Secure Two-Way Transfer of Measurement Data*

Marko Hassinen, Maija Marttila-Kontio, Mikko Saesmaa, and Heli Tervo
(Corresponding author: Marko Hassinen)

Department of Computer Science, University of Kuopio
PO Box 1627, FIN-70211 Kuopio, Finland (Email: Marko.Hassinen@uku.fi)

(Selected paper from ITNG 2006)

Abstract

We introduce a measurement system architecture, which has three main qualities: secure two-way transfer of measurement data, quick adaptability to different kinds of measurement tasks and strong data presentation capabilities through XML techniques. In the architecture, we take advantage of well-tried technologies, like a commonly used visual programming language that offers predefined and adaptive measuring tools for managing measurement devices and tasks. XML is a widely adopted standard for a flexible text format and data exchange. It brings along a vast selection of ready made facilities for processing and transforming the content into any format desired. We also propose a secure environment into the architecture, which can be accessed on demand using a wide range of terminal devices.

Keywords: Data transfer, measurement system, secure, two-way, XML

1 Introduction

In this paper, we introduce a measurement system which has unique qualities like two-way transfer of measurement data, quick adaptability to different kinds of measurement tasks, and strong data presentation capabilities using XML (Extensible Markup Language) techniques. The system is not limited to any particular context, that is, the system is general enough to cope with various measurement tasks.

Consider, for example, the field of health care, where measurement typically involves professionals, who use measuring devices. Then, the data flows in one-way from patient to the professional. We have devised a system which uses common state-of-the-art equipment for the benefit of both parties. For this we define a two-way transfer of measurement data. For example, a patient might be given a blood pressure measurement device with a PDA (Personal Digital Assistant) to be taken home and used according to some predefined schedule. In addition to sending the measurement data, the PDA can receive data to advice or alert the patient on some issue. This two-way communication of measurement data brings true interaction between the parties. As mentioned earlier, the measurement system is general in design and can therefore also be used in various other scenarios, in entirely different contexts.

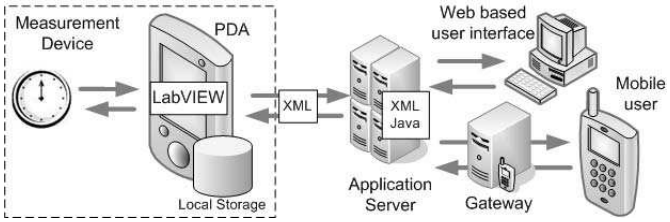

Figure 1: Overview of the measurement system architecture

The rest of the paper is organised as follows. Section 1 describes the measurement system as a whole. The following sections concentrate on the various parts of the system starting with building the measurement application by using LabVIEW in Section 3. Section 4 discusses data processing and formatting, which plays an important role as to how the data is handled in the system and presented in various devices. Section 5 discusses security issues involved in transferring the measurement data between parties. Finally, we give our conclusion in Section 6.

2 System Overview

The architecture is presented in Figure 1 and consists of measurement equipment, an application server, and end user devices.

The measurement equipment consists of physical measurement devices, such as sensors and a PDA device. Visual programming language LabVIEW is used for implementing the measurement application. We use LabVIEW for building the application, because it provides a wide

*A preliminary version of this work appeared in proceedings of international conference on Information Technology: New Generations (ITNG 2006).

range of predefined measurement tools.

The Application Server contains a database used for permanent storage of measurement data. Communication between the PDA and the Application Server is bidirectional. The information is transmitted in XML format. This allows us to accommodate possible third party implementations in either end. The main task of the Application Server is to process the measurement data and provide the end user devices with content in suitable format. The end user devices range from stationary third party information systems to mobile hand held devices, and can communicate with the Application Server using TCP/IP based protocols, such as HTML, WML or XML-RPC.

Presented architecture is general and can be used in several areas, where secure two-way transfer of measurement data is needed. The strength of the architecture is easy modification for different requirements. By changing only the sensors, the system can be used for a different purpose with slight modifications to the measurement application.

