Date of acceptance Grade

Instructor

Architectural Solutions for Mobile RFID Services on Internet of Things

Martin Peter Michael

Helsinki, December 17, 2007

Master's Thesis

UNIVERSITY OF HELSINKI

Department of Computer Science

Tiedekunta/Osasto Fakultet/Sektion – Faculty/Section	Laitos Institution Department
Faculty of Science	Department of Computer Science

Tekijä Författare Author
Martin Peter Michael

Työn nimi Arbetets titel Title
Architectural Solutions for Mobile RFID Services on Internet of Things

Oppiaine Läroämne Subject
Computer Science

Työn laji Arbetets art Level	Aika Datum Month and year	Sivumäärä Sidoantal Number of pages
Master's Thesis	December 17, 2007	83 + 6

Tiivistelmä Referat Abstract

Mobile RFID services for the Internet of Things can be created by using RFID as an enabling technology in mobile devices. Humans, devices, and things are the content providers and users of these services. Mobile RFID services can be either provided on mobile devices as stand-alone services or combined with end-to-end systems. When different service solution scenarios are considered, there are more than one possible architectural solution in the network, mobile, and 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 thesis illustrates these ideas. It also shows how generally the solutions can be used in real world use case scenarios. A case study is used to add further evidence.

ACM Classification: D.2.11 Software Architectures, D.2.10 Design, D.2.12 Interoperability, H.3.4 Systems and Software, H.3.5 Online Information Services

Avainsanat – Nyckelord Keywords
Domain-specific architectures, Service architecture, Mobile RFID, Mobile RFID services

Säilytyspaikka Förvaringställe Where deposited
Kumpula Science Library, serial number C-

Muita tietoja Övriga uppgifter Additional information

To my parents, brothers, and grandmother

Acknowledgements

This thesis was written at the Connectivity Application Laboratory of Nokia Research Center in Helsinki. It was my pleasure to research an interesting and challenging subject. I would like to thank my employer for giving me the opportunity to carry out this thesis work.

I wish to express my gratitude to my instructor Dr. Jukka Viljamaa at the University of Helsinki for his encouraging instruction and supervision. I also would like to express my gratitude to the supervisors of this thesis, professor Juha Taina, and professor Inkeri Verkamo, for their advise and instruction.

I would like to express special thanks to my supervisors at Nokia, Dr. Kari Hjelt, and Dr. Mohsen Darianian, for introducing me to the research area of software service architectures, and for providing guidance, valuable inputs and encouragement during the whole process of working on this thesis.

Finally, I would like to thank my dear parents, brothers, and friends for their continued support and encouragement during the work on this thesis and my studies.

Helsinki, Finland, the 17th of December 2007

Martin Peter Michael,

Contents

1. Introduction

The internet has become the platform to launch Information Technology (IT) services for all kinds of industries. Network and software services are bound together by all means. Nowadays the network is seen as the computer and software industries are evolving to design IT solutions as services. Software architectural principles for creating software products and services have been also evolving. Software architectural principles like REpresentational State Transfer (REST), Event Driven Architecture (EDA), and Service Oriented Architecture (SOA) are used to create software services that are used by millions of people and businesses around the world. For example, REST has a proven record of being a successful distributed application design principle because the whole World Wide Web (WWW) is designed based on the REST principle [FiT00, FGM97]. SOA is widely used for creating both user centric and enterprise web applications and services. It has evolved to a new level of maturity when compared to other technologies available in the past for providing quick service solutions. All these architectural principles also complement each other to design new service solutions. The latest trend in the service solution area is that software is viewed as a "service agent" to the user. User-friendly and user-centric services are very much appreciated by the users of the services. Such services provide users the possibility to choose, design, and publish their own services without even having to master IT skills. Users are also given the possibility to manage and control their contents in different forms and they are able to impose their own policy over the use. Typical examples of such services are the blooming social networking services on the internet [BrD07].

Smart mobile phones are ubiquitous devices with desktop processing capability. Mobile web access has overtaken desktop web access. The new trend in the desktops is that the web is used as a common platform for many applications. Web is evolving to Ubiquitous web and Web 2.0. The web browser has become an application platform for most of the applications. There is no big difference between PCs of the 1990s when compared to modern smart mobile devices. The processing power and storage capacity is increasing everyday. Most of the applications developed for PCs are now available for smart phones too. Like Web 2.0, mobile Web 2.0 has arrived, and provides a platform for web services on mobile devices. Not only that, new devices and sensors are being integrated to smart phones. Smart phones are not just phones that depend on the

operator network for communication. They can use other radio access technologies and thus provide a way for creating cost-effective ubiquitous services everywhere. This is especially true as mobile devices are carried by users wherever they go. People are switching to smart phones and getting used to the services they provide through various sensors. Everyday interaction with "things" is unavoidable. As things, humans, and devices are interacting with each other, there is a huge communication gap between them. We need services that enable humans, devices, and things to interact with each other. Obviously, mobile devices are the natural mediators for hosting some of these services.

