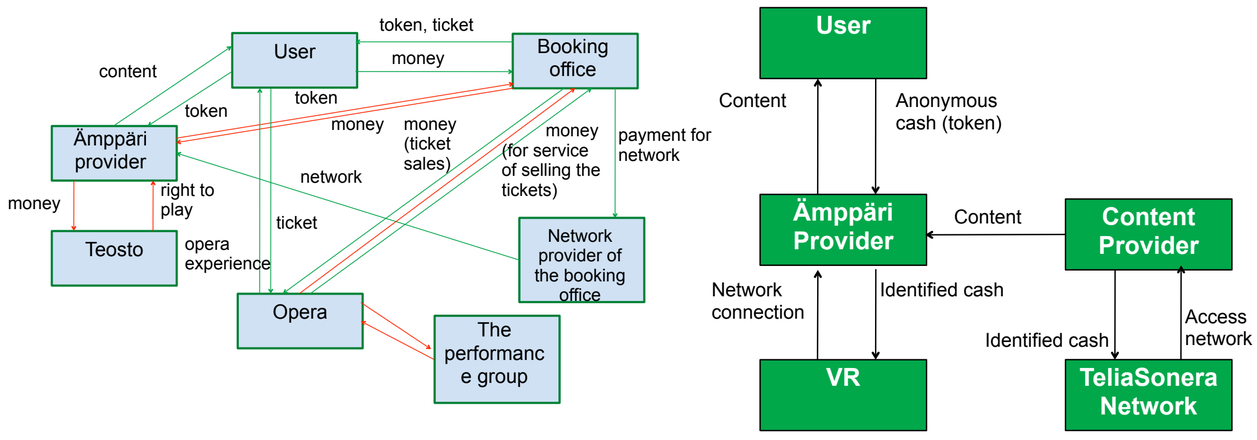


Figure 3: Two profit models on mobile music player provision chain

PM&RG research group
Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

eMail: pmrg@cs.hut.fi
WWW: http://www.cs.hut.fi/~pmrg
Visiting address: Konemiehentie 2, Espoo, Finland

Experimentation: Determining audio buffer size from LAN-WLAN vertical handover delay

Tatu Kilappa, PM&RG

Keywords: vertical handover, application-level, cache measurement, network audio playback

Introduction
The experimentation investigates the delay experienced by the application streaming audio over TCP/IP when an uninformed LAN to WLAN vertical handover occurs.

Measurements in the experimentation focus on recording the distinct cache sizes in the application receiving the media stream, and their behavior in the event of the handover. Additionally, the mobility management events and the network traffic during the handover are recorded.

The experimentation platform consists of Ericsson's hip4bsd for FreeBSD 6.1 and the PM&RG Ämppäri mobile music player, both versions developed in the MERCoNe project.

Hypothesis
The hypothesis is, that it is possible to determine the required cache size for the playing application from the cache size recordings done during the vertical handover.

Experimentation setting
The required cache size is specified as the minimum amount of cache as seconds of uncompressed audio that will not cause a depletion in the hardware or operating system audio buffer. The cache size is considered insufficent if the handover causes depletion in the OSS/hardware buffer, the size of which the software does not explicitly state.
The experimentation setting consists of a stream player and a stream sender, connected through a router. Both the stream player and stream sender use HIP, and the stream sender is connected to the router via both LAN and WLAN and uses LAN by default.

Experimentation procedure
The experimentation consists of the following procedure repeated 20 times for each cache size setting from 8192 bytes to 81920 bytes and for two distinct media bitrates of 192 and 320kbps. Constant bitrate MP3 audio is used for streaming, since every packet of audio data uncompresses to the same amount of audio.

1. Stream sender opens a connection to the stream player and starts playing an audio stream.
2. Playback continues for 15 seconds, after which the LAN cable is disconnected, forcing a switch to WLAN.
3. Playback continues in WLAN for 15 seconds.
4. Recordings are saved to the disk on program exit.

Recording file details
The audio processing path follows the conventions of typical audio players, there are three distinct audio buffers, the compressed audio packet cache, uncompressed ring buffer to which the audio packets are uncompressed and the OSS/ hardware audio buffer to which the uncompressed audio data is fed.
All data, incording the ESP packet indexes and the mobility management events are transformed into value-timestamp pairs, that are analyzed in MATLAB.

Phase	Methods	Result
Data collection	Recording of cache sizes, traffic	Packet cache size Ring buffer size OSS audio buffer size ESP packet capture LINK_DOWN timestamps HANDOVER timestamps
Data processing	Determining delays from recordings	Delay in ESP packets (ms) Delay in audio packets (ms) LINK_DOWN reaction time (ms) HANDOVER reaction time (ms)
Data analysis	Determining maximum handover time and typical delays	Maximum handover delay Audio packet delay distribution Reaction time distribution

