

# flexible services

Eden

## Pilot case demonstrations D1.3.3

Document Name	Pilot case demonstrations
Project/WP Title:	EDEN WP1: Architecture and Technology
Document Type, Security	P (Public)

Document Title:	Pilot case demonstrations
Agreed date of delivery	31.8.2010
Actual date of delivery	30.9.2010
Editor	Heidi-Maria Rissanen, Miljenko Opsenica, Tomas Mecklin
Version and	version 1.0
Date Last Change	31.08.10
File:	Pilot case demonstrations D1.3.3.doc

Participants	Name	e-mail
Elisa	Olavi Karasti	
Ericsson	Heidi-Maria Rissanen Miljenko Opsenica Tomas Mecklin	
NSN	Janne Parantainen	
TeliaSonera	Olli Jussila	
TKK/CSE	Jaakko Kangasharju Sanna Suoranta	

## 1. Table of contents

1.	Table of contents .....	2
2.	Executive summary .....	3
3.	Architecture .....	3
3.1.	Principles of architecture design .....	3
3.2.	Component of the functional architecture .....	3
4.	Pilot analysis .....	4
4.1.	Lucre Wedding Use Case .....	4
4.1.1.	Description .....	4
4.1.2.	Architecture design analysis .....	4
4.1.3.	Component analysis .....	7
4.2.	EnviTori Use Case .....	9
4.2.1.	Description .....	9
4.2.2.	Architecture design analysis .....	9
4.2.3.	Component analysis .....	11
4.3.	MoFS Use Case .....	13
4.3.1.	Description .....	13
4.3.2.	Architecture design analysis .....	14
4.3.3.	Component analysis .....	15
4.4.	UDOI .....	18
4.4.1.	Description .....	18
5.	Next steps .....	18
6.	References .....	18

## **2. Executive summary**

This document lists the pilots for the use cases defined by the associated projects. Key technology components and enablers are listed and mapped to the pilot cases. The usage of the technology components is analysed for each listed pilot.

The results of the pilots will be aggregated and analysed with respect to their suitability and completeness to act as part of the ecosystem. The key enablers must not only provide for a certain service but also be a natural fit into the set of chosen enablers. The analysis results will in turn be used as input as the reference architecture evolves.

## **3. Architecture**

FSE Architecture Document [FSECO] lists and number of principles to be used for architecture design as well as component of a functional architecture. These are used as basis for the analysis of the pilots.

### ***3.1. Principles of architecture design***

The principles of architecture design, as defined by [FSECO], are

- Modularity
- Transparency
- Loose Coupling
- Composability
- Late Binding
- Reflection
- Bootstrapping through Established Infrastructure
- Easy Ecosystem entry

### ***3.2. Component of the functional architecture***

The component of a functional architecture, as defined by [FSECO], are

- Information
- Service
- Compensation
- Security, Trust and Privacy
- Authentication
- Authorization and access right management
- User management
- Media Management
- User tools

## 4. Pilot analysis

### 4.1. Lucre Wedding Use Case

#### 4.1.1. Description

The Wedding use case gathers the relevant functionality from the scope of the LUCRE project. The overall framework for the use case is the planning and execution of usual wedding in Finland. The use case covers functionality that happens before, during, and after the wedding. A few months before the wedding, together with the maid of honour and best man, the pride and groom use LUCRE tools to set up a service, which will serve all the participants of the wedding throughout the different steps in the wedding planning and execution.

The original scope of the Lucre wedding use case was changed. Different components of the service were studied and implemented individually but not integrated. The components were social media recommendations, maintenance of social lists, calendar integration, and the use of location information in user-created services. Thus, only individual components will be used in the analysis.

The main component in the uSpace underlying the Lucre pilot components is called space. A space can be used by multiple users, who can use widgets from the widget library and access sub-spaces of the current space.

#### 4.1.2. Architecture design analysis

The analysis is based mainly on Lucre D2.4 and D2.2. As described above, the Lucre use case concentrates on collaborative organization of a complex event or activities.

##### **Modularity**

If a system is modular, it consists of independent components, each providing a well-defined piece of functionality. One of the main targets in LUCRE project was to come up with common building blocks for services that can be used independently. Modularity of the system comes already from the used service creation platform, uSpace. uSpace is a widget framework where each widget can interact as an independent component. Each system component like an implemented toolbar in the Web browser, service creation platform uSpace or any implemented widget provides a well-defined piece of functionality. For instance, the social list widget contains collaboratively collected items that can be easily shared increasing flexibility. Another example is the integrated calendar widget, to which any new client can be added just utilizing adapters.