Radio Frequency Identification (RFID) technology is more than 50 years old. It is a mature technology. Recently, it has been getting more attention for two reasons. First, prices of RFID chips have dropped dramatically. Second, Integrated Circuit (IC) technologies have advanced in terms of speed and size. RFID technology has been used in many industrial applications. Initiatives like Near Field Communication (NFC) Forum, European Telecommunications Standards Institute (ETSI), Automotive Industry Action Group (AIAG), European Article Numbering (EAN), Uniform Code Council (UCC), International Telecommunication Union (ITU), EPCglobal, and others have been defining and standardizing RFID technology, network, interfaces, and architectures [Sei05, EPC05]. For instance, the main focus of EPCglobal is to create both a world-wide standard for RFID and the use of the Internet to share data via the EPCglobal Network. But it does not define the solutions specific to the mobile domain yet. Mobility adds more flexibility to RFID technology. A RFID reader can be manufactured in a single chip, so it can easily be embedded into a mobile device. Mobile devices with RFID readers have already arrived in the market (see NFC mobile phones [Nok04]). Mobile phones with UHF readers (which can read EPC class1 gen2 tags) are also under development. They are known as mobile RFID readers.

The Internet of Things (IOT) [ITU05] concept created interest in using RFID technology to enable even "things" to connect. Smart phones and humans are becoming intimate friends. Smart phones can be made into mobile RFID readers with some additional hardware. Mobile RFID services are essential on smart phones in order for humans, devices, and things to interact with each other. Mobile RFID services are services which enable humans, devices, and things to interact with each other using RFID tags. These services need to be carefully designed so that they can be easily used by humans,

devices and things to mashup services, and to federate new services along with others. The new service solutions should overcome future technological challenges, but they should also keep up with legacy systems.

RFID service solutions can be provided by innovating the existing architectural principles, software engineering principles, services on the internet, and mobile and RFID domains. Thus one can realize the end-to-end scenarios to create the framework for these services on IOT [ZCD04, NSM05]. Such services can be used by humans, devices, and smart or dummy things. For example, at the moment there are many research studies being conducted in pervasive computing, and service oriented computing based on these principles.

This thesis addresses a way to find the efficient and optimal solution among the available options when certain application needs are considered. As there are many architectural principles and software engineering principles which can be applied to domains like network, mobile device, and back-end systems, there are many possible ways to create innovative RFID service solutions. This thesis analyses a few service solutions from different perspectives and suggests the best possible candidate for the given scenario. Also it formalizes and generalizes the service solutions that can be used for similar kinds of use case scenarios. It also shows how architectural principles can be combined to create a solution that fits certain types of service scenarios. It classifies the service models and addresses the challenges in the mobile devices and network while designing service solutions. It also includes a real case study to add more evidence and to apply the theoretical principles. The case study explains the design of a service solution from beginning to end. It describes the problem, possible solutions, design, and implementation of the solution, the analysis of use cases, quality attribute requirements, architectural models from different perspectives, constraints in the technology and devices, and the evaluation of the solution.

The scope of this thesis is limited to mobile RFID service solutions on the mobile device side and for end-to-end systems. It doesn't go into the details of the RFID network infrastructure or the backend systems up to the protocol level. However it provides the network configuration alternatives. It doesn't explain the detailed design or

implementation when it talks about the architectural solution. Also none of the security issues are covered in this thesis.

The content of the thesis is organized as follows. Section two gives an overview about the topics such as the Internet of Things, RFID systems and mobile RFID services, which we need to know before reading the following sections. Section three highlights the theoretical part of mobile RFID service architecture solutions. It explains the state-of-the-art architectural styles and how they can be applied to provide mobile RFID services. It also explores alternative options of communication styles the services can use, important actors that interact with the services, mechanisms of information handling, and service types. Section four describes a concrete problem case and the solution implementation. And finally section five concludes the thesis.

2. Internet of Things and Mobile RFID Services

This section gives technical background information on the Internet of Things, RFID systems, mobile services, and mobile RFID services. Domain knowledge and understanding of the Internet of Things and mobile RFID services will help us to interpret the relevance and reasoning for the architectural solutions that we will discuss in the following sections.

2.1 Internet of Things

Internet of Things (IOT) [ITU05] is a conceptual vision to connect the dummy things (everyday things from tiers to tea cups), devices (mobile phones to microwave ovens, and computers), and humans of the physical world to the digital world of information processing to create a ubiquitous computing world using enabling technologies like RFID and sensors. It is a vision of the future where everything of value will be connected to share a record of their interactions with people, context, and other objects [GKC04, Ger99]. Things that are connected with the IT infrastructure systems can be identified automatically. The information stored on the connected things can be shared, provided, and processed automatically. This creates a world of ubiquitous environment. In this ubiquitous environment human, things, and machines can communicate with each other seamlessly depending upon the context. In order to do that the internet should be extended to things by some means. IOT is thus the extension of the internet to the next level, i.e., bringing the internet even to the dummy and real physical world of things. It will create new models for businesses and industries and take the world into a new era [ITU05].

