



Research Center
NRC-TR-2008-003

Architectural Solutions for Mobile RFID Services on Internet of Things

Martin Peter Michael and Mohsen Darianian
Nokia Research Center, Helsinki, Finland
<http://research.nokia.com>
April 18, 2008

Abstract:

Mobile RFID services for Internet of Things can be created by using RFID as an enabling technology on mobile devices. Humans, devices and things are the content providers and users of these services. Mobile RFID services can be provided either on mobile devices as stand-alone services or combining with end-to-end systems. When different service solutions are considered, there are more than one possible architecture solution in the network, mobile and the back-end server areas. Combining the solutions wisely by applying the software architecture and engineering principles, a combined solution can be formulated for certain application specific use cases. This paper illustrates these ideas to create some reference solution models. It also shows how commonly the solutions can be used in real world use case scenarios. A case study is used to add further evidence.

Index Terms:

Domain-specific architectures
Service architecture
Mobile RFID
Mobile RFID services

Architectural Solutions for Mobile RFID Services on Internet of Things

Martin Peter Michael

Nokia Research Center, Helsinki, Finland
martinpeter.michael@nokia.com

Mohsen Darianian

Nokia Research Center, Helsinki, Finland
mohsen.darianian@nokia.com

Abstract

Mobile RFID services for Internet of Things can be created by using RFID as an enabling technology on mobile devices. Humans, devices and things are the content providers and users of these services. Mobile RFID services can be provided either on mobile devices as stand-alone services or combining with end-to-end systems. When different service solutions are considered, there are more than one possible architecture solution in the network, mobile and the back-end server areas. Combining the solutions wisely by applying the software architecture and engineering principles, a combined solution can be formulated for certain application specific use cases. This paper illustrates these ideas to create some reference solution models. It also shows how commonly the solutions can be used in real world use case scenarios. A case study is used to add further evidence.

1. Introduction

Internet of Things (IOT) is a concept [1] that visualizes the vision for bringing the internet even to dummy things. By bringing the internet to the dummy things, new services can be created and be used by things, devices and humans. To make the dummy things to smart things, IOT report suggests Radio Frequency Identification (RFID) as one of enabler technologies. Smart mobile devices are ubiquitous devices with desktop processing capability and have confluence of services, sensors and technologies. Thus they are the natural mediators for interaction between humans, devices and things. A smart mobile phone with RFID reader is called Mobile RFID (MRFID) reader. MRFID reader brings mobility concept to the RFID technology and expanding its horizon to create services.

Mobile RFID services are mobile services that use RFID as an enabling technology to access the information on the RFID tagged objects over a telecommunication or other network [2],[3],[4]. Simply, mobile RFID services provide information on RFID tagged objects. These services need to be

carefully designed so that they can be intuitively used by humans, devices and things, to mash up services, and to federate new services along with others. Moreover, the new service solutions should overcome the future technological challenges.

Considering the requirements for MRFID services, there is a need for architectural level solutions as the services itself could be created by mashing up other available services either on MRFID reader alone or by combing some end-to-end systems. In doing so, there are various issues to be considered. They are enabling a mobile device into a MRFID service platform, managing the services and contents, MRFID service classification, recognizing the pattern of use cases, type of service solutions and factors affecting the quality attributes of services. Moreover, when more than one solution option is available at different levels, there needs a way to make a rational judgment.

This paper analyzes few service solutions from different perspectives to create reference models that fit for similar type of MRFID service use cases. It presents a reference service architecture which can be used for developing MRFID services. Services are analyzed and classified along with the solution types. A framework for creating MRFID services are provided at different levels. The factors involved to judge a solution among the available options are also shown. Combination of solutions is used to overcome drawbacks of certain solution model. In addition, challenges in mobile device and network while designing service solutions are addressed. A real case study is used to prove the concepts practically.