Table 1: Summary of methods

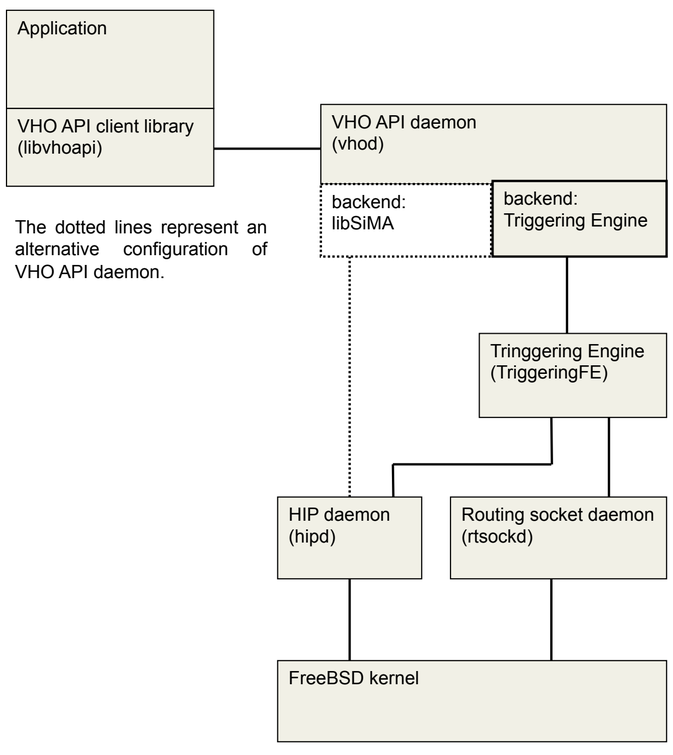


Figure 1: Mobility management architecture

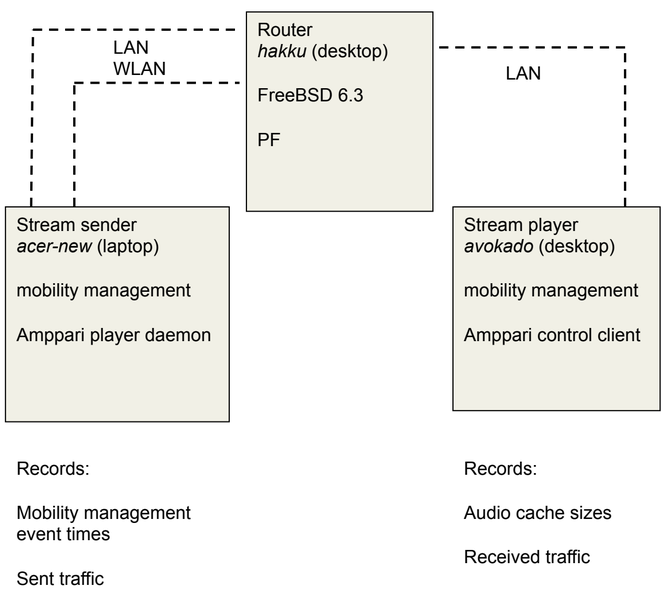


Figure 2: Test network configuration

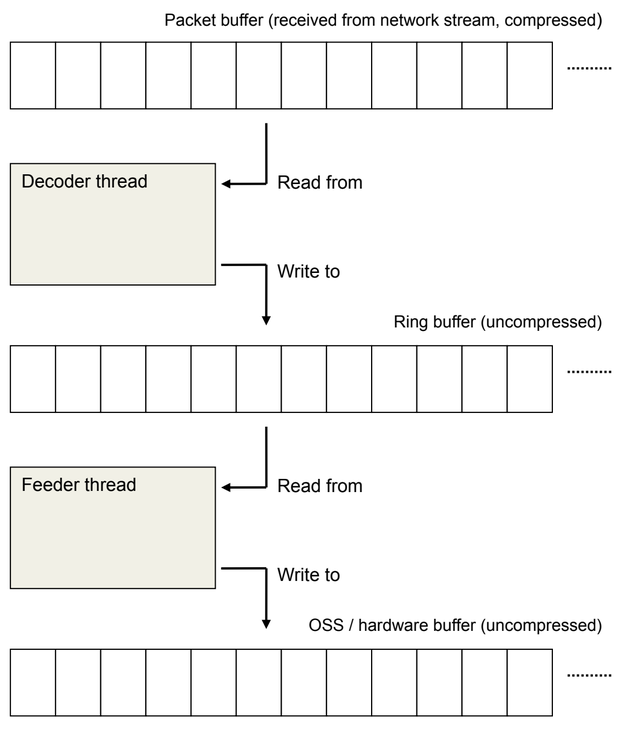


Figure 3: Audio processing path

Determining audio buffer size from LAN-WLAN vertical handover delay
Tatu Kilappa, PM&RG

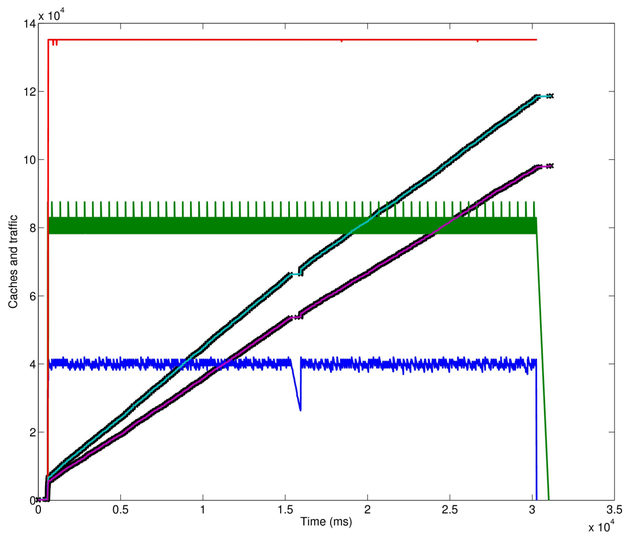
Email: pmrg@tkk.fi
WWW pages: http://www.cs.hut.fi/~pmrg
Telephone: +358 9 451 4807

Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

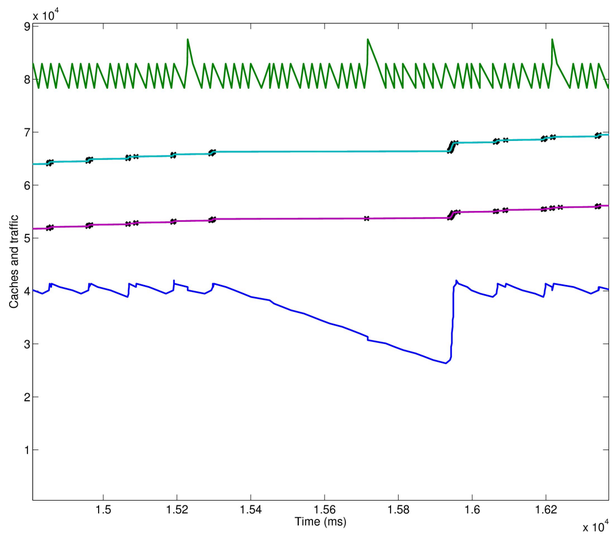
Experimentation: Determining audio buffer size from LAN-WLAN vertical handover delay