Figure 2: The user interface of a flood measurement system consists of virtual instruments

3 Building Measurement Application by Using LabVIEW

Measurement equipment consists of physical devices and measurement applications on a chosen platform. Data acquisition from a physical phenomena into an application is handled with sensors, a signal conditioning device, and a data acquisition (DAQ) card. Sensors and signal conditioning can be combined together into a simple measurement device. Required physical devices depend on the phenomenon to be measured. In addition, used devices, especially the application platform, are strongly determined by a measurement environment and measured phenomenon. A wireless PDA platform offers an opportunity for building a hand held measurement system [1].

A PDA is compact and costs less than a laptop. In addition, it is practical for simple measuring processes and it is easy to use with a help of a touch panel that improves usability (compared, for example, to a minimal keyboard). On the other hand, PDA's confined capacity, such as available memory, view, processor power, and energy management has to be taken into account [2, 3, 4]. Therefore, an implementation of a measurement application can be made easier by choosing a proper programming language.

The measurement application we present in this article is created by using visual programming language LabVIEW [5, 6, 7]. LabVIEW is commonly used in information technology, for example, in data acquisition, presentation, and analysis, and also in control [8]. LabVIEW offers predefined and adaptive measuring tools for managing a PDA and measurement devices.

LabVIEW's visual language is based on visual components called virtual instruments (shortly VIs) and their possible sub-VIs. The user interface is implemented with virtual controls (such as slides, knobs and meters) and indicators (such as gauges, bars, graphs and charts). In Figure 2, a user interface of a water level controlling application for a flood situation is illustrated. The user interface consists of two indicators and seven controls. All instruments placed into the UI also appear in the program code. Visual program code is based on data flow principle, whereas conventional programming languages, such as C, C++, and Java, are based on control flow principle [9, 10]. In visual programming language, data flows without unnecessary control from one VI to another through wires connected to their input/output terminals, see Figure 3. With the help of LabVIEW's PDA Module, the program code and UI are implemented considering requirements and constraints of the PDA. After an application is implemented, it is compiled and transferred into a PDA by LabVIEW. All modifications of the application are done in PC, laptop or other device capable of supporting the LabVIEW application development environment.

In Figure 4, a measurement system is illustrated. It consists of physical and virtual components. The data (signal) is gathered with sensors, then modified by signal conditioning and transferred into the PDA through a DAQ interface or a USB port. Gathered data can be saved into a local database, a file, or it can be transferred into a remote database by using selected communication (IrDa or Bluetooth).

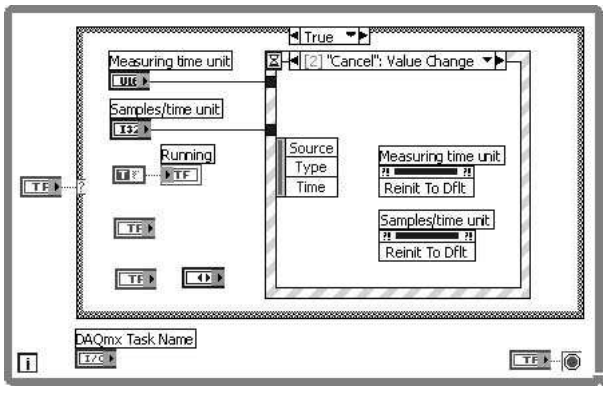

Figure 3: An example of visual program code

UI modification with LabVIEW is easy because of PDA tools and instruments. If the measurement application is

used by people having different user capabilities, a user interface has to be adaptive. For example, users of a blood pressure measurement system could have debilitated eye vision, low hand or finger coordination, or other immobility. In this case, the measurement application has to be so simple that the user could handle a measurement session by pressing only a few buttons. With a help of modifiable virtual controls and indicators, there is an opportunity, without major work, to customize even an individual user interface for each user.

Compared to conventional text based programming languages, the visual code is simpler to create and understand [9, 10, 11]. Programmer's work is getting easier and less error prone because of predefined instruments and predefined functionality.