### **Transparency**

In a system with many actors involved, interfaces need to be clear and effects of operations well defined. In the LUCRE project, information and services are communicated through a set of open interfaces that support transparency in general. The modules/functions of the pilot are less integrated with each other than originally planned but individual components make a good base for the further research. Thus, all interfaces can be accessed by multiple service entities but there seems to be still some overlapping between different interfaces.

### **Loosely-coupled**

In a loosely-couple system, the dependency between components is low. Thus, it's easy to make changes without having to propagate changes throughout the system. In the LUCRE project, the connections are made in a loosely coupled manner and the services are anonymous to each other and the interaction happens asynchronously. Pilots follow loosely-coupled principle and use extensible data formats and protocols.

The main focus on this principle has been done in the calendar integration pilot. In the integrated calendar widget, which is a loosely-coupled service wiring using a RDF event-based store and adapters. Adapters mediate between the shared space and the services. The implementations of the services do not need to be changed and the services need not to know about the existence of the shared space. Each component can be replaced with an alternative by changing only adapters.

This principle is followed also in the social lists pilot where socially constructed connections across different media services are loosely-coupled in the cross-media widgeting environment. The list contains collaboratively collected items that can be shared and in that way increases flexibility.

Loose-coupling established in the project can be generalized to an interaction style that could be used also in other scenarios. Loosely-coupled connections enable participating services to remain relatively independent of each other, enabling independent evolution of the services.

### **Composability**

Composability means that it's possible to alter relationships between components dynamically. This enables creation of new concepts from existing ones. Project follows composability principle separately for an each pilot.

For instance in the social list widget, the communication is realized asynchronously via Ajax calls. This increases flexibility in the connections so that it is possible to alter the relationships between components dynamically.

In the location based pilot, the approach is based on operator positioning, which works in a broad range of mobile devices. This makes it possible to use the location information also in other services.

### **Late binding**

Late binding means that the connection between components is done as late as possible. Subsystems can be shared independent of existing services. Each LUCRE pilot follows late binding principle.

For instance in the integrated calendar widget, adding a new service doesn't require implementation change. Services don't have to be aware of platform existence and a new service can be added at any time.

### **Reflection**

Reflection is the ability of a system to inspect itself and its state, and to modify its state and composition at runtime through generic interfaces. Information is accessible from multiple sources in a transparent, easily verifiable, reproducible and scientifically defensible manner. In general, LUCRE follows reflection principle, although pilots don't really implement multiple services and integrate them to a final, compact solution. Currently implemented individual components are hard to test in reflection principle scope.

### **Bootstrapping through Established Infrastructure**

Interactions between services are bootstrapped through an established infrastructure using provided service discovery mechanism and interface definitions. The service creation platform uSpace used in the LUCRE project is based on widget framework approach, which provides bootstrapping principle environment.

### **Easy Ecosystem Entry**

The service developer will choose the most enticing ecosystem. For example, simple and unsurprising design, and usage of mainstream development platform entice service developers. Also, the ecosystem cannot be competitive unless there are enough services to choose from.

In the LUCRE project, uSpace provides a collaborative programming environment. Using the uSpace environment, service developers can build new widgets. For example, a REST API is possible to be used to build new widgets using the uSpace.

However, the biggest challenge for LUCRE is still how to get enough developers and services.

What it comes to individual pilots already implemented, for example mainstream development languages are used. For instance in the social lists pilot, the toolbar is a regular web application implemented using Java Server Pages (JSP) technology. The functionality and the user interface are

implemented with HTML, CSS, JSP standard tag library (JSTL) and JavaScript along with jQuery JavaScript library. Also Haavi Server provides access for multiple users to multiple databases. The toolbar is a standalone web application that provides easy integration, since the host site includes the toolbar via an iFrame element. Client side for Android provides communication with the server based on a REST API using JSON.

In the shared todo-lists pilot, OtaSizzle platform has a REST based interface used in collaboration widget.

In the calendar pilot, a microformat standard for displaying a semantic (X)HTML representation of iCalendar-format calendar information about an event is used between services and adapters. In addition, a REST-style API is used.