Cache behavior examples

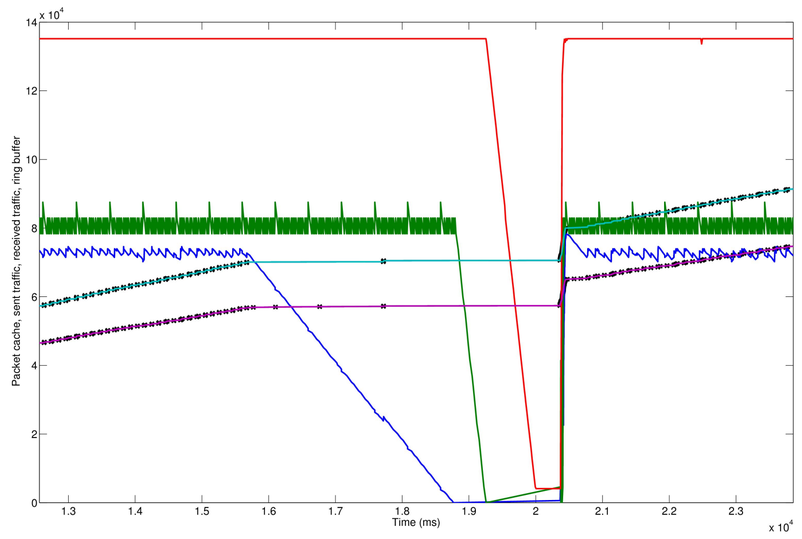
The following presents cache behavior in two example cases, the left being an ideal handover situation with a closeup, and the right one being a worst-case situation, where a buffer underrun occurs. Horizontal lines represent the packet cache sizes in bytes. The lowest line (blue) is the packet cache, the middle line (green) is the decoded buffer, and the top line (red) is the OSS / hardware buffer. The diagonal lines represent the sequence numbers of received (cyan) and sent (magenta) packets respectively, they index numbers are scaled to fit in the same image area.



Ideal case



Ideal case closeup

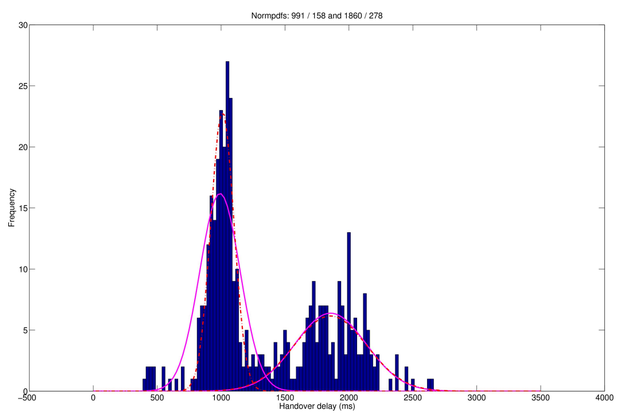


Worst-case with underrun

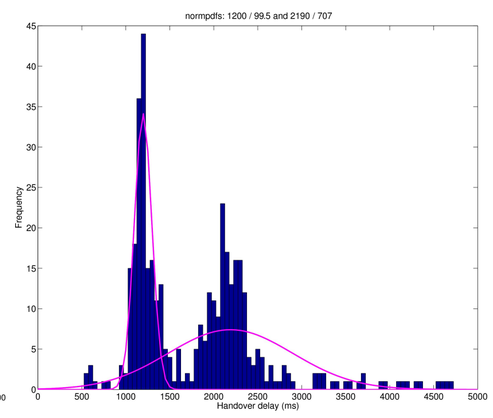
Delay distributions

Traffic delay from the handover is roughly divided between 0.5 and 2.5 seconds, but this measure does not give the full application-experienced delay in the media stream. As visible from the worst-case detail above, an individual packet is received in the middle of the gap, but the delay extends beyond it. The actual stream delay can be described as a combination of the two longest delays. 95% of the delays are in the range of 1.05 to 3.24 seconds.

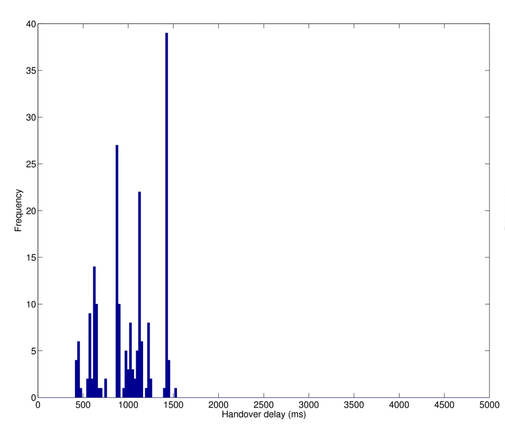
The distributions on the left are acquired with playback status messages sent individually. If they are sent coupled with new stream packet requests, the distributions are sparser, but 95% of the delays are in the same general range of 0.74 to 3.4 seconds.



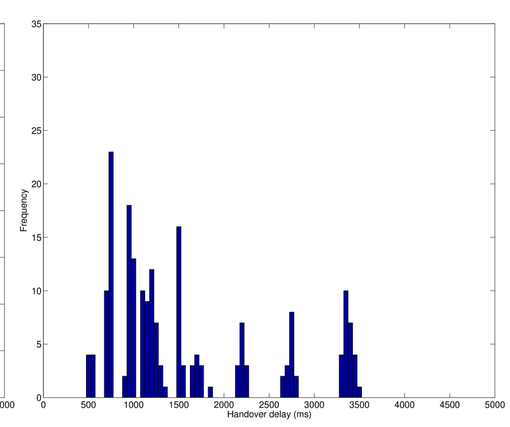
Traffic delay with approximating normpdfs



Packet cache delay (2 segments)