4 Data Processing and Formatting

XML [12] is a flexible text format designed for electronic publishing, and for the exchange of wide variety of data on the Web and elsewhere. XML is developed by a non-commercial consortium W3C (The World Wide Web Consortium) [13], and techniques and languages generated by W3C can be compared to world wide industrial standards. XML techniques support interoperability of data processing; data transmission between independent services, data transformation, updating and formatting, and distribution of data in different formats. For these reasons, the use of XML is spreading all over the world among different organisations to improve their collaboration. Also, end-users on the web meet XML more often every day.

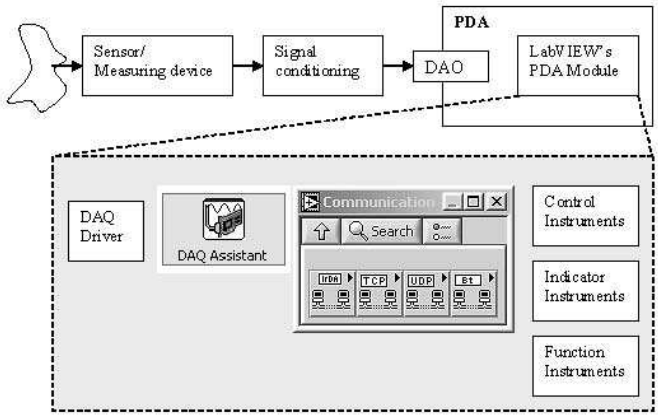

Figure 4: Components of the measurement system architecture

There exists a huge amount of application standards based on these W3C standards, for example HL7 [14], which contains a group of standards for health care, or Finvoice [15], which is a standard for electronic invoices used by Finnish banks. In practice, the use of application standards may slightly differ from organisation to organisation. However, many standards used in real life are data and system specific. In many cases, when transmitting data to other systems, we need data transformations. Using a standard like XML (or a standard based on XML), in general, makes this easier than with a totally unfamiliar form of data.

In the architecture of the secure two-way transfer of measurement data, the receiving end, that is the Application Server, will handle all data transformations, see Figure 1. In order to make the architecture generic, we need to resolve the issue of using new standards or general technologies — one unsolved stumbling block in using XML is the formatting of data in a general level.

Formatting XML data, or presenting it, could be quite tricky. End-users need to see the data in some reasonable form and not in pure XML form with tags. The most used languages for formatting XML data are CSS (Cascading Style Sheets) [16] and XSL (Extensible Stylesheet Language) [17]. CSS is designed for web display, and XSL is a powerful styling language with a strong support for print page. However, there are two major disadvantages in formatting data with these languages, from the point of view of end-users: the formatting task is difficult, and the result is form specific. Therefore, there is a need for implementing multiple formatting scripts for multiple result formats. For example, the user may need to handle a lot of different data coming from separate systems, and data with similar information may look very different when coming from disparate systems. The work would become easier, if the views of different data could be formatted to look alike, without changing the actual data. A formatting script for each incoming data source can be defined with XSL, for example, but using XSL appropriately needs not only quite deep knowledge of processing structured documents but also programming skills. Also, the user may need to deliver data to multiple result forms: for databases, paper printouts, and web pages. This could be done with XSL, but a style sheet for every result form must be defined separately. This would become easier if there was a possibility to define a uniform layout for data in one script without taking into account forms of results, see Figures 5 and 6.

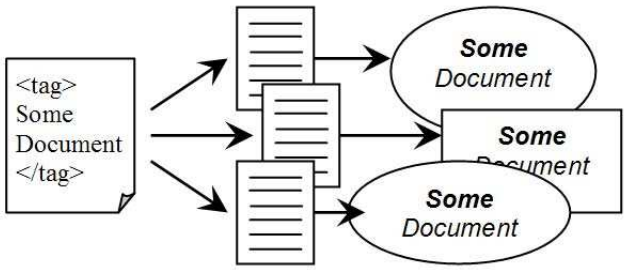

Figure 5: Describing formatting of a document for multiple result formats with existing systems

In these circumstances, there is a need for an easy-to-

use generic styling language for defining uniform layouts for documents. We are developing a language, or more like an interface to XML formatting languages, that is to be a high-level solution to the problem of defining XML document formatting. Because the XSL language is a powerful styling language, there is no reason to try to override the actual formatting language. The general interface for describing the XML formatting makes use of existing languages and techniques, thereby not being a competing language or technique.