In the location-based pilot, a REST-like interface, RSS forms and SMS messaging are provided. Chosen positioning solution works in a broad range of mobile devices.

#### **4.1.3. Component analysis**

The analysis is based mainly on Lucre D2.4 and D2.2. As described above, the Lucre use case concentrates on collaborative organization of a complex event or activities.

##### **Information**

In EDEN D1.2.1 information entity is defined to consist of several subcomponents such as collection and processing of information, access and validity, and metadata handling. For example, authorization is needed for collection to ensure that proper access rights for collection are followed. As the scope of LUCRE changed, there are only individual components with internal information.

At the current version of the calendar pilot, for example, information about calendars of the users is synchronized. Thus, the visibility of the workspace can be limited for example to a particular group of users.

The location-based pilot collects information about the current location of the user. Location information can be further used in different services.

Information flow between different widgets was not implemented.

##### **Service**

A service is modeled in EDEN D1.2.1 as a collection of operations. Each operation has a specified set of parameters and results. Service component

consists of several subsystems, which are search, locate, delivery, management, monitoring and provisioning.

The service creation platform called uSpace (Lucre D2.2) was implemented as part of the Lucre project. uSpace platform provides a library, which can be used to search for widgets to be used in services.

New services are created using Service Composer, which is part of uSpace. Services are also managed using the uSpace.

OtaSizzle platform provides monitoring solutions and those solutions were used to monitor the components of Lucre.

### **Compensation**

Not valid for LUCRE use cases.

### **Security, trust and privacy**

Security and privacy issues are especially important as information is shared between different users. Also, there are a lot of privacy concerns related to cases where location information is forwarded to different widgets. However, privacy and security was potential area for the EDEN project to develop tools and applications so the LUCRE project did not study these issues further.

What it comes to social media, users can recommend services to other users. This type of social recommendation relies on trust.

### **Authentication**

Users in the Lucre pilot are part of the OtaSizzle community. Authentication in uSpace is done using the Aalto Social Interface (ASI) platform. A RESTful interface is provided for authentication in the services. Authentication uses Spring Security Module, which support basic authentication based functions such as role-based views. ASI acts as an authentication provider for the Spring Security Module.

Authentication to the uSpace can also be done using Facebook, but the implementation is working only in the development version.

### **Authorization and access right management**

Access and available functionality for a user is based on the role that the user has. For example, role-based views and role-based pages are supported by the authentication mechanism used in uSpace.

### **User management**

Users can have specific roles, and the uSpace system provides the means for user management. Spaces can be public, private or visible for a specific group defined by the administrator of the space.

### **Media management**

Services in the uSpace can utilize content from external sites. uSpace can also be integrated with existing social media services.

Haavi-Alma component combines content (news) from different social media. Haavis can be created and managed for collecting items around a specific theme, for example. Haavis can be shared with other users, and other users may also update the content of the Haavi. The host site including the Haavi toolbar fetches and stores the data from the database, which is located in the Haavi Server.

### **User tools**

uSpace is used as the service composer to build and manage new services. uSpace offers different possibilities to users with varying computing skills. Users can combine widgets from a library into a shared space and create new services. For example, shared todo-lists consist of three different services.

## **4.2. EnviTori Use Case**

### **4.2.1. Description**

The Air Quality use case developed by EnviTori-project is based on air quality measurement, collection and service creation based on collected information. Air quality is measured by individual customers, official information providers and private companies. Also metadata type of information is collected and linked to actual measurement information. This information is collected and processed and provided for service creation. Users of service are administration, companies and private users.

### **4.2.2. Architecture design analysis**

The analysis is based mainly on EnviTori market place service architecture draft (v. 0.0.6).

### **Modularity**

If a system is modular, it consists of independent components, each providing a well-defined piece of functionality. In EnviTori, the data is distributed to independent systems managed by different organizations. Each entity, that is organization, builds own modules (e.g. electricity processing EON) which can be aggregated in the marketplace using exposed APIs. Marketplace backbone APIs are connecting entities as loosely-coupled modules.

Different systems and individual contributors are supposed to be connected in a peer-to-peer network where a peer is an environmental data source.

### **Transparency**

In a system with many actors involved, interfaces need to be clear and effects of operations well defined. Open interfaces usage impacts transparency. Information and services are communicated through a unified set of interfaces. The implementation is not described in the documentation.