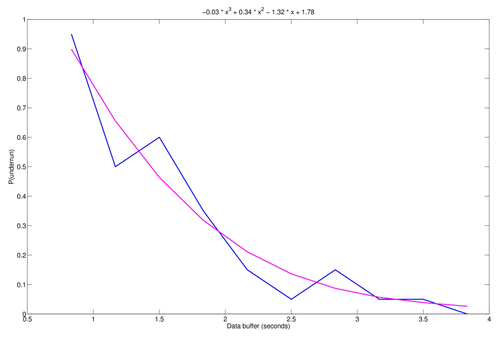
Traffic delay w/o status packets

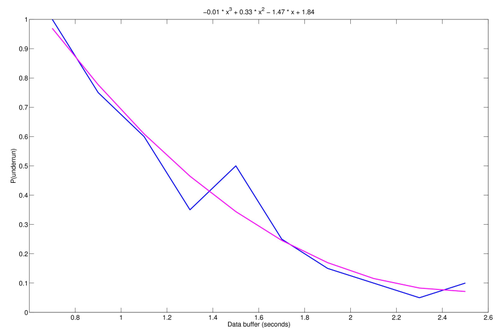


Cache delay w/o status packets

Underrun probability

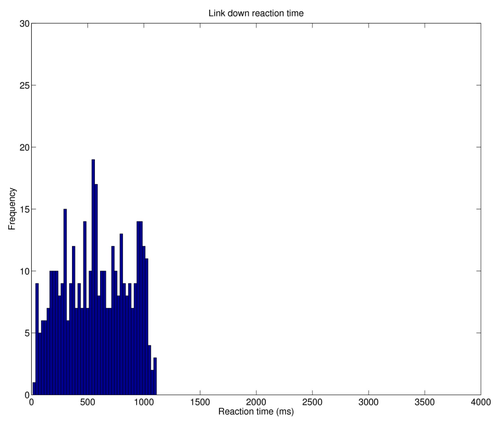
The delay described above gives a certain probability of a buffer underrun happening during the streaming. Underrun probability in the graphs below is presented for 192kbps and 320kbps streams both. Note that the underrun probability flattens to near zero after the 3 second mark, as expected from the delay distributions. Without status packets, the underrun probability curve is similar, but flattens to zero faster.



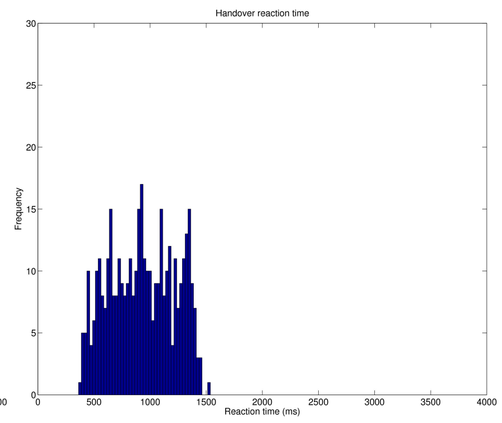


Reaction time

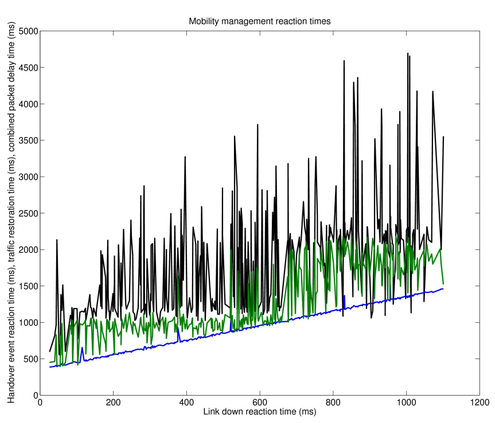
Reaction time in this context is defined as the time it takes from the last successfully recieved packet for the distinct phases to happen in the handover. The LINK_DOWN and HANOVER events are received from the mobility management, the handover event coming almost diligently 350 milliseconds after the link down event. Other graphs represent the combination of the events to the next successfully received packet and the media packet delay (2 segments). The combined reaction time graph in the very right is the result from embedding the status packets to the packet requests, which reduces the number of received packets.

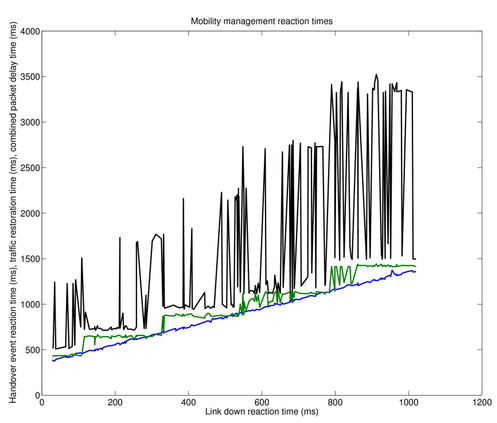


Link down reaction time



Handover event reaction time





Reaction times, traffic (green), packet (black) for status packets and w/o

Determining audio buffer size from LAN-WLAN vertical handover delay
Tatu Kilappa, PM&RG

Email: pmrg@tkk.fi
WWW pages: http://www.cs.hut.fi/~pmrg
Telephone: +358 9 451 4807

Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

Ämppäri user interface

Nadezhda Kasinskaja

From the user point of view, Ämppäri is a music and video player which, in addition to normal music player features, can use external speakers and screens wirelessly for playback. User's own or publically available speakers and screens can be used.
The music is stored and accessed online, which means that the available music selection is essentially unlimited. No break in music playback occurs if the user moves between different network coverage areas.

Ämppäri user interface prototype

Ämppäri user interface prototype was developed for testing the ideas behind Ämppäri implementation on potential end users. The testing areas are