The formatting language we are developing contains only commonly used and most typical styling features (font properties, simple table properties, etc.), which makes it easy to write and read. The formatting description is simply a readable definition of the document layout in XML format. XML syntax also makes it possible to process both the source document and the formatting specification using common XML tools. Most importantly, the language is independent of any specific style sheet language implementation or form of the result document; the formatting description is common both to an electronic document and to a paper printout.

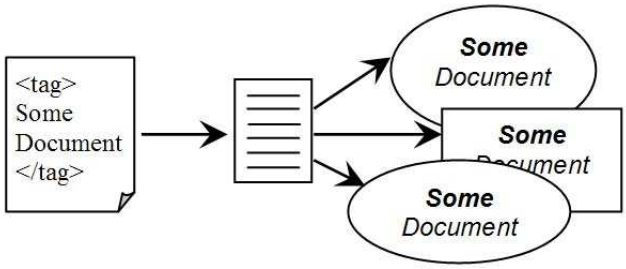

Figure 6: Describing formatting of a document for multiple result formats with one style sheet

In our architecture, all the data handling is done in the Application Server. Also forwarding any data takes place through this part. When a user has an implementation of the generic formatting language introduced above, data processing is possible without specialized skills (for instance, programming) or specialized commercial software. A manipulation of data could be done with freely available XML techniques and presenting data is easy for end-users through this generic formatting layer, taking advantage of most used XML techniques.

5 Privacy, Integrity and Authentication

Strong mutual authentication and access control are crucial between the Application Server and the PDA connected to the measurement device, see Figure 7. Otherwise, any party in the web would be able to send random measurement values. The same applies also between the Application Server and the web client. We rely on SSL (Secure Sockets Layer) [18] to provide authentication in these situations.

The SSL is a group of protocols designed to provide end-to-end security for TCP traffic. It can provide privacy, integrity of data, and authentication of communicating parties [18]. The SSL protocol can be configured to provide mutual authentication. Our web client, see Figure 7, uses HTTPS (HTTP over SSL) and a certificate to authenticate itself to the server. Furthermore, it is possible for an application to get the X509 certificate of the opposite party from the SSL protocol [19, 20, 21].

The Authentication and Authorization Server can authenticate a user based on a signature the user has generated. This scheme is necessary, when the device is not capable of using SSL, for example, when the traffic is in form of SMS messages.

5.1 Certificate Authority and Timestamp Service

In order to achieve mutual authentication using SSL, both parties must have certificates that the opposite party can verify. We are using a CA (Certificate Authority), which generates pairs of private and public keys for users and signs corresponding certificates. The keys are generated and stored on a tamper resistant smart card. This makes it practically impossible to make duplicates of the card [22]. Signatures using the private key are done on the card using the card's processor. All privileged operations on the card are protected with a PIN.

The CA also maintains a CRL (Certificate Revocation List), which is used to revoke a certificate, in case a private key corresponding to a certificate becomes compromised. All the certificates signed by the CA can be found in an LDAP (Lightweight Directory Access Protocol) directory.

5.2 Role-based Access Control

Access control in our web client employs Role Based Access Control [23]. This means that it is possible to define different roles, such as an administrator or a user, which have different privileges in handling the data. Our system employs an Authentication and Authorization Server, which is responsible for authorizing user actions based on the role the user has. Authentication of the user is done using the certificate obtained from the SSL protocol. The database server has knowledge of all legitimate users and their roles. The identity of the user is compared to this database and privileges are granted according to the roles found in the database.

5.3 SMS

SMS (Short Message Service) is probably the most used data service in the GSM (Global System for Mobile Communications). SMS messages have been used in several systems, where a person needs to be alerted of some event.