The need for IOT is very obvious when we realize the communication gap between humans, devices, and everyday things. The amount of growth in the numbers of internet and mobile phone users creates an opportunity for using the information in everyday life services. There is also an enormous increase of information contents about the people, devices, and things. The number of services and users is growing every day on the internet. This is because there are rapid technological advancements in microchips, sensors, computers, networks, mobile devices, information processing software, and services. At the same time, there is a decrease in the size and cost of manufacturing all these devices. Telecommunication and computer data networks have also evolved to more than the internet. Network service providers are adapting the next generation network

(NGN) which provides a multi-service environment using multiple broadband technologies [ITU05]. The services are designed independent of the underlying technology layers in the NGN. The context-aware services provide information to the users depending on the context. The availability of information at the right place at the right time to the right context is the key value for IOT services.

The creation of IOT needs a lot of innovation from different fields of technologies. Converting the “dummy objects” to smart information providers in the ubiquitous world needs a huge effort. There is a need for a large database to store the information and network infrastructure support in order to connect the everyday objects and to handle the huge information flow [TPT05]. The database should be available in the internet at anytime to be accessed by any device, person or thing. Every object which carries a tag also carries the data on it. Thus the database is spread everywhere. Access to the data needs some security policies. When it comes to the network, there is a need to connect networks of networks ranging from private networks to corporate intranets. All the networks where the things, devices, and people are connected should be connected worldwide. IOT also needs a system of item identification. The ITU report [ITU05] identifies the technologies for item identification as RFID, wireless sensor technologies, smart technologies and nanotechnology. One or more combinations of the identification technologies can also be used to uniquely identify the items. Already some big companies identify their assets using RFID, sensors, and barcode identifications. With all the above mentioned, existing data, network infrastructures and identification systems, the services need to be created so that the contents can be exchanged and used seamlessly to realize the dream of IOT.

2.2 RFID Systems

One of the potential identification technologies proposed in the ITU report is Radio Frequency Identification (RFID) [ITU05]. RFID is an automatic identification method which uses radio waves to automatically identify tagged items such as people, animals or objects [Wei05]. It is similar to a bar code but uses a microchip and radio waves. An RFID tag, also known as an "electronic label," "transponder" or "code plate," is made up of an RFID chip attached to an antenna. Tags can be read from several meters away and beyond the line of sight of the reader. The most common method of

identification is to store a serial number that identifies a person or an object, but a RFID tag can also store some additional information [Wan06]. RFID systems use RFID tags for item level identification. We believe that the RFID system is the most relevant and cost effective technology for item identification services for IOT.

RFID systems consist of three main components. They are RFID tags, the RFID readers, and RFID software [ITU05]. An RFID tag or transponder carries the ID data which is a unique string or code. In addition, a tag can also store information contents depending on the size of its memory. A tag is tagged or attached physically to the object to be identified. The tag is an electrical device designed to receive a specific signal and automatically transmit a specific reply. It consists of a coupling element (such as a coil, or a microwave antenna) and an electronic microchip (less than 1/3 millimeter in size). Tags can be passive, semi-passive or active, based on their power source and the way they are used, and can be read-only, read/write or read/write/re-write, depending on how their data is encoded. Passive RFID tags take the energy from the electro-magnetic field emitted by readers. Tags use the transmitting frequency in kilohertz, megahertz, and gigahertz ranges.

An RFID reader or interrogator is a hardware device that is used to read the transmitted data from the tag. For reading passive RFID tags, the reader has to use extra power in electromagnetic wave form. The reader hardware consists of a bipolar antenna and a microprocessor chip. The bipolar antenna is used to transmit the signals and power to the tag. It is also sensitive to read the reflected signals from the tag. The microprocessor controls and runs all the reader related processes. The reader can be a dedicated handheld device or embedded in a mobile device or it can also be embedded in a wall (fixed readers). Compared with tags, readers are larger, more expensive, and power-hungry depending on their capacity. Reader hardware must also have the communication capability (wired or wireless). Readers themselves should be identified and authenticated by the RFID systems.

RFID software is a middleware that runs on the RFID reader. There are various services included in this software. At the lower layer level, a tag protocol service is used to communicate with the tags. Services like reading a tag, locking, writing, and killing a tag are included as the basic set. In addition, filtering, aggregating, searching, event handling, logging, and security services are included in the software. The middleware also

needs to communicate with the network and backend systems. It should process the commands it gets from a remote control system as well.

There are two types of RFID: near-field RFID and far-field RFID [Wan06]. Near Field Communication (NFC) RFID enabled phones have been manufactured a few years now and a plenty of NFC services are already working on NFC phones [Nok04, Sei05, FRD07 , Ruk06]. NFC readers read the tag within 5 cm range or sometimes need a touch. Far-field RFID covers a distance between three and six meters by fixed readers. When far-field RFID communication is considered on the mobile devices, there are extra challenges to be solved in the areas of power consumption, reading range, security, and the way of reading tags. Mobile phones with integrated far-field RFID readers are not yet commercially available. But they are expected to be launched at any time.

Choosing RFID for IOT as an enabling technology is rational because it has many engineering benefits. RFID is a mature technology and RFID tags are very cheap to manufacture nowadays. A tag costs only few cents when it is manufactured in bulk. Tags can be also reused. It is very convenient to read tags by sensors and devices like handhelds or mobile phones. Tags are already being used in logistics, tolling, manufacturing, and in other fields [BFM06, FLR07, LeK06, LNY06]. Most of the big manufacturers are using RFID tags in their products. Some of the large vendors and governmental organizations in some countries mandate RFID as a must for their supplier products. Also in Japan and South Korea, many people use the RFID-enabled mobile phones to use RFID services. So RFID is a proven concept and has the potential to be used for IOT.