Currently, MRFID services are provided by applying few software architectural principles like REpresentational State Transfer (REST) [5], Event Driven Architecture (EDA) [6], Service Oriented Architecture (SOA) [7] and push/pull [8] model on the existing architectural framework provided by the mobile device platforms along with web services on the internet. Our approach addresses how these principles can be used in a reasonable manner on mobile devices to meet their quality requirements of services. Also the framework solution models that we propose are easy to design new MRFID services.

The rest of this paper is organized as follows: Section two presents architectural solutions; Section three describes a case study and Section 4 presents our conclusions.

2. Architectural Solutions

A MRFID reader platform consists of RFID reader hardware and software integrated on a smart mobile device. It is a basic platform to host RFID software services.

2.1. Mobile RFID Reader Platform Architecture

The simple way to create a MRFID reader platform is to attach an RFID reader hardware to the mobile device via some standard hardware interface. A smart mobile device has interfaces in the hardware to connect new hardware devices and sensors. It is also possible to simulate such an interface. Smart phones usually have hardware interfaces like USB, serial interface and Bluetooth. Using one such interface a RFID reader can be connected to the phone hardware. The RFID reader hardware requires a power supply and a communication interface. The communication interface is used to send and receive the control and data signals [9].

A smart phone software platform usually provides software interfaces to connect other hardware devices or sensors. One way to connect the RFID reader hardware to the phone software is to use an UDP port [9]. The software connection to the underlying communication varies depending upon the implementation of the particular software device platform. The phone software platform also needs some driver services. The driver services should be layered very thin so that they would accommodate future changes in the RFID hardware. Thinner and modular driver services are recommended here.

2.2 Reference Service Architecture for a mobile RFID reader

The RFID software services run on the RFID reader platform. The proposed reference service architecture is depicted in Figure 1.

This is a typical layered architecture which accommodates the changes in the software and hardware components over time. RFID reader services consist of middleware and core services which communicate to the RFID hardware via driver software. RFID core services include basic services

like reading a tag, locking, writing and killing a tag using reader tag communication protocol. RFID middleware services are built on top of the RFID core services. RFID middleware services are high level services such as filtering, aggregating, grouping, counting, performing differential analysis, decoding the data into various formats, providing data for multiple clients, configuration and power control [10].

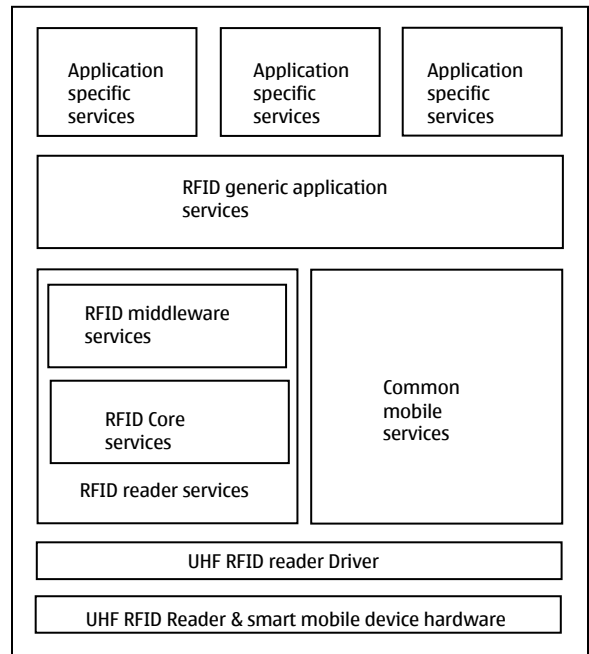


Figure 1: Reference service architecture of a MRFID reader

Common mobile services are the services which are generally available on any smart phone device. Some examples of these services are connectivity, networking, security, radio access, storage, multithreading, multimedia content handling, application management, web, location and graphics.