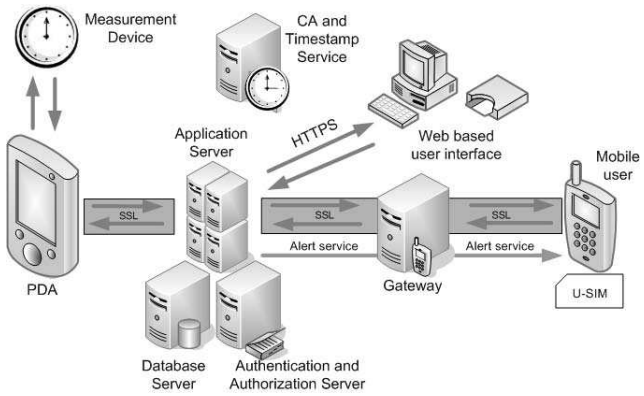

Figure 7: Overview of the security architecture of the system

Usually a monitoring process sends an SMS message to a human, when something happens that needs attention. Examples of such applications include home security systems, industrial processes and alike. SMS messaging has also been used in reminder systems, where the GSM user needs to be reminded of something, for example a doctor's appointment.

In our system, SMS messaging is utilized by the Application Server. Any measurement value exceeding some threshold value will generate an event which sends an SMS message. In case of a health monitoring equipment, the recipient could be a relative or a health care professional.

5.4 Privacy, Integrity and Authentication in SMS traffic

Access to the GSM network is accomplished using an SMS gateway, provided by a GSM operator. Our Application Server sends SMS messages to the gateway, which in turn delivers them to recipients. This scheme, however, has some security concerns, which have to be addressed. Since all unprotected data in the internet can be read by outsiders, we have to consider how to protect the data we transmit. The Application Server – SMS gateway contact can be protected with SSL. However, as the message is received by the operator, it will be stored until it can be delivered. Sometimes a message can wait for delivery a considerably long time, when, for example, the targeted mobile device is turned off.

For SMS messaging, the SSL handshake protocol is much too complicated. To guarantee privacy and integrity of the SMS traffic we use public key cryptography for encrypting and signing the data. It was shown in [24] that privacy, integrity and strong authentication can be achieved using the J2ME (Java 2 Micro Edition) [25] platform on a mobile phone. The Wireless Messaging API [26] and the Security and Trust Services API [27] are required on the device.

When sending a message m, the Application Server encrypts it with the public key KU_M of the intended recipient M. This way the message can only be decrypted by the recipient with the correct private key. The Application Server also signs a hash value $h(m)$ of the message m with the private key KR_{AS}. By verifying the signature, the client can be sure that the message came from the Application Server and that the message has not been altered in any way after it was signed. To ensure the freshness of the message, we use a dual signature, where the Application Server sends the hash of the message $h(m)$ to the Timestamp Service. The Timestamp Service calculates a xor ($\oplus$) of a timestamp, the hash, and a random nonce n, and signs the result. This signature $E_{KR_{TS}}(h(m) \oplus n \oplus ts)$ with the timestamp and the nonce are then sent to the Application Server which forwards them with the message to the client. Hence, the Application Server sends M the message $E_{KU_M}(m, ts, n, E_{KR_{TS}}(h(m) \oplus n \oplus ts))$. The nonce n is used because otherwise the Timestamp Service would have to use it's private key to encrypt any chosen data sent by the Application Server.

The responsibility of determining the freshness of the timestamp is left to the recipient. The client may have some predefined time window for accepting timestamps.

5.5 Two-way SMS Traffic

Achieving authentication and privacy for SMS messages sent by the application server is rather straightforward. The situation with messages sent by the device to the server is somewhat more complicated. Considering that we are using the SMS system to notify a user of an event which might be of urgent nature, it seems natural that the mobile user should be able to react on the message. Using the mobile device the user could log into the system using WTLS [28] and respond to an urgent message.