Data is gathered in web based data templates in secure way and the quantification system is automated. Marketplace provides tools and definitions required for buying and selling data, like catalog services, and metadata about existing data sources and services.

### **Loosely-coupled**

In a loosely-couple system, the dependency between components is low. Thus, it's easy to make changes without having to propagate changes throughout the system. In EnviTori, data sources and processing components are implemented as loose-coupled services. Interfaces seems to support loose-coupled independence, and extensible data formats and protocols are used (WMS, WFS, REST, SOAP, HTTP, CGI).

### **Composability**

Composability means that it's possible to alter relationships between components dynamically. This enables creation of new concepts from existing ones. EnviTori marketplace is a peer-to-peer network, which guarantees flexibility in the connections on the top of architectural flexibility. Fitting together different actors owned equipments and technology was not an issue since all actors had a positive attitude towards testing and adapting something new. The implementation is not described in the documentation.

### **Late binding**

Late binding means that the connection between components is done as late as possible. Late binding principle has not been enabled by composability. Contributors' subsystems can be independently associated with a marketplace at any time. The implementation is not described in the documentation.

### **Reflection**

Reflection is the ability of a system to inspect itself and its state, and to modify its state and composition at runtime through generic interfaces. Information is accessible from multiple sources in a transparent, easily verifiable, reproducible and scientifically defensible manner.

### **Bootstrapping**

Interactions between services are bootstrapped through an established infrastructure using provided service discovery mechanism and interface definitions.

### **Easy Ecosystem Entry**

Using WMS, WFS, REST, SOAP, HTTP, CGI interfaces simplifies an entry point and speeds up integration. An access point can be automatic generated using interface definitions e.g. WSDL. There is a strong focus on simple design. There is no competitive ecosystem of services but rather a single choice of service per single field.

MediaMarket provides a path from an initial single person effort with 10 users to a national-wide service.

### **4.2.3. Component analysis**

The analysis is based mainly on EnviTori market place service architecture draft (v. 0.0.6).

#### **Information**

In EnviTori, there is an environmental information marketplace that provides tools and definitions needed to sell and buy environmental information. It is not centrally managed, but data is distributed into systems managed by different organizations. Clearly, information is an important architectural enabler as defined in EDEN D.1.2.1. Information is provided to users through unified APIs (Application Programming Interfaces).

Information might also have a model, which models a phenomenon. A sample of a phenomenon would be for example weather forecast model.

- *Collection and processing of information* - Information in the EnviTori pilot is collected from the environment using different kind of information producers. Information is produced by normal users, public sector, companies and also sensors. Users can also act as "sensors" and provide subjective observations about the surrounding environment. Information collection can be simple, one type measurements or complicated measurements consisting of both simple and sophisticated measurements. It can be collected for example about demographics, usage of electricity.

Information usually has to be processed from raw data to value-added data before it is forwarded to end users. In the same way as measurements can be either simple or sophisticated, information processing might be simple or excessive. Databases are used for storing data and metadata.

Common environmental data formats are used in the project.

- *Metadata handling* – Metadata is data about data, defining for example properties of the data. In EnviTori, metadata can be

collected either from the same source as normal information or from a different source.

For the environmental data to be useful there needs to be enough metadata available. Metadata is collected in all steps of the service chain and it is maintained for validation purposes. Metadata for environmental services can be for example location information. Metadata describes for example who is allowed to access the information and how it can be used.

- *Access and validity* – Strong security is applied on all levels. Metadata of the information can be used to limit the access. The implementation is not described in the documentation..

### **Service**

A service is modeled in EDEN D1.2.1 as a collection of operations. Each operation has a specified set of parameters and results. In EnviTori, users of the services can be either consumers or companies that use provided services in their own business. Each service has to provide at least one interface, which can be either for human users or for M2M usage.

Services are then created based on the provided information, which is possible processed. There are different subcomponents related to service, which are used by service users.

- *Search* – Catalogue service is provided to search data sources and users. A Catalogue Service API for searching and browsing for services is provided for M2M communication. For human users, an UI is provided.
- *Management* – Services can be registered manually by service providers, or by using a provided API.
- *Provisioning* - Services can be registered manually by service providers, or by using a provided API.

### **Compensation**

As service providers provide input to the marketplace, some kind of compensation is needed. In the EnviTori architecture, there is Service Usage Agreement, which handles compensation to the information providers. However, it seems that handling of payments is not implemented. *The implementation is not described in the documentation.*

To make the agreements, an Agreement Manager Interface is provided. Dynamic service agreements are not supported.