RFID generic application services layer consists of services such as attaching a tag, detaching a tag, reading information from a tag and updating information on a tag. These services are not specific to any application, but they can be generally used to compose specific application services. RFID generic application services are composed by using middleware services with common mobile services. They can be implemented in a common mobile runtime platform like Java Platform, Micro Edition (J2ME) [11].

The top layer of the architecture hosts MRFID application specific services. For example a library use

case which uses RFID tags to identify the items is an application specific RFID service.

2.3. Service Types

Mobile RFID services can be classified based on the location where the information content and services are accessed from and the way the services are initiated.

Content and service storage:

The content and services can be stored locally on a tag or on a mobile device or somewhere in the network. Stand alone services run locally and rely on a tag or a MRFID reader for the content of the services. End-to-end services rely on a local device or network and backend systems for the information content.

Service initiation:

Mobile RFID services can be initiated by humans, devices or things. Depending upon the initiators the services can be classified into Human Initiated Services (HIS), Device Initiated Services (DIS) and Thing Initiated Services (TIS).

HIS are designed for specific use case applications. A user starts an application service manually (by hand) to use the RFID services along with or without other services in the smart mobile device. A typical example is a user tries to read information about an item in the shopping mall using his MRFID reader. TIS runs when the mobile device reads a tag. There is no human intervention needed for these kinds of services. TIS is pre-registered on a device to run on specific things or events produced from things. For example, a security service which logs the visits of security personnel near a door can be created by identifying a tag on the door by his/her MRFID reader device. Whenever he/she visits the door, the MRFID service wakes up and runs to fulfill a certain task. DIS is started by a MRFID reader device itself or by some other device in the network. The service initiation in DIS is not by a user or by a thing. An example of TIS service could be a scheduled service which runs on the MRFID reader to read the nearby tagged things.

2.4. Basic Use Cases

All RFID services need to use four basic use cases. They are 1) *Attaching information to a tag* 2) *Detaching information from a tag* 3) *Reading information from a tag* 4) *Updating information to a tag*.

When RFID tags are used for item identification, in addition to the physical attachment of the tags to the things, the information content should also be attached with the ID of the tag in the digital world. The information content of an item can be either on a tag or on a device or somewhere in the network. But the procedure of attaching information is the same. The attaching information use case is usually executed only once for a physical thing. This use case could also be executed if there were a need to physically replace the tag. Usually, the use case is handled by some authorized person.

Detaching information from a tag is done only when there is no need for a particular item in the digital world anymore. This usually happens only once like when an item's lifetime is over or item identification is not needed anymore due to privacy reasons.

Reading information about an item is the most frequently used use case. This is because anyone who is interested in knowing about an item would need to read it first. Reading not only provides information but also services an item could offer. Today most of the MRFID application services are built on reading use case.

Updating information is needed when an item goes through changes of state over a period of time. To store the state of change, an RFID tag needs information updating and thus needs special access.

All these basic use cases can be used as reference use cases for building specific RFID application services. In order to identify a reference use case, that is encapsulated in an application specific use case, the application specific use case must be broken into a level of granularity same as the reference use case. Thus it is possible to breakdown the application specific use cases into reference use cases and other use cases. Once this is done, they can be normalized and reused.

2.5. Stand-alone service solutions

There are three solution models we propose under stand-alone services. They are based on the location in which the information content and services are kept. They are a) content and services on mobile device; b) content on tag and service on device; and c) content and services on tag. These three types are illustrated in Figure 2.

Solution (a) can use inexpensive tags like EPC class1 Gen2 tags. In this case, as the storage on the tag is limited, only IDs are stored on the tags. RFID services and content are hosted on a mobile device.

When HIS is considered, the user brings the MRFID reader to the tag. The RFID reader service which runs on the mobile device recognizes the tag ID and provides the information content accordingly.