When we use SMS messaging for response, the authentication solution is based on a challenge-response protocol. The alert message contains a random nonce n, which is encrypted using the public key KU_M of the mobile user M. The user can subsequently be authenticated with the signature $E_{KR_M}(n \oplus h(m))$. As stated earlier, the Authentication and Authorization Server can verify this signature. The content of the response message M is encrypted using the public key KU_A of the Application Server.

The Application Server replies to each of these messages with a confirmation. In case the message contained a request of some operation, the confirmation will contain a return value which indicates the result of the operation. The confirmation message contains a new random nonce which can be used for subsequent communication. The system is protected against replay attacks by a demand that each nonce n can be used only once by the user M. However, this would not protect the system for a delayed message attack. In such an attack, the adversary has control over the transport media and is able to block and record a message for sending it at a later time. Whether this attack is relevant, depends on the specific use of the system.

There are some situations, where the client may need an authenticated timestamp. These situations include protection against the delay attack, and the case, where the communication is initiated by the client. The client M will ask the Timestamp Service TS, see Figure 7, for a timestamp ts and for message m, and sends the message as follows:

1) $M \rightarrow TS : E_{KU_{TS}}(ID_M, h(m))$.

2) $TS \rightarrow M : E_{KU_M}(ts, n, E_{KR_{TS}}(h(ID_M \oplus h(m) \oplus ts \oplus n)))$.

3) $M \rightarrow AS : E_{KU_{AS}}(ID_M, E_{KR_M}(h(m)), m, ts, n, E_{KR_{TS}}(h(ID_M \oplus h(m) \oplus ts \oplus n)))$.

The Timestamp Service uses a random nonce n to protect its private key. After receiving the message, AS can verify that the timestamp ts is within a predefined time window and greater than the timestamp of the previous communication.

6 Conclusion

In this paper we presented a system architecture for secure two-way transfer of measurement data. Using a PDA costs less than a laptop for the client platform for the measurement data. On the other hand, PDA has an environment with far more possibilities than a mobile phone would have. Also, a PDA is compact and lightweight solution which is easy to move around. Being a wireless device that can connect to a measurement device wireless makes it optimal for our purposes. One of our criteria for choosing a PDA was the easy implementation of measurement software with LabView.

Our choice of data structure for data exchange is XML. Although, some other scheme would require less control information in respect to the payload data, other benefits of using XML clearly overcome the effects of the required additional space for the tagging information. Being a widely adopted standard, XML gives us excellent possibilities for defining interfaces for third party implementations. Choice of XML also brings along a vast selection of ready made facilities for processing and transforming the content into any format desired.

Strong mutual authentication is a crucial requirement for any data system, where access to services is restricted. In order to handle large amount of users and user privileges, we have adopted the role based authentication. Using roles, handling of access privileges becomes much easier and less error prone. With it, we propose a secure environment which can be accessed on demand using a wide range of terminal devices.

References

[1] B. A. Myers, J. Nichols, J. O. Wobbrock, and R. C. Miller, "Taking handheld devices to the next level," *Computer*, vol. 37, no. 12, pp. 40-51, 2004.

[2] M. A. Viredaz, L. S. Brakmo, and W. R. Hamburgen, "Features: Energy management on handheld devices," *Queue*, vol. 1, no. 7, pp. 44-52, 2003.

[3] L. Zhong and N. K. Jha, "Power and energy: Graphical user interface energy characterization for handheld computers," in *Proceedings of the 2003 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, pp. 232-242, 2003.

[4] P. Yu and H. Yu, "Lessons learned from the practice of mobile health application development," in *Proceedings of the 28th Annual International Computer Software and Applications Conference*, vol. 2, pp. 58-59, 2004.

[5] National Instrument's LabVIEW. (http://www.ni.com/labview)

[6] K. Bisking, *Building a PDA-Based Measurement System*, National Instrument. (http://www.ni.com)

[7] National Instruments, *Develop of Wireless Measurement System*. (http://www.ni.com)

[8] N. K. Swain, J. A. Anderson, A. Singh, M. Swain, M. Fulton, J. Garrett, and O. Tucker, "Remote data acquisition, control and analysis using LabVIEW fron panel and real time engine," in *Proceedings of the IEEE Southeastcon 2003*, pp. 1-6, 2003.