2.3 Mobile Services

Mobile services are the features offered by mobile device software. Mobile device software makes use of the network, integrated hardware, device sensors, connected devices, storage, and multimedia applications to provide such features. Mobile services can be viewed as a commodity that provides services to the device owner. Mobile users are not interested in how the services are implemented in their mobile phones rather they are interested in what services the device would provide. They don’t see the mobile phone as a phone just to make a call, as much more.

A decade ago, mobile devices were very simple cell phones that were mostly used to make voice calls. Every year mobile operators introduced new services to make revenues from their network. Examples of such services are Short Messaging Service (SMS), Multimedia Messaging Service (MMS), Wireless Application Protocol (WAP), and mobile internet. Mobile devices have also evolved at the same time. PDA features and cell phone features were integrated into a single device. At the same time, processor speed, screen quality, power consumption, wireless radio, connectivity, and software have been greatly improved. New sensors and hardware devices have been also integrated. Thus the simple mobile phone has quickly evolved into a smart phone [Mic05] that has more or less the same processing speed that the personal computers (PC) had in the 1990s [Nok06]. People like to use these smart phones, since they don't like to carry many gadgets around. The sale volume of smart phones has crossed tens of millions. Demand for services on smart phones is growing every day.

Nowadays smart phones ship with some built-in services from the manufacturer itself. Additional services can be subscribed to from the network operator. Smart phone devices can also be integrated with third party service providers or system integrators. Services on the smart phone can be classified into three types. The first types of services are called device services. Device services don't need any services from the internet or mobile network to operate or they simply don't need the network connection at all. Examples of such services include calendar, alarm clock, Global Positioning System (GPS), Frequency Modulation (FM) radio, music player, voice recorder, and camera. Some sample applications which provide device services can be found from [NoS07]. The second types of services are called operator services. Examples of operator services are voice calls, SMS, MMS, fax, Push to talk, and broadcast. The third types of services are mobile web services. They need the mobile network and internet to operate. Examples of mobile web services are Voice Over Internet Protocol (VOIP) calls, email, instant messaging, internet browsers, blogs, news, online maps, and mobile podcasts. Google, Yahoo, Microsoft, Nokia and other big players provide free web services for smart phones irrespective of the device manufacturer [Goo07, Yah07, Mic07]. Some of these services run on thin-clients and others use smart clients which run on some common mobile run-time like Java.

There are also plenty of software applications developed for smart phones. Some of them are manufacturer specific and others are generic. The manufacturer specific applications can be run on mobile run-time platforms provided by the platform providers [NoS07, NSA07]. Examples of such platforms are Symbian, Nokia s60, Windows Mobile .Net, and Linux/C-C++. Figure 1 illustrates the layered architecture of the Nokia s60 platform. The lower layers provide different software services to the upper layers. Services are also grouped according to characteristics like base, telephony, security, and so on. The s60 platform also provides a way to integrate new device sensors via Bluetooth, and other wired links like USB. Additional services can also be created by implementing the services as plug-ins in the lower layers. Additional application-level services can be created on top of application engines and messaging. They can also be implemented as MIDP applications.