The second solution (b) needs tags with bigger memory in order to store information content along with tag IDs. Obviously these tags are more expensive than the tags used in the first solution. But RFID services are the same as in the first solution. Solution (b) provides more flexibility because the services are not bound to a particular mobile device as the information contents are distributed. Security must be implemented on tag level contents.

In solution (c), in addition to the information content, the RFID services are also kept on a tag. Thus it really distributes not only the content but also the RFID services on things. The physical thing which is attached to a tag can thus provide data about itself and its services. Any capable mobile device can run those services on the fly. Security must be enforced on data and content in both directions. The mobile device needs to know if it can trust the service stored on the tag and vice versa. A security certificate can be stored on a tag to enforce the protection. For example, a java binary file which implemented a service can be stored on the tag. When a MRFID reader reads the tag, the certificate is also read by the reader. If it is an authenticated device, the binary is then read on the device and installed. By running the service the data on the tag can be interpreted.

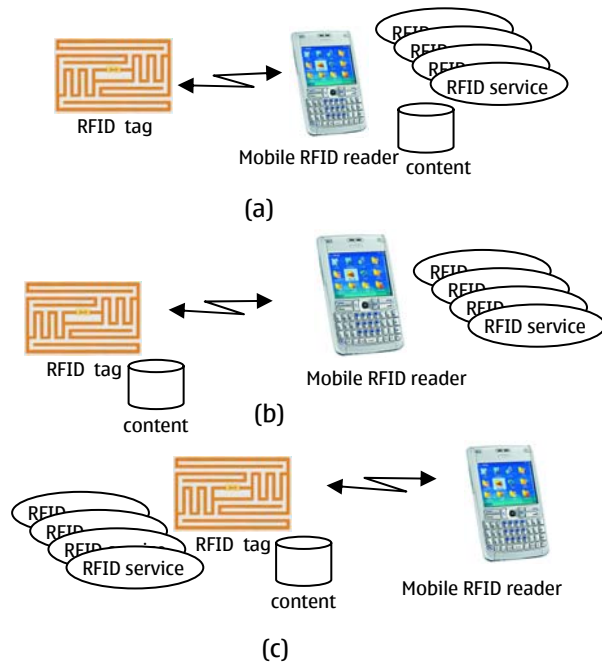


Figure 2: Stand-alone service solutions

For using any of the three solution models, the mobile device should run some common services. They are security service and data handling service. Security service implementation varies depending upon solution. Data handling service interprets the data from a tag to meaningful content.

2.6. End-to-End service solutions

The three solutions introduced for end-to-end service solutions are a) Thin client based b) Rich client based and c) Quasi rich client based.

A reference service architecture for end-to-end service solutions shown in Figure 3 use the same concept of layers as in section 2.2. Layers above the RFID application services differ because of the functionality between thin and rich client solutions. A thin-client based solution (a) uses an adaptation service layer in between a web browser application and RFID generic application services. Whereas rich-client based service solution (b) directly uses the underlying RFID application services for mobile applications. Note that the basic use cases which are translated to its equivalent services run on the lower layer for both of the solutions.

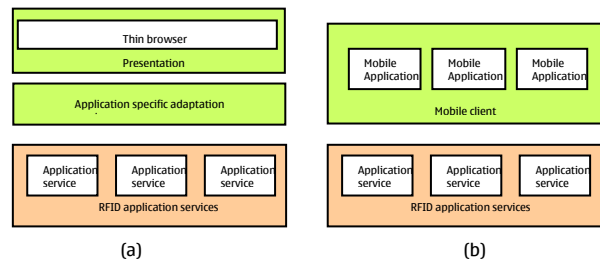


Figure 3: Thin-client and rich-client based service architecture on MRFID reader

A thin-client based solution uses a web browser as a platform for providing RFID services. Application services and content reside on a server across a network. Services are also executed on the server. However MRFID services which run on the web browser need some support from lower layers of the device itself. For example, transferring data from a MRFID reader to web browser and vice versa must be supported on the device. This support can be achieved by adding a thin adaptation service layer in the service architecture which acts like a glue. Installation and maintenance of thin-client is easy as the services are on the server. Only the thin glue services need a rare update. Thin-client solution is also compatible with

multiple MRFID readers which are browser compatible. The main drawback of this solution is low performance. It takes sometimes a long response time for a web browser to respond.