There are possibilities for different big actors to use the data, and to build up or make better different services or products. There are also possibilities for different individual actors to use collected data as a compensation for providing input. However, there is still a gap between current data collection

solution and the cloud thinking. In addition, there is not yet clear and concrete “need” among the customers for the environmental data. Further building up customers' environmental awareness, motivation for contribution and usability of collected data will fill that gap.

### **Security, Trust and Privacy**

EnviTori system supports strong security in all levels, but implementation is not discussed in the documentation. However, it is mentioned that logging into services is handled using HTTPS, SSH and restricted IP addresses. The implementation is not described in the documentation.

Information is protected from miss usage and user is anonymously protected. The situational awareness and security in general is provided although individual actors still needs an additional motivation and stronger awareness on trustfulness. The implementation is not described in the documentation.

Some measurement data (e.g. electricity) is considered as confidential data which is a barrier for delivering the data. The EU transparency generates positive choices from the environmental perspective.

### **User Management**

As there are several user types involved for example as information providers, there needs to be different versions and adaptation of the User Interface (UI). How this is implemented, is not described in the documentation. The implementation is not described in the documentation.

### **Media Management**

Media management is not valid for the environmental information marketplace as such, but the applications built on top of the will need to have mechanisms for media management.

### **Authentication**

Authentication is needed for example for collecting information. TODO How implemented?

### **Authorization and Access Right Management**

As there are different type of information providers, access right management and authorization is needed to be used. The implementation is not described in the documentation.

### **User Tools**

User tools enabler is not applicable to this use case.

## **4.3. MoFS Use Case**

### **4.3.1. Description**

These use cases are developed by the MoFS (Mobile financial Services) project. The fundamental service for all use cases is strong mobile authentication and security. The payments can be used for ticketing, e.g. public transportation, and micro-payments. The focus of the first project year is Ticketing, Payment with digital services and Rewarding. The use cases for the second project year are Proximity payment and Mobile banking.

Private persons in various roles are the main users of these services; public transportation passenger, on-line bank user and web-site (on-line store) customer.

Two pilots are conducted within the scope of MoFS;

- **Pro-active banking:** Enables end-user to control personal everyday economy and invoice acceptance via mobile application. Pilot will be conducted together with Nordea.
- **Mobile ticketing:** Enables end-user to purchase ticket products and value to his/her public transportation travel card. Also enables checking of his/her travel card balance and the shopping history via mobile application. Pilot will be conducted together with Tampereen joukkoliikenne.

NOTE – MoFS Pilot evaluation reports are not available yet, so this document is based on the planning information and documents in MoFS. Therefore, analysis of the architecture within MoFS was in some cases not possible to do.

#### **4.3.2. Architecture design analysis**

The analysis is mainly based on “MoFS deliverable 2.1, Strong end-user authentication with mobile devices – use cases, authentication methods and implementations” version 1.0 [MoFSAuth].

##### **Modularity**

The listed authentication mechanisms within MoFS are well-defined, thus enabling compositions of multiple services using various IdP's. Functionality of the identity provider modules are also quite well-defined making them the enabling modules in a service ecosystem where strong authentication is required.

##### **Transparency**

The interoperability between different Identity Service Providers is not in the scope of MoFS, limiting the transparency of the services.

However, the vision state that it shall be easy to take the mobile authentication method into use and maintain it while changing the mobile

device, mobile operator or the Identity Service Provider. Thus, for an user the service is somewhat transparent independent of the aforementioned selection.

#### **Loose coupling**

Not enough data for analysis.

#### **Composability**

The interoperability between different Identity Service Providers is not in the scope of MoFS, limiting the composability of the services.

#### **Late binding**

Not enough data for analysis.

#### **Reflection**

There is no information on how or whether reflection is considered within MoFS pilots.

### **Bootstrapping through Established Infrastructure**

The proposed solutions for authentication within MoFS relies on the well established mobile networks, usage of smart cards and banking authentication systems. The enabling technologies for user interfaction are

- SMS
- SIM application Toolkit (SAT) client
- Java MIDP Client
- Native Client (e.g. Symbian)
- Browser
- Widget

#### **Easy Ecosystem entry**

There are quite well defined tooling for the aforementioned enabling technologies, thus making it attractive and easy to enter the ecosystem with new services.