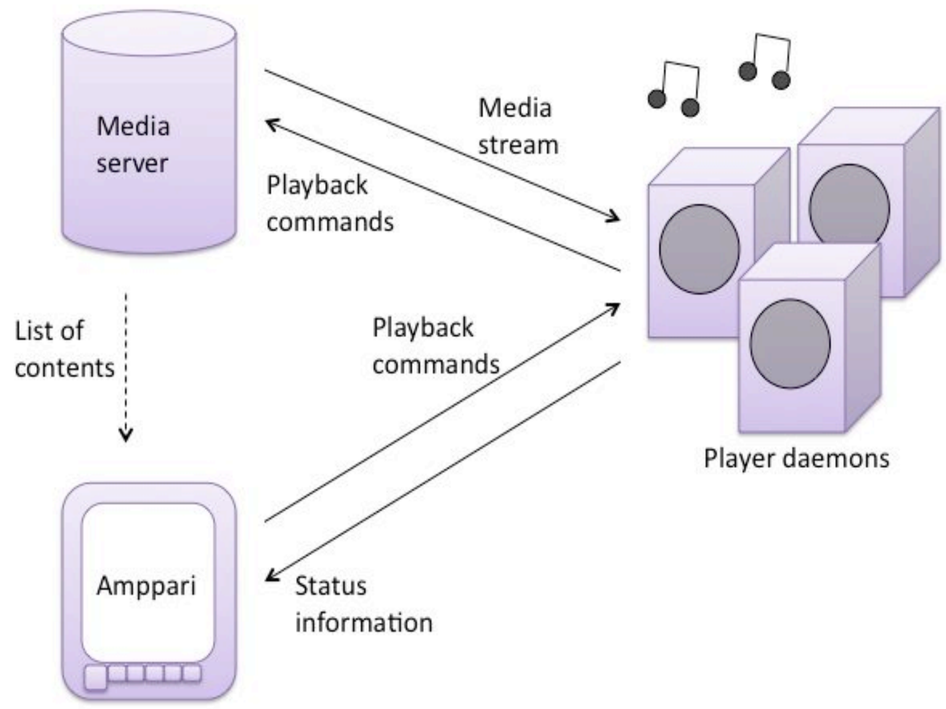
- selecting speakers and screens for playback,
- changing the selection during the playback,
- online music storage and access, and
- adjusting music quality when switching between networks with different capasity.

The objectives of designing the user interface are as follows:

1. Since Ämppäri is an experimentation tool that is made to allow researching the alternative designs and solutions, the user interface must be implemented to anticipate future additions and changes to the interface, and even to the functionality of the application.
2. The research characteristics of Ämppäri set the requirement that the user interface implementation must be limited only to the features that are necessary for the planned experimentation usage. A simple reason for this requirement is to avoid wasting effort on the unnecessary implementation. However, the really crucial reason is to ensure reliable experimentation with proper focus, limiting the elements that could affect the experimentation result to the minimum.

Implementation details

Ämppäri is connected to one or more player daemons (located in screens or speakers) via the network. The player daemon retrieves the music for playback from media servers using its own network connection and handles the low-level playback functionality according to the commands received from Ämppäri.
Ämppäri is implemented on an iPAQ h5500 PDA with WLAN connection using



User interface for a mobile music player - design and automated event logging (M.Sc. Thesis)

Nadezhda Kasinskaja

Keywords: Automated event logging, user interface event log analysis, design datas, mobile application

Introduction

The goals of this thesis are to present a case study on designing and developing a user interface for a music player prototype and to research the possible use of an automated user interface event log collection system in evaluating a user interface of a mobile application.
Since a mobile device is small, designed to be used by one person at a time, and often used in different and changing contexts of use, the recording and observation of usability tests performed on those devices is challenging. Automated tools can be used to facilitate usability evaluation and allow earlier and more thorough testing. An automated event logging and analysis tool was embedded in Ämppäri user interface to allow further usability testing. A set of simple rules were developed to analyze whether a user has completed the tasks given and what types of errors she has possibly made when performing the tasks.

Research questions

Automated user interface event logging was embedded into the Ämppäri user interface to collect information about user actions during usability tests. In addition, an automated log analyzing system will be developed. The system effectiveness in finding and classifying erroneous events will be evaluated. The research questions related to the logging system are:

1. Can a correct event sequence representing an ideal, error-free way to complete a task can be found in the event log,
2. Is it possible to classify the events that are not in this correct event sequence based on some pre-defined rules and
3. What is the event data that should be logged in order to get positive results on the two previous questions?

User tasks in the experimentation

Three Ämppäri related tasks were selected to gather user interface event logs used as sample data for evaluating analysis system. A correct event sequence was defined for each task. The logs were gathered when the user carried out the tasks (example in figure 2). The correct expected sequences are matched against the logs. The event sequence is found in the log file, if all the events from the correct sequence are present, and they are performed in the same order as in the sequence file.

```
<entry>
  <time>14</time>
  <event>
    <action>Click</action>
    <dialog>ÄmppäriUIDlg</dialog>
    <control>
      <name>Devices</name>
      <type>MenuItem</type>
      <repeatable>false</repeatable>
    </control>
  </event>
</entry>
```

Figure 2: Example log

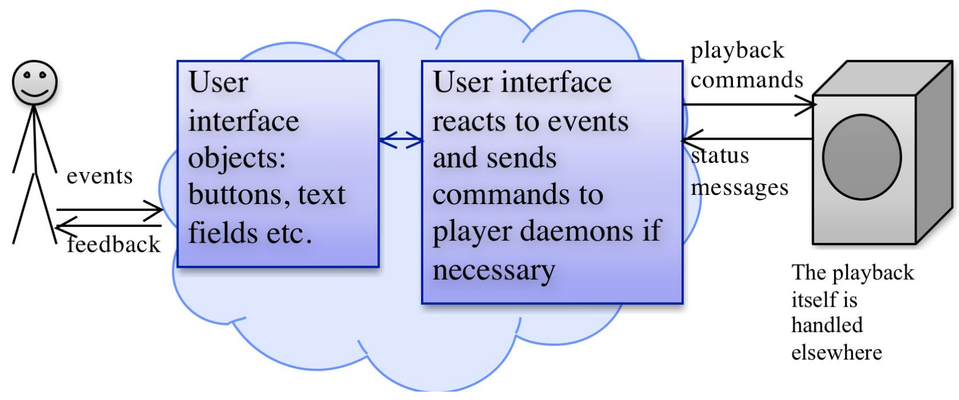


Figure 1: Logging system

If there are several correct sequences for doing a task, the task is considered complete if at least one of the correct sequences was found. All log events that do not match the sequence file are considered possibly erroneous and will be analyzed further.