A rich-client based solution uses a separate stand-alone application which is network independent. Services and content are kept locally on a device, but content is also in the server. Thus the services can be executed on a device without any network support. However these services may periodically need some network support as they use end-to-end services. Thus rich-client based solution needs some extra services such as data synchronization and local data storage. Moreover a rich client needs to be installed separately on every device. The maintenance of the service is costly. Rich client is also not compatible with multiple devices. The main advantage of the rich client is that the performance of MRFID services is very quick. The main drawback is the periodical update of content with the central server. If multiple clients have to be used for MRFID services, rich client solution cannot offer up-to-date information about the items.

A quasi-rich-based solution is a compromise between a thin-client and rich-client solutions. It is not exactly a rich-client but acts like a rich-client. It can also be perceived as more than a thin-client. In quasi rich-client solution, the drawbacks of thin-client and rich-client solutions are removed. A quasi rich client is a stand alone mobile application but it is optimized to run the services on the device itself by keeping the content on the server. REST is used to achieve the speed to access the content across the network. For every MRFID service, the quasi client uses one or more RESTful services. It creates high data traffic than a rich client but not a heavy one. It is because it doesn't use XML or SOAP. And it doesn't need any extra memory. It is compatible with multiple devices as it doesn't need to store any contents locally. It also offers good performance as it quickly responds to the user. The only drawback of this solution is that it needs extra maintenance and support as the application has to be installed on every device separately.

2.7. Network Configurations

All three solutions need an end-to-end network infrastructure. When communication is concerned, fixed connections are used to connect the desktop clients to the back-end servers. MRFID readers are mainly connected via UMTS or WLAN networks to the back-end system.

Back-end systems consist of information servers, web servers and data base systems. A database system

is used to store the information table about the items. Usually a Relational Data Base Management System (RDBMS) is a good choice. Information services provide information content for information queries over the web using some database server. Thus a web service is an example of an information service.

RFID services provided by enterprises usually involve heterogeneous networks like the enterprise intranet, internet and mobile operator UMTS network. They are all needed to exchange data between a back-end server and MRFID readers. Proper attention must be paid if the corporate services and content need to be mashed up with the RFID services. Security analysis must be done for every RFID service and access control mechanisms should be implemented on network elements. Moreover, network elements must be kept at correct zones of a network in order to avoid any security holes.

Usually there can be more than one network configuration possible. By choosing an optimal configuration at the architectural phase a significant amount of work can be avoided at the implementation of services.

2.8. Quality attributes

The two most important qualities of MRFID services that need to be considered during the architectural phase are Runtime System Qualities and Non-Runtime System Qualities. Runtime System Qualities are the measurements that we would like to have in the MRFID services while the RFID system executes. It includes functionality, performance, security, availability, usability and interoperability. Non-Runtime System Qualities cannot be measured as a system executes. Modifiability, portability, reusability, integrity and testability are considered as Non-Runtime System Qualities.

Quality attributes affect service design. For example, when the performance attribute is considered, there is a demand from the end-user that RFID tags should be read by a MRFID reader between 1 and 1.5 seconds. And, while reading, a service should not hang or crash the device. Thus the performance attribute adds extra constraints to the service design. In order to overcome these constraints, an RFID reader platform must be designed by considering performance as a key factor. When the availability attribute is considered, a user expects an RFID service to be available 100% of time. Thus it puts an additional constraint to the design to make a service available all the time. A stand-alone MRFID service solution meets the availability criteria as it doesn't depend on the network to provide the

service. Similarly every quality attributes brings extra constraints to the service design. These quality attributes thus drive us to choose the right option.