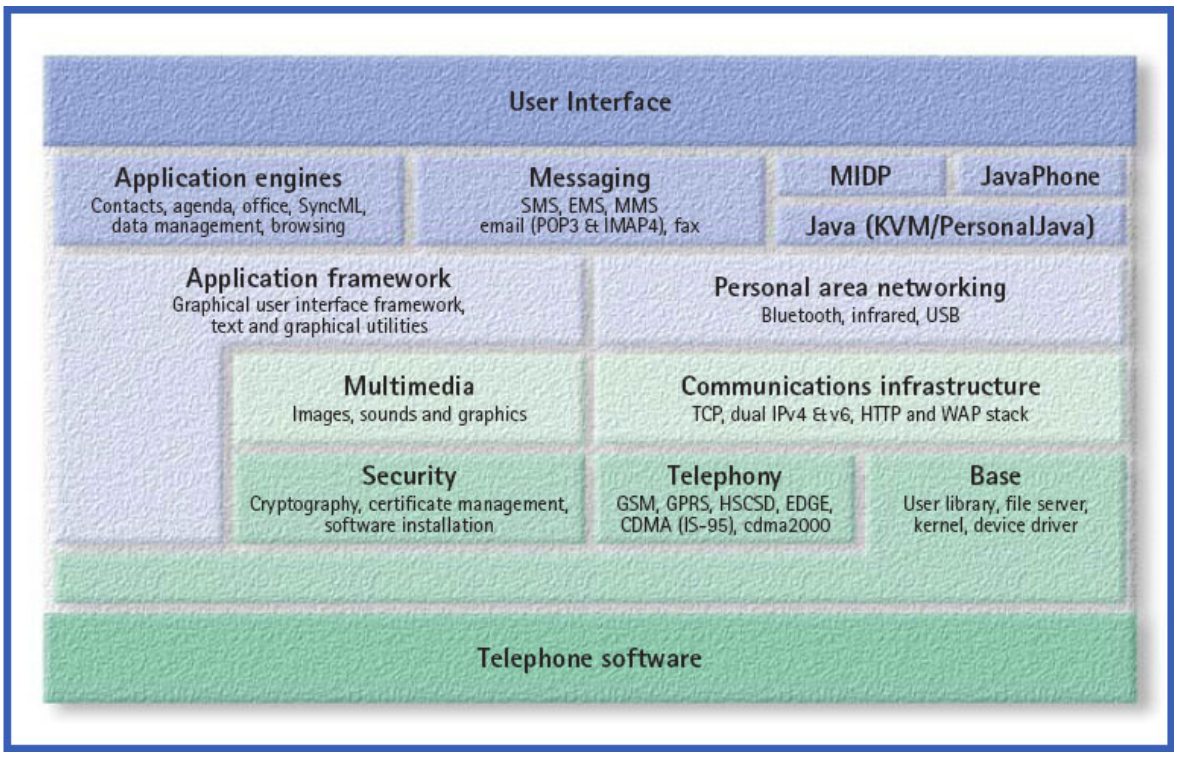

Figure 1 : A mobile terminal platform architecture – Nokia s60 [Nok02]

Generic mobile applications can run on devices which meet the required capabilities. Device manufacturers publish such capabilities so that the users and developers know what features are supported on the devices. For example Sun Microsystems has specified the Mobile Service Architecture (MSA) [JCP06, GuJ05] Java Specification Request (JSR) for this purpose.

Mobile manufacturers are innovating new ways to create services, both seamless and web or network-based ones. They try to integrate new devices and sensors which provide some new services. For instance, Nokia has integrated NFC readers in their mobile phones already since 2004 [Nok04]. The new version of the NFC phone now comes with services like exchanging business cards by touching, credit card payment (all kinds of cards can be integrated), and content download [Nok07]. Lately, the phones released on the market have built-in GPS. This is another key enabler to create context aware services on mobile phones [IbH04]. Also these latest devices can connect to the internet not only via UMTS or GSM network but also via WLAN or Bluetooth. They also have a higher processing capacity, Java Runtime, JavaScript, Flash, and other high-end rich multimedia application hosting capabilities. Thus they open a possibility for creating context aware, reusable, true interoperable services on smart phones. For IOT, these improvements are indispensable.

Mobile applications are transforming to mobile services on the web. Web 2.0 inspired the mobile domain to go for mobile 2.0. In [Fli06], mobile 2.0 is identified as having some key properties supporting the concept of Web 2.0. It is recommended that the mobile software applications should be created as mobile services. New mobile services should be able to be created from service mashups with loose coupling over Web 2.0.

2.4 Mobile RFID 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 [Sei05, KCK06, LeK06, PaP07, YLK07, HPL06]. Simply, mobile RFID services provide information on RFID tagged objects. Mobile RFID services use, combine, and mash up mobile services with RFID reader services and backend network infrastructure and database services. However, the network and backend servers are not always necessary to provide the services.

Mobile RFID services can be classified into two types. They are RFID based mobile telecommunication services and RFID based mobile web services. RFID based mobile telecommunication services are the services that provide information on RFID tagged objects over telecommunication network [Sei05]. In those services, the RFID reader-integrated mobile phone is treated as a tag and a reader at the same time. Examples

of the services are information retrieval, data transmission, automated messaging, voice services, device integration, presence indication (uses phone as a tag), and mobile payment (uses phone as a tag). RFID based mobile web services are the services that provide information on RFID tagged objects using other radio networks. In this case, the RFID enabled mobile phone is connected to the internet via other radio networks like WLAN, Wibree, or Bluetooth but not via the GSM or UMTS telecommunication network.

Mobile RFID is a completely different approach from the traditional RFID. In mobile RFID, the readers are mobile and the tags are fixed and not the other way around. There are some obvious advantages over traditional RFID. There are no connected wires needed and only a few mobile readers are enough to cover a large area instead of dozens of fixed readers [BiI05]. Moreover people always carry mobile phones with them. Smart mobile devices are intelligent devices with senses [RLC06].They can see, hear, identify, locate, and measure seamlessly. People come across the need to interact with things around them every day. By making things smarter, a ubiquitous environment can be created around people using all the senses the device has. Ubiquitous services can live in a ubiquitous environment. Mobile RFID services are one enabler to achieve this goal.

Mobile RFID services need a mobile RFID platform. This platform can be provided easily by using a smart mobile device. A mobile phone integrated with an RFID reader is called a mobile RFID reader. These are already available in the market. It is also possible to combine a UHF reader with NFC and other identification technologies like barcode to create a hybrid reader. Such combination creates a unique identification platform [RLC06]. In the future, a smart mobile phone with a combined identification platform would be a key enabler for IOT services.

Mobile RFID readers face many challenges as well. All the services currently running on fixed readers need to be ported onto the mobile RFID reader platform. In addition, mobility brings its own challenges. The Mobile RFID reader cannot cover the same reading distance as the fixed RFID reader range (3m to 6m). Power consumption in the mobile RFID reader is a serious issue. Also unlike the near-field RFID, far-field RFID can read tags without a physical touch. This adds extra challenges in identification, selection, and filtering the tags. The reading range should also be configurable depending on the context of the RFID application use cases.