Error classification

The events that did not match the correct event sequence are compared to the error classification rules in order to match them to the pre-defined error types. The classification rules are defined as follows:

1. **An event is repeated** one or several times.
2. **A correct event sequence or part of it is repeated** one or several times.
3. **An event is close, yet not fully identical, to the correct one**.
4. In the next event after the event in question, the **dialog is changed incorrectly**.
5. **Several events with same control type** occur in a row.

Evaluation results

It was confirmed that with the currently defined correct sequences the analyzer will not give false positive or false negative match with the log.
For evaluation purposes, also erroneous user interface log sequences were generated. When the analyzer was given an erroneous log sequence as the input, all erroneous events were recognized correctly using the error classification rules. The errors matching the rules were classified correctly, and no errors that should not match any of the rules were classified. Classification of erroneous events worked correctly also for event logs containing several error types.

Conclusions

The experiment results show that a correct user interface event sequence can be found in the user interface event log, when it is present in the log, and its absence shown, if it is not present. Moreover, the results show that errors can be classified based on predefined rules. The actual content of the logs should be designed according to the purposes of the analysis, and should be independent of the user interface and how it is implemented to allow using the same analysis tool even if the user interface is changed, and to allow testing different alternative user interfaces.

The most important result was to see the additional benefits of the design and development of the analysis tool for the user interface design process itself. In fact, the work needed to design the analysis tool brings an additional evaluation phase to the user interface early in the development process. The potential user errors must be considered when testing the logging system. This forces the developer to pay special attention to the possibility of a human error – which, again, is a general design guideline and should be kept in mind all the time, but in this case constructive work is required and the possible errors cannot be bypassed.

Implementation: Secure Bluetooth authetication

Ossi Rautiainen, Eetu Pilli-Sihvola, Juha Loukkola
Keywords: Bluetooth, authentication, GINA, secure logon

Introduction

➢ In the project, a Bluetooth authentication solution was implemented that aimed to improve on the security of two existing commercial Bluetooth authentication solutions that were proven vulnerable to the Bluetooth device address spoofing attack.
➢ The scenario that the implementation focused on is that of a user in possession of a Bluetooth-enabled mobile phone who wants to be able to log on to a Windows computer without having to type in his username and password.
➢ The motivation behind the implementation was to make the log-on process more secure and faster than was the case with the existing solutions.
➢ For the mobile host (a Bluetooth-enabled mobile phone), a Symbian service was developed to handle the connection on the mobile end.
➢ Correspondingly, a replacement for the Graphical Identification and Authentication dynamic-link library (GINA DLL) used in the Windows logon process was implemented for the Windows host.

Mobile host software

▪ Software for the mobile host consists of a service that runs in the background and provides the user with a simple graphical user interface for starting and stopping the service and viewing the logged activities.
▪ The service was implemented in Symbian S60 3rd Edition. Symbian CryptoAPI was used for implementing the authentication functions.
▪ The service first opens a listening socket on a free channel, and then starts advertising itself on that channel by adding an entry into the service discovery database.
▪ When the mobile host is within Bluetooth range, the Windows host connects to the service and is authenticated using a signed certificate, and key exchange according to the Diffie-Hellman algorithm is performed. Finally, the credentials are encrypted and sent to the Windows host.

Windows host software

▪ The software contains a Dynamic-Link Library (DLL) that replaces the original Windows XP GINA DLL file.
▪ A setting in the Windows registry tells Windows which GINA DLL it should use during the logon process. By changing this setting it is possible to replace the original GINA with a customized version, as was done in this project. The customized version can utilize the functionality of the original GINA to an extent of its choosing or bypass it completely if necessary.
▪ For the Bluetooth authentication mechanism, a private and public key pair is created for the Windows host, and the private key along with the certificate containing the public key signed by the Certificate Authority (CA) is placed in the location specified in the configuration file.
▪ The implementation of the GINA replacement DLL also contains a polling utility that inquires at specified intervals whether any Bluetooth devices with approved credentials are in the proximity. If such a Bluetooth device is found, the credentials are supplied back to the customized GINA replacement that takes care of authenticating the user to the Windows host.

Encountered issues

▪ The Symbian documentation seemed lacking at times, and many issues were resolved by consulting developer forums instead of the official documentation.
▪ One major issue was the co-operation of the Crypto++ library and the Symbian CryptoAPI. There were a lot of problems getting them to work together in the authentication process. It turned out that when verifying SHA-1 hash codes generated and signed by the Windows host, it is necessary to append the prefix ***\x30\x21\x30\x09\x06\x05\x2B\x0E\x03\x02\x1A\x05\x00\x04\x14*** to the beginning of each SHA-1 hash code generated on the mobile host before verifying the hash code against the signed one.

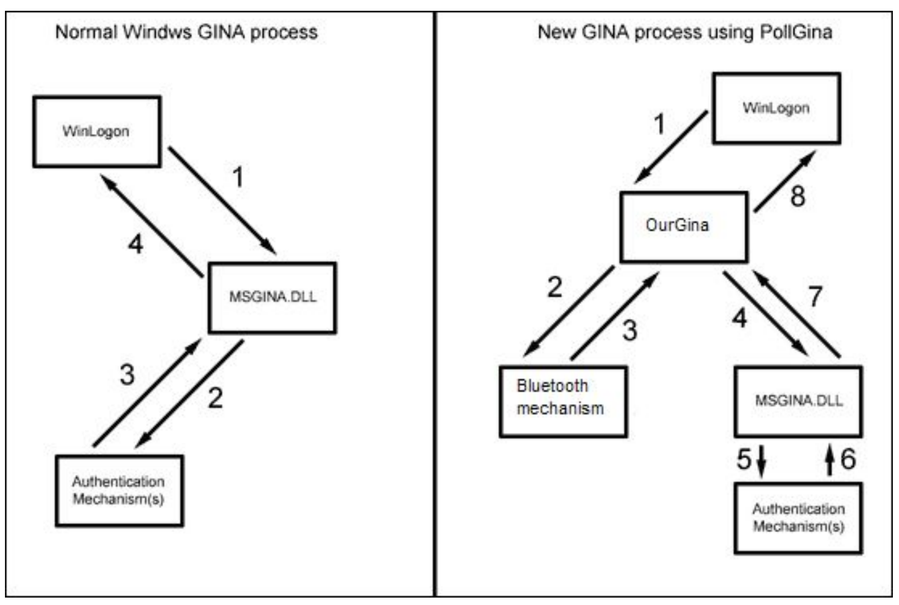


Figure 1. Replacing the GINA DLL using a customized version to enable logging on using Bluetooth.