3. Case Study – Library Item Tracking System

This section describes a case study in which we designed a service architecture and a solution based on introduced performance optimized quasi-rich service architecture for a corporate library by using RFID technology. The Library Item Tracking System (LITS) has been developed for the corporate library to manage and track the items such as books, media items, magazines, journals, and so on. LITS provides services which benefit the library, helps a postman to collect and deliver items, informs the subscriber (employee) about the status of the items and interacts with other legacy systems in the corporate to search and manage items, to make decisions, and to produce reports.

Functional requirements of LITS were gathered by interviewing stakeholders and by studying the ongoing process of the library in place. We found the use cases of the LITS as attach tag, track journals, sort journals (by reading the subscriber info), collect journals, deliver journals, skip subscriber, extend subscription period and detach tag. From these specific use cases, we found out the generic RFID use cases. Then use cases are named as equivalent application specific services as shown in Table 1. Finally, from those services, four reference services (section 2.4) were identified. In addition, some service parameters were also determined.

We concluded to use the same reference architecture for MRFID readers and RFID tags on every library item. The quality attributes for the services were listed and used as a driver to make decision about a particular solution among the available solutions. Few possible network configurations have been analyzed to solve some problems at the architectural level. For example, external postal staffs don't have access to the corporate services or content. In order to solve this, we placed the LITS server in the internet. The MRFID readers were connected to the LITS by the GPRS operator network. Also we took advantage of some of the corporate IT services and security services. The hybrid alternative of Internet & Intranet - UMTS Based combination was chosen for network configuration (see Figure 4).

Table 1: LITS use cases mapping

Sl. no.	LITS application specific Use cases	Generic RFID use case	LITS application specific services	Generic RFID application services	Generic RFID application service parameters	Service classification (unique, repeated)
1.	Attach tag	Attaching info with an item	Attach info	Attach tag	{service_name, reader_id, tag_id, info_field1, info_field2, ..., info_fieldn}	unique
2.	Track Journal	Tracking an item	Read subscriber	Read info	{service_name, reader_id, tag_id, info_fieldn}	unique
3.	Sort journals (by reading subscriber info)	Reading info from an item	Read subscriber	Read info	{service_name, reader_id, tag_id, info_fieldn}	unique
4.	Collect journals	Updating information	Collect journal	Update info	{service_name, reader_id, tag_id, status_fieldn}	unique
5.	Deliver journals	Updating information	Deliver journal	Update info	{service_name, reader_id, tag_id, status_fieldn}	repeated
6.	Skip Subscriber	Updating information	Skip subscriber	Update info	{service_name, reader_id, tag_id, status_fieldn}	repeated
7.	Extend subscription period	Updating information	Extend subscriber	Update info	{service_name, reader_id, tag_id, status_fieldn}	repeated
8.	Detach tag	Detaching info from an item	Detach info	Detach tag	{service_name, reader_id, tag_id}	unique

Logical service blocks of LITS consists of services provided on mobile device, web client and backend server. A LITS service architecture for mobile device was created from the reference architecture by grouping all the common services by layers. It was obvious that the LITS RFID services would use an end-to-end service solution so that the library items could be tracked online. We chose a quasi rich-client service architecture as depicted in Figure 5. Service initiation was always done by humans in LITS. REST principle was used to communicate between the client and server. Security was ensured as HTTPS protocol was used with REST.