### **4.3.3. Component analysis**

The analysis is mainly based on “MoFS deliverable 2.1, Strong end-user authentication with mobile devices – use cases, authentication methods and implementations” version 1.0 [MoFSAuth].

### **Information**

Not valid for MoFS pilot.

### **Service**

Not enough information for analysis.

### **Compensation**

By using the pro-active banking widget, the end user can handle everyday economy, such as pay invoices, check balance, locate the nearest ATM and also contact the banking personnel by e.g. calling. Being able to access the account information, strong authentication is required. However, the pro-active banking widget does not allow for purchasing content or products dynamically.

The mobile ticketing widget allows the end user to purchase travel cards and check the history and balance of the card. Also, using NFC enabled devices, the mobile phone is used to pay the ticket on public transportation. In MoFS User Study on Mobile Ticketing (Owela & focus groups) [MOFSUG], some of the target group pointed out the lack of NFC devices.

Note - Ticketing pilot will be conducted 2H2010 and reports on proactive banking are not available.

### **Security, trust and privacy**

Security of user transactions in MoFS is handled using one of the well-established technologies for mobile security. As the final report from the MoFS project is not available, the recommendations and results of the various methods is not known.

### **Authentication**

The list of high-level authentication methods defined in Mobey Forum Mobile Authentication Survey [MOB06] lists the following methods:

- Digital Signature
- Locally Generated OTP
- Remote OTP
- Mobile device as a Secure Element reader
- Phone number
- Biometrics

Of these, phone number only is not seen as a strong authentication method and biometrics is not studied within MoFS due to lack of compatible devices in Finnish market.

In MoFS, the following strong authentication methods are listed as potential base for implementation;

- Local OTP generation with the ObCs framework
- Mobile CAP
- Mobile OTP-list on a SIM
- Mobile OTP-list on ObC hardware
- Sim based Wireless PKI
- Software based OTP generation
- HTTP Digest with UICC based shared secret

These mechanisms will be used for further usability analysis within the scope of MoFS.

Interoperability between the authentication schemes of different Identity Service Providers (IdP) is not in the scope of MoFS. However, the MoFS vision state that "it shall be easy to take the mobile authentication method into use and maintain it while changing the mobile device, mobile operator or the Identity Service Provider."

Note - Results of usability analysis not available.

### **Authorization and access right management**

In MoFS, authentication is used to identify the user that attempts to access a certain account. This information is used to authorize the user to perform transactions on the account based on the provisioned access rights of the authorized user.

### **User management**

See above.

In addition to the authorization and access rights management system of the banks, the pilots depend on the user database of the mobile operators and the mobile handset identification performed by the mobile network.

### **Media management**

In [MOFSUG], the participants brought up ideas where the mobile ticketing could be combined with e.g social media and other services. In the MoFS use cases, media management is however not valid.

### **User tools**

The main tool for the end user is the mobile phone, which acts as the payment device for public transportation as well as the access point to travel plans.

#### **4.4. UDOI**

##### **4.4.1. Description**

UDOI project analysed the methodology and processes of User Driven Open Innovation, and is not concerned with the technological solutions used within the innovations. Therefore, UDOI analysis in terms of the architecture and components is not valid in the scope of this document.

UDOI selected 3 cases to study the processes and methods for open innovation:

- Augmented Reality
- Electronic Service Vaucher
- Mobile Ticketing

The results of UDOI are decribed in UDOI Case Research; Used Methods & Learnings [UDOIML]

## **5. Next steps**

Since the final reports of the pilot cases are not yet available, analysis of the implementation was not possible. The analysis could be completed once the final reports are available.

## **6. References**

[FSECO] D1.2.1 FSE Architecture Document

[FSEUC] FSE Use Case and Architectural Enabler Analysis

[MOFSTICK] Mobile ticketing application UI design document, MoFS WP4

[MoFSAuth] MoFS deliverable 2.1, “Strong end-user authentication with mobile devices – use cases, authentication methods and implementations” , version 1.0

[MOB06] Mobey Forum Mobile Authentication Survey Analysis, Juha Risikko, Bishwajit Choudhary, 2006

[MOFSUG] MoFS User Study on Mobile Ticketing (Owela & focus groups), version 1.1

[UDOIML] UDOI Case Research; Used Methods & Learnings