Future development possibilities

▪ *Improved inquiry process*. As it stands, the discovery of devices is pretty slow for practical purposes, and it is even more so when multiple Bluetooth-enabled devices are within range of the Windows host.
▪ *Click-and-authenticate feature*. The current implementation automatically logs the user in to the Windows host when his Bluetooth device is discovered, and so situations could arise where information inconvenient to the user is displayed unwittingly. An improvement to this would be a feature that logs the user in only after he clicks a button or a link on the screen and if he is authorized to use the host in question.
▪ *Adjustable detection range*. The implementation could be configured to detect Bluetooth devices only within a user-defined range. This would improve usability as unwanted logons could be avoided.
▪ *Password and token*. By requiring the use of both a password and a token in order to log on additional security can be provided.
▪ *Mobile phone as a key*. This is a larger avenue of development that could prove fruitful. The user should be able to use his Bluetooth-enabled mobile phone as key to enter places in the workplace, for example.

Conclusions

✓ The implementation consists of a Symbian service running on a mobile host, and a replacement dynamic-link library for the GINA, used in the Windows logon process, for the Windows host.
✓ The project implementation is resistant to the Bluetooth device address spoofing attack as the address is not used to authenticate the devices.
✓ An authentication scheme using a USB flash drive instead of a Bluetooth-enabled mobile host was also developed.
✓ The question of feasibility of the implemented solution on a larger scale is dependent upon the speed of the authentication process. If logging on to a computer takes more than a couple of seconds, a user would much rather just enter his credentials than wait for the automatic authentication.

Email: pmrg@tkk.fi
WWW pages: http://www.cs.hut.fi/~pmrg
Telephone: +358 9 451 4807
Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund

Experimentation: Measuring QoS parameters to determine media transport capabilities

Tatu Kilappa, Jaakko Salo

Keywords: streaming, QoS, network probing

Introduction

It can be desirable to be able to make conclusions on the feasibility of transferring desired media types over an available network link in realtime. However, the simple classification of the network interface rarely, if ever, offers sufficient information to determine its real-world capabilities.

Thus the objective of this experimentation is to determine whether or not a link is feasible for the transmission of certain distinct types of media. The results are rough classification of media transport capability instead of numeric representations of QoS values. All requirements are assumed realtime for use in e.g. medical or surveillance applications. The following media types are used in the experiment:

- Video stream, H264, HD quality, ~25Mbps
- Raw numerical measurement data, ~1Mbps
- Numerical measurement data packed using lossy compression, ~256kBps

Hypothesis

Using a few simple measurements on the network, it is possible to determine what kind of media can be transferred over the current link.

Description of methods

The expriment uses three distinct network probes to gather data. All of the probes require a network tester program running on both connection endpoints.

1. Data collection. The network tester software intiates several connections to the data recipient to gather data about the end-to-end link. They are described below in sections 1.1, to 1.3.

1.1. Collection of data transmission records using unreliable connection, active replies. The tester opens an UDP connection to the end host and sends a stream of maximum MTU size packets containing an index. The data contained in the packets is random – this will prevent any effective lossless compression from taking place in lower network layers. The entropy in real media formats is typically also very high, so this reflects a real situation well. The end host replies with an acknowledgement packet for every received packet. The initiator stores the timestamp and index of each sent packet and each acknowledgement. The test is repeated for several commonly used transmission speeds (64kbps, 256kbps, 1Mbps, 10Mbps, 100Mbps) and ran for 10 seconds.

1.2. Collection of data transmission records using unreliable connection, no replies. As in 1.1, but instead of replying, the server stores the timestamp of each received packet and when they stop arriving, opens a reliable connection to the initiating host and sends a full record.

Methods (cont.)

1.3. Collection of data transmission records using reliable connection. The tester opens a TCP connection to the end host and sends a stream of data. The end host records timestamps of every new read that can be done, along with the total amount of data currently received.

2. *Data processing.* The transmission records are Processed in Matlab to produce throughput (average bytes of data transferred in regard to time), packet loss (number of packets lost as a % of total transmitted packets) and jitter (variance of the time to next packet or next read segment).

3. *Data analysis.* Metrics produced by phase 4 are compared against the predefined requirements set by the media types presented earlier.

Focus

The distinct network types are not taken into account – only the end-to-end –capabilities.

Phase	Methods	Result
Data collection	1. Test connections to data recipient	Traffic records of packet index vs. timestamp or sent data vs. timestamp.
Data processing	2. Extracting data from the traffic records.	Throughput, packet loss rate and jitter for every connection
Data analysis	3. Matching of the processing results against the predefined media requirements	Information on fitness to transfer certain media types.

Table 1: Summary of methods

Experimentation setting

An essential part of the experimentation setting is the network in which the experimentation is conducted, ie. through which the traffic of the test software is passed. Four different networks were considered: a public ADSL ISP, public 3G ISP, laboratory Fast Ethernet switched network and laboratory 802.11g WLAN network. A small network testing software needs to be implemented for conducting the experiment. The network testing software and host configuration need to be carefully crafted so that they do not produce unintended delays affecting the test results, for example by bad program design so that the network stack is not powerfully utilized. A part of the software runs in two separate hosts in the selected network. For each network selected, the test is run individually, thus the results produced reflect the capabilities of that individual network.