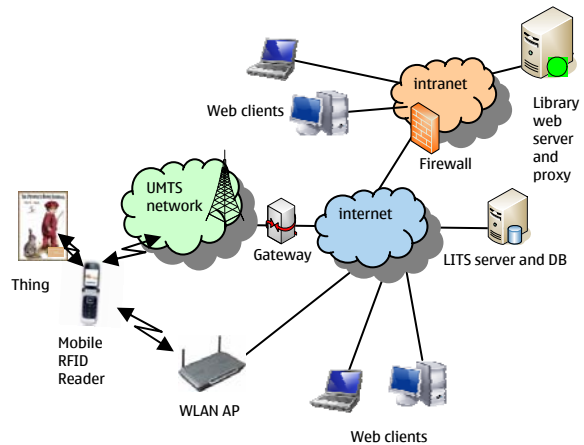


Figure 4: LITS network configuration

We evaluated the implemented MRFID service solution of the LITS system against functional requirements and quality attributes. The whole LITS system has been very successful in providing RFID services to its mobile clients and desktop clients as expected. Thus it met all the functional requirements. We have verified the system by conducting a couple of field trails with 50 journals, 200 subscribers, two librarians, two postal staff members and one administrator. The non-functional requirements such as runtime qualities, non-runtime qualities, business qualities, and architectural qualities were evaluated in different tables. The evaluation tables showed how the selected solution fitted to the problem.

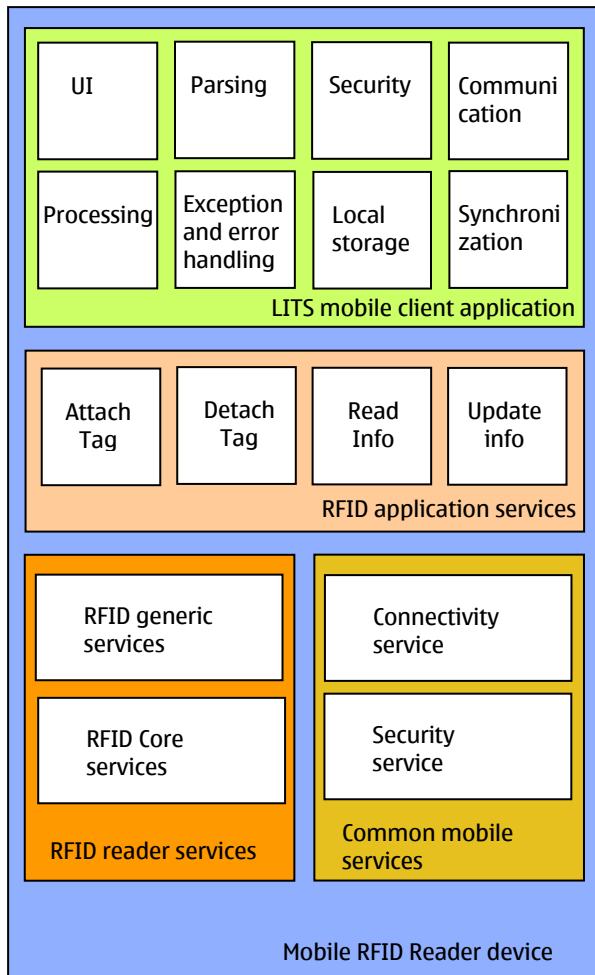


Figure 5: A quasi-rich-client based service architecture

As an example we depict the performance quality attribute in Figure 6 as a graph. The average response time for the read service was measured from end-to-end both locally in Finland and also remotely in Germany. The backend server was always kept in

Finland whereas the MRFID reader was connected to the server via GPRS network in both the cases. From the graph, we can clearly see that the average response time for RFID read service is around 1.1 seconds in Finland and 1.35 in Germany. Thus we achieved the best performance for the service by using a simplified REST principle instead of some heavy architectures. The designed service architecture clearly met the functional and non-functional requirements of LITS.