[9] B. A. Myers, "Visual programming, programming by example, and program visualization: A taxonomy, ACM SIGCHI bulletin," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 59-66, 1986.

[10] G. G. Roy and J. Kelso, "Craig Standing, Towards a visual programming environment for software development," *Proceedings of the International Conference on Education and Practice*, pp. 381-388, 1998.

[11] M. Hirakawa and T. Ichikawa, "Advance in visual programming, Systems Integration," in *Proceedings of the Second International Conference on Systems Integration*, pp. 538-543, 1992.

[12] T. Bray, et al. (eds.), *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation, Feb. 2004. (http://www.w3.org/ TR/REC-xml)

[13] World Wide Web Consortium (W3C). (http://www.w3.org)

[14] Health Level Seven (HL7). (http://www.hl7.org/)

[15] The Finnish Bankers Association: Finvoice. (http://www.fba.fi/finvoice/)

[16] B. Bos, et al (eds.), *Cascading Style Sheets*, level 2, W3C Recommendation, May 1998. (http://www.w3.org/TR/CSS2)

[17] S. Adler, et al, *Extensible Stylesheet Language (XSL) Version 1.0*, W3C Recommendation, Oct. 2001. (http://www.w3.org/TR/xsl)

[18] W. Stallings, *Cryptography and Network Security, Principles and Practice*, Prentice-Hall, 1999.

[19] C. Trick and C. Turfus, *Secure sockets on Symbian OS v6.x Symbian Developer Library*, May 2002. (http://www.symbian.com/developer/techlib/papers /sslapi/sslapi.html)

[20] E. Rescorla, *HTTP Over TLS, Request for Comments: 2818*, Network Working Group, May 2002. (http://www.ietf.org/rfc/rfc2818.txt)
[21] Sun Microsystems Inc, *J2ME API Documentation*. (http://java.sun.com/)
[22] Gemplus About Smart Cards, *Related Technologies Gemplus 2005*, Apr 2005. (http://www.gemplus.com/smart/cards/tech/)
[23] J. Park and R. Sandhu, "Role-based access control on the Web," *ACM Transactions on Information and System Security*, vol. 4, pp. 37-71, Feb. 2001.
[24] M. Hassinen and K. Hyppönen, "Strong mobile authentication," in *Proceedings of the 2nd International Symposium on Wireless Communication Systems*, pp. 96-100, 2005.
[25] Sun Microsystems, Inc., *Java 2 Platform, Micro Edition (J2ME)*. (http://java.sun.com/j2me/)
[26] Java Community Process, *JSR-000120 Wireless Messaging API*. (http://jcp.org/aboutJava/communityprocess/final/jsr120/)
[27] Java Community Process JSR-000177 Security and Trust Services API for J2ME. (http://jcp.org/aboutJava/communityprocess/final/jsr177/)
[28] Open Mobile Alliance, *Wireless Transprort Layer Security, Wireless Application Protocol Forum*, 2001. (http://www.openmobilealliance.org)

Marko Hassinen is a senior assistant and a Ph.D. student at the University of Kuopio, Finland. His main research interests include security protocols and implementations, mobile security and mobile payments. He also teaches several programming courses at the University of Kuopio, specializing in mobile and network programming.

Maija Marttila-Kontio is a Ph.D. student at the University of Kuopio, Finland. Her main research areas are visual programming languages and measurement systems. In addition to her research, she has taught several mathematic and data management courses.

Mikko Saesmaa is an assistant in the Department of Computer Science in the University of Kuopio. He is starting his post-graduate studies and is interested in structured document processing languages and their implementations.

Heli Tervo is a Ph.D. student at the University of Kuopio, at the East Finland Graduate School in Computer Science and Engineering. Her research interests are in structured documents, especially XML languages and technology with multi-channel publishing. Previously, she has worked in a research project "XML-Based Interfaces" (Univ. of Kuopio) and has experiences of information logistics industry.