Mobile RFID service architecture needs different service design approaches depending on the choice of the mobile RFID reader platform. For example if a NFC reader is used, the physical touch saves lots of service implementation when compared to the far-field RFID. This applies also for energy consumption, security, and other issues that we discussed. Far-field RFID services need extra overhead for selection, filtering, and aggregation of tags.

3. Mobile RFID Service Architecture Solutions

This section discusses the state-of-the-art architectural styles and how they can be applied to provide mobile RFID services. It also explores and considers alternative options of communication styles the services can use, the important actors that interact with the services, information handling, and the service initiation types.

3.1 Basic Use Cases

Any RFID service needs to use one or more of the four basic use cases. They are 1) Attaching information to a tag 2) Detaching information from a tag 3) Reading information from a tag and 4) Updating information to a tag.

When RFID tags are used for item identification, in addition to physically attaching a tag to an item, the information content should be attached to a tag in the digital world. Information content of an item can be either on a tag or on a device or somewhere in the network. But irrespective of where the information content is stored, the attaching procedure must be followed. The use case is usually executed only once for an item unless there is a need to replace a tag. Also the attaching procedure must be handled by someone who is authorized. A practical example will be depicted in section 4.2.1.

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 when an item's life time is over.

Reading information about an item is the most common use case. This is because anyone who is interested in knowing about an item would need to read it. One could also browse through the services it provides. Today most of the mobile RFID application services are built based on the 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.

3.2 Representational State Transfer

REpresentational State Transfer (REST) is a software architectural style of the WWW introduced by Roy Fielding [Fie00, FiT00]. It is a composition of two architectural styles. It employs the layered, code-on-demand, client-cached-stateless server (itself a composition), and uniform interface principles. It defines a uniform interface with constraints. All connectors of the system must conform to the REST interface constraints. The constraints are: identification of resources, manipulation of resources through representations, self-descriptive messages, and hypermedia as the engine of the application state. In WWW, these constraints are realized as Universal Resource Identifiers (URI), Hyper Text Transfer Protocol (HTTP) request & responses, HTTP headers with Multipurpose Internet Mail Extensions (MIME) types and URIs embedded in a media type (which are in turn, potentially dereferencable via HTTP) respectively.

The REST principle was used to build the WWW, the largest distributed system ever built. The main challenge while building the WWW was to find a universal way to communicate between different devices across the world. This was solved by the REST architecture style by defining resources, representations, and interfaces that every device can understand.

According to REST, resources are concepts. Devices find the resources by their Universal Resource Locators (URL). URLs are the nouns. So any device can talk to another one using the URL noun to identify the resource. Examples of such resources are images, text, sound, video, and services. All of them can be accessed by URL nouns. And a representation is the way the resources are presented. A web page is the representation of a resource. Representation is mainly for humans. The four verbs that are used on the nouns are GET, PUT, POST, and DELETE. Interfaces are generic enough to extend them and provide a unique way to access the resources. The protocol used should be stateless, cacheable, and layered. This is the principle used in HTTP for WWW [FGM97]. Many wonderful web applications and services have been developed using just these few principles.

Most of the REST principles can be applied to mobile RFID services as follows. REST suggests providing a distinct URI for each resource we wish to expose. Every physical thing on this earth is a resource and it can be exposed by its URI. The

physical thing can be physically tagged with RFID and can be uniquely identified. REST suggests using nouns in the URI. So we can use the tag ID in the URI. To get the services a thing provides, we can use verbs. Methods that map to GET should not change any data according to the principle. We don't want anyone to change the information about the tagged things. However, we want to provide access to information and services about the things. And by using the hyperlinks in the responses, we can atomize the paring of the information about the items and services using standard browsers on the devices. Finally, the services can be made stateless. But if sometimes the state information is required for services, it can be maintained as a resource. Thus we could use the REST architecture style for providing mobile RFID services on IOT.

RESTful web services treat the services as resources. They can be implemented using HTTP methods. HTTP methods can be used to describe create, read, update, and delete (CRUD) actions to be performed on contents. CRUD operations are generally performed to maintain content information in the databases. When CRUD operations are used to maintain content at the backend databases and HTTP methods are used to implement the services, a mapping is possible as in Table 1 [Sun06].

Action	**SQL**	**HTTP**
Create	`Insert`	`PUT`
Read	`Select`	`GET`
Update	`Update`	`POST`
Delete	`Delete`	`DELETE`

Table 1 : Relationships between SQL and HTTP Verbs [Sun06]

Mobile RFID services created on end-to-end systems use internet or some other private network like intranet as a medium to access the information content from a central database. They can also use other legacy systems like (Enterprise Resource Planning) ERP or Systems Applications and Products (SAP). These services can be referred to as mobile RFID web services as they use web as the primary medium for communication.

Based on the case studies we have conducted, we conclude that the REST principle fits very well with mobile RFID web services. Any mobile RFID web service falls into one of the four basic RESTful services as listed in the first column of Table 2. The case study will be described later in section 4.