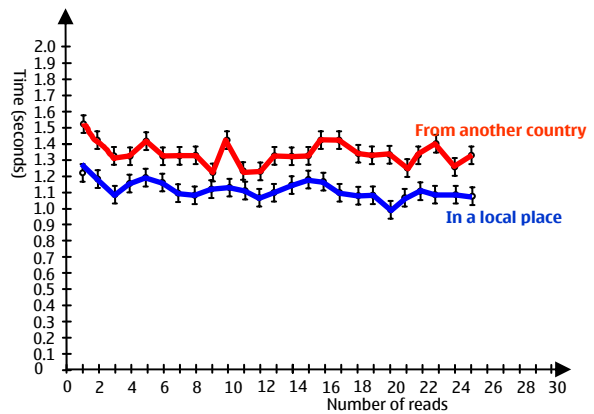


Figure 6: End-to-end empirical performance measurement of MRFID reader

4. Conclusion

This paper introduced some of the architectural solution models for mobile RFID (MRFID) services. Software architectural principles are applied to formalize service solution models from practical and theoretical perspectives. A case study has been used as a reference to add more evidence for the theoretical hypothesis.

It was found that MRFID services can be provided on the device itself or using end-to-end systems. For both cases, solution models were created by applying software architectural principles including SOA, REST, EDA, and Push & Pull. We found out that it is possible to achieve a compromise between a thin-client and rich-client service solution. We called it a “Quasi-Rich-Client Service Solution”. A quasi-rich-client service solution was proved to achieve the maximum value for the quality attributes. When there are more than one solution option is available, the best way to make a judgment is to use the quality attributes of the services from end-to-end.

The conventional ways of building a software product from use cases need to be changed while building software services. During the architectural phase, the functional requirements have to be grouped into use cases and then the common denominators are

grouped as services. Use cases must be exploded to find out the common services. These services can be further exploded into fine-grained services or kept as close-grained depending on how they will help to achieve the expectations of the quality attributes.

References

- [1] International Telecommunication Union (ITU), ITU Internet Reports, The Internet of Things, November 2005.
- [2] Seidler, C., RFID Opportunities for mobile telecommunication services. ITU-T Lighthouse Technical Paper, 2005.
- [3] Kim, J., Choi, D., Kim, I., and Kim, H., Product Authentication Service of Consumer's mobile RFID Device. In: Proceedings of the 2006 IEEE Tenth International Symposium on Consumer Electronics (ISCE 2006), St.Petersburg, Russia, 2006, 1-6.
- [4] Park, D. and Park, S., Composition Elements and their Characteristics of the Business Model for Providing Mobile RFID Service. In: Proceedings of the 9th International Conference on Advanced Communication Technology, Phoenix Park, Republic of Korea, February 2007, 1, 432-435.
- [5] Fielding, R., Architectural Styles and the Design of Network-based Software Architectures. Doctoral Dissertation, Department of Computer Science, University of California, Irvine, 2000.
- [6] Chandy, K. M., Charpentier, M., and Capponi, A., Towards a theory of events. In: Proceedings of the 2007 Inaugural International Conference on Distributed Event-Based Systems, Toronto, Ontario, Canada, June 2007. DEBS '07, 233, ACM, 180-187.
- [7] Perrey, R. and Lycett, M., Service-Oriented Architecture. In: Proceedings of the Symposium on Applications and the Internet Workshops (SAINT'03 Workshops), January 2003. IEEE Computer Society, 116-119.
- [8] Acharya, S., Franklin, M., and Zdonik, S., Balancing Push and Pull for Data Broadcast. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, Tucson, Arizona, USA, May 1997, ACM Press, 183-194.
- [9] Ballagas, R., Memon, F., Reiners, R., and Borchers, J., iStuff Mobile: Rapidly Prototyping New Mobile Phone Interfaces for Ubiquitous Computing. In CHI '07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 1107-1116, New York, NY, USA, 2007.
- [10] The EPCglobal Architecture Framework, EPCglobal Final Version of 1 July 2005, URL: http://www.epcglobalinc.org/standards/architecture/architecture_1_0-standard-20050701.pdf
- [11] JSR 248: Mobile Service Architecture, Version 1.0, Final Release, December 21 2006, <http://www.jcp.org/en/jsr/detail?id=248>