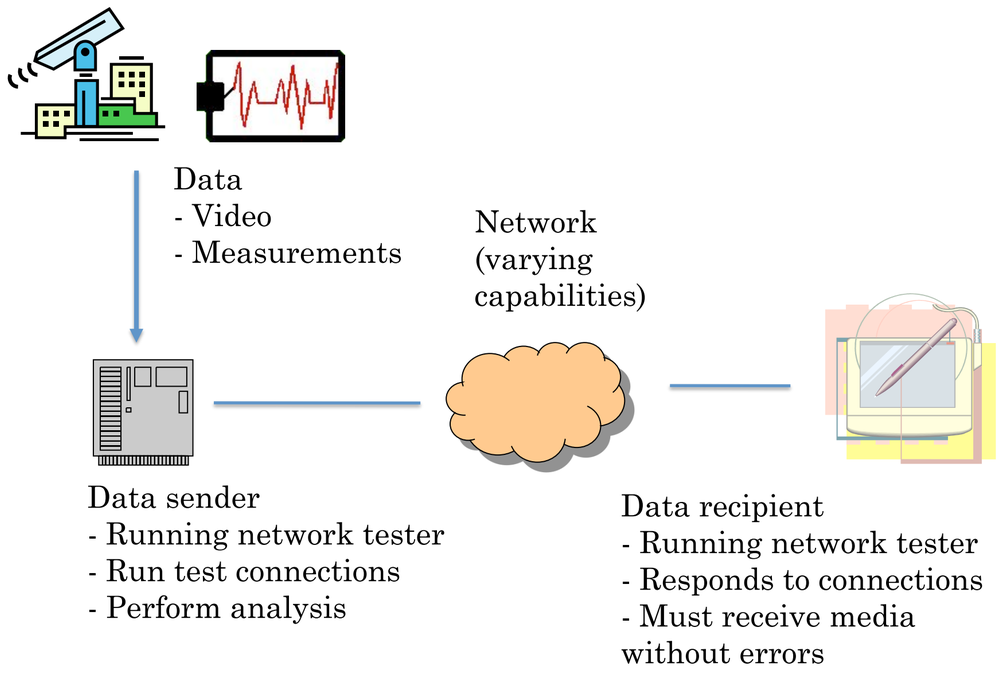


Figure 1: Experimentation setting

Experimentation: Applying speech recognition software to wearable accelerometer data

Mika Iivonen, Timo Helenius

Keywords: accelerometers, speech recognition, signal processing, sphinx

Introduction

The experiment aimed to research an innovation prototype of a real-time system, which can read a continuous accelerometer signal and interpret discrete symbols from it via open-source speech recognition software, Sphinx.

While sign languages differ from oral languages, they both serve the same purpose: to enable communication between humans. Interesting in the experiment was researching the possibility to utilize and adapt a set of speech recognition tools and methods for interpretation of simple sign languages.

Key concepts

Acceleration sensor (Accelerometer) Device which detects acceleration, or changes in speed
Acoustic model Defines sound patterns for recognition
Sign language Language utilizing visual information instead of acoustic
Speech recognition A way to interpret spoken language with computer by analysing continuous signal and recognising symbols from it.

Hypothesis

The experimentation will show that, when properly adapted, a speech recognition program is capable of interpreting signals other than an acoustic signal originating from human speech.

Description of stages

1.In **Sign definition stage** the test set of signs is defined.

2.In **Data recording phase** training data is collected to adapt speech recognition system for a new language.

3.In **Sound modulation phase** the information contained in the input signal is embedded into sound waves.

4.In **Speech recognition training phase** the generated audio is used to train a new acoustic model. That is, the speech recognition system is adapted to a new virtual speaker speaking some previously unknown language.

5.In **Recognizing phase** the new acoustic model can be used to recognize signs defined in the language by modulating the sound from the gestures on-the-fly.

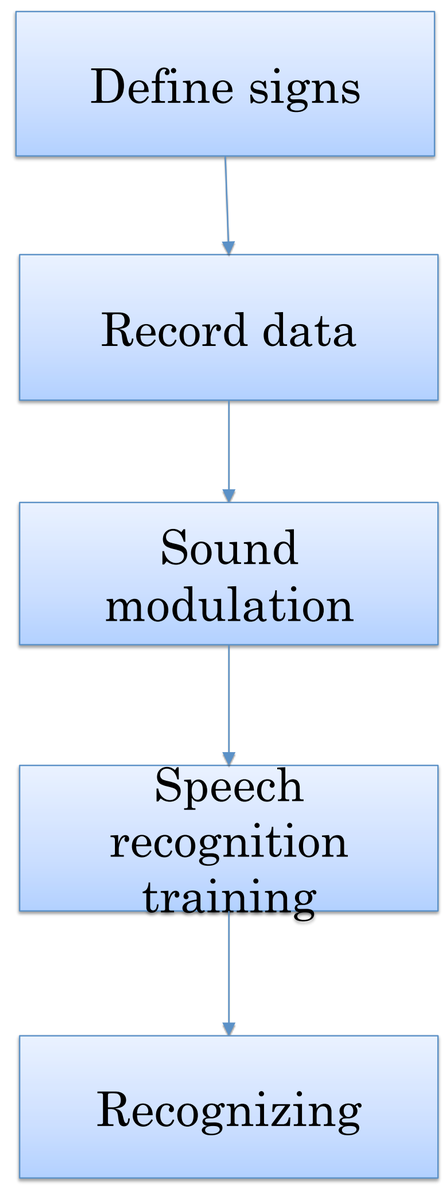


Figure 1: Stages of the experimentation

Experimentation setting

1. Accelerometer(s), preferably three-dimensional.
2. Data recorder software easing the task of saving a huge amount of accelerometer data from the performed signs.
3. Sound modulator software for converting input signals to audio.
4. Speech recognition software with support for training new languages, e.g. open-source Sphinx.

Conclusions

It was confirmed that suitably pre-processed accelerometer signals amplitude-modulated in base waves produces audio from which the performed signs are easily distinguished by humans – i.e. the information is not completely lost in the process.

Much of the signal processing in a speech recognition system is tuned to enhance results especially with human voice. This makes it hard to use speech recognition tool as a generic pattern recognition tool. The underlying technology in speech recognition systems is probably suitable for recognizing signs, but for real applications for sign language recognition modifications are needed to the software.

Email: pmrg@tkk.fi
WWW pages: http://www.cs.hut.fi/~pmrg
Telephone: +358 9 451 4807

Head of the research group: Mervi Ranta
Coordinator: Henrik J. Asplund