When we think of services for real world things, there would be a huge amount of data flowing across the web, if WWW was used for the mobile RFID services. The bandwidth is very important and needs to be limited. REST is particularly useful for limited-profile devices such as PDAs and mobile phones, for which the overhead of headers and additional layers of SOAP elements on the Extensible Markup Language (XML) payload must be restricted [Sun06]. REST style also easily enables service mashups. Rather than starting from scratch, services can be exposed with XML and consumed by HTML pages or other web services without significantly refactoring the existing web service architecture. New services can be created even without much in-depth knowledge about software engineering or programming languages [PaH05]. REST style systems are highly scalable and highly flexible because they can take full advantage of the scalability features of HTTP such as caching and proxies as well.

Basic RESTful mobile RFID services for IOT	**HTTP verbs**	**Parameters needed**
Tagging/attaching info on a physical thing	PUT	The URI of the service, the tag ID, and the item information
Information retrieval from the tag id	GET	The URI of the service and the tag ID
Information update, status update, etc	POST	The URI of the service, the tag ID, and the item information to be updated
Detaching the information from the thing	DELETE	The URI of the service and the tag ID

Table 2 : Basic RESTful mobile RFID services

3.3 Service Oriented Architecture

Service Oriented Architecture (SOA) [PeL03, MLM06, FuM06] is an architectural style whose main emphasis is on the loose coupling among interacting software agents or components. A service is a unit of work done or a function by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents or components on behalf of their owners.

Service-orientation is a design paradigm that specifies the creation of automation logic in the form of services. It is applied as a strategic goal in developing a SOA. Like other design paradigms, service-orientation provides a means of achieving a separation of concerns. Services can be combined or distributed over the network. They communicate with each other by passing data from one another, or by coordinating an activity between them. A service in SOA has its own characteristics. The granularity of service is defined by the service consumer depending on its needs.

The basic SOA building blocks are the service provider, service broker, and the service requestor Figure 2 [Sun05]. The service provider publishes the interface to the services, but it keeps some security constraints. The service broker is also known as service registry which enables the service interface to be available for any service requestor. For example, the Universal Description Discovery and Integration (UDDI) specification defines a way to publish and discover information about Web services. Service requestors or clients locate the services in the service registry and bind the service provider. Once the binding is done, if the constraints are met the service is accessed. SOA uses the find-bind-execute paradigm. Web services can be used to implement a SOA.

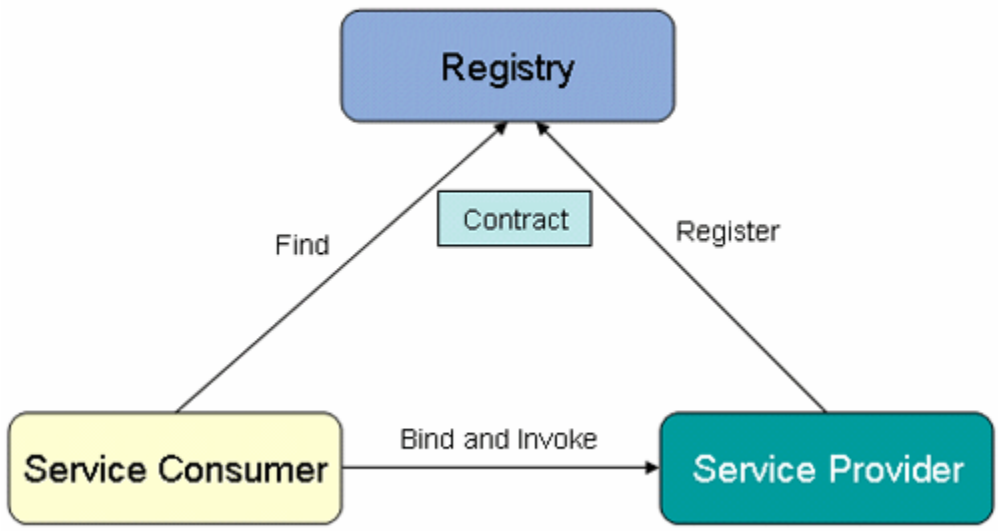

Figure 2 : SOA's Find-Bind-Execute Paradigm [Sun05]

In mobile devices the service consumer can be easily implemented. The other blocks such as the service registry and service provider are harder to implement as the device is constrained to limited resources and processing power. That's why they are usually implemented at the backend servers which can be accessed via the internet.

HTTP support is implemented on almost all Java Mobile Environments (J2ME) [Sun04]. An XML parser is typically required to parse data returned by web service invocations. XML parsers are available for inclusion on all Java ME environments, either via the J2ME Web Services Specification optional package or via open source projects like kXML. REST-based services are loosely defined and require custom coding to marshal data (often embedded within the URL of the service invocation), invoke a service (a simple HTTP call), and unmarshal the response (often an XML document). Simple Object Access Protocol (SOAP)-based services have more complicated semantics and require additional support beyond HTTP communication and XML parsing.

It is also possible to host mobile RFID services on the device itself [WiD06]. In this case the device would act as a service provider. However, the service availability on mobile devices is not reliable because of the device constraints. In the future this trend will probably change. Every mobile device would host its own service and would give universal access interface to the clients. There are plenty of use cases for this concept. Device-to-device services can be enabled by just using two mobile RFID readers or one mobile RFID reader and another mobile device. At least one of the devices would have to host the mobile web server to provide the services via the web. By this way many personal mobile RFID services could be created for the IOT.

So the SOA principles can be used to provide mobile RFID services on the device by mashing up the existing services of the mobile device and the RFID services. Mobile devices can be used as a service consumer or service provider. The mobility and the device sensors add an extra dimension to the services. SOA and REST can be combined to create service solutions. However, the mobile device constraints must be taken into account while designing service solutions because these constraints directly affect the quality attributes of the provided services.

3.4 Event Driven Architecture

An Event Driven Architecture (EDA) style stresses on the events that the software systems exchange between themselves [CCC07]. Events are produced by event producers. They publish the events to an intermediary event manager. Event consumers subscribe to the event manager for the interested events. When the event manager receives

the events they are forwarded to the respective event consumers. Event producers and event consumers can be software agents or processes.

The EDA style can be used to provide mobile RFID service solutions along with SOA. SOA services can be triggered by events [Han05, Sli03]. Any service in the mobile RFID device can be programmed to create events. RFID services running on the device can produce events. For example, when the mobile RFID device is brought in close proximity of a smart thing, the RFID reads the tag ID and creates an event. These events can be processed at the device with some intelligent mashup services to create new meaningful mobile RFID services. Automatic RFID services can run on the mobile devices or remote mobile devices based on specific events. Services can be designed in a way that they can be easily configured to run automatically without human intervention. The events can use the web service bus to travel between the publisher and subscriber of the services.

3.5 Push and Pull Technologies

The way in which the communication style is used to exchange the contents between the service providers and service consumers can be classified into two categories: push technology and pull technology [PHJ02, FrZ98, AFZ97, BDK02]. In pull technology, the client pulls the contents from the server. The communication is initiated by the client. The client sends a request to the server for the contents or services and the server responds with appropriate contents back to the client. The pull technology is very popular and widely used. For example, REST uses a pull-based concept to browse the contents of the web. Internet users use the browser clients to access the information by clicking or typing the URLs. When they do this a request is submitted to the server which is listening on the internet and it will respond with proper content. Thus the content is pulled by the client from the server. Web browsing, RSS feeds, and web emails, are all examples of pull-based communication. Post Office Protocol version 3 (POP3) and Internet Message Access Protocol (IMAP) are pull protocols used in email client services.

The push technology works the other way around: instead of the client requesting the content or services, the server itself initiates the communication to the interested client. For example, a mobile phone can receive SMS text at anytime if the network coverage is provided and the SMS service is subscribed. MMS, WAP Push are other examples of mobile push communication. In internet, Instant Messaging (IM) and

some peer-to-peer programs also allow pushing files. In these cases, the sender initiates the transfer rather than the recipient. Unix to Unix CoPy (UUCP) and Simple Mail Transfer Protocol (SMTP) are push protocols used for emails in the servers. The IMAP IDLE can be used for push email service on mobile phones [Iso07], even without mobile network services if the device supports WLAN.

Mobile RFID services can be implemented as push-based or pull-based services depending on the use cases [CuJ02]. Pull-based services can use the REST or SOA principle to request the service providers in the occurrence of some events on the device side. Push-based services can use any push-based solutions currently available like IMAP IDLE for email clients, polling to make the pull-based to push-based services, mobile web server, keeping the HTTP connection alive, or by combining any push-based mobile services like SMS, MMS, WAP Push along with pull-services. There is always a price to pay if the mobile RFID services have to be available in real time. The extra cost has to be paid to the mobile operators to use any mobile push services or to provide extra infrastructure support on the internet. But unfortunately for some use cases, this is unavoidable.

There are also other solutions to provide push services. In case of using a mobile web server on the smart RFID-enabled phone to provide mobile RFID services, we need additional infrastructure support on the internet such as an extra gateway [WiD06]. The gateway keeps track of the devices and maintains the address mapping table. For use cases which apply for the devices on the same subnet, the mobile web server needs no extra gateway at all. Thus it creates an interesting scenario for peer-to-peer mobile RFID services. Using dynamic DNS is another solution to keep the DNS of the smart phone to provide access to mobile RFID services. By doing this, the device can be accessed by any other service on the internet by the DNS name.

Pull-based mobile RFID services are initiated by the device itself. For example when a RFID tag is read by the device, the service would display information about the corresponding item. For such services, the REST principle can be used to browse the contents which are provided by the server somewhere on the internet.

3.6 Things, Devices, Humans and Services

In the service realm the three important players are things, devices, and humans. Most of the services on the internet are targeted to human and human-to-device interactions. Some of the services are used for device-to-device interactions. There are services that are automatically used between the devices already. When we think about adding things to the existing services, we need to also stretch our thinking to ask how would the things-to-things communication take place? What kind of services would a thing provide to another thing? But at present, most of the services are designed to provide services from things to devices and humans. In this thesis, we limit our focus to how a mobile device with RFID reader capability could be used for enabling the services for the main players like things, devices, and humans. Before designing any service solutions, it is important to identify the players for the service and categorize it. This can be easily done once we know the service use cases.

Table [Table 3] shows how a service can be placed in one of the nine possible combinations when we consider the main players: thing, device, and human. We focus only on the three categories of services because "thing" is the common denominator for such services. Thus the considered IOT services are thing-to-thing services, thing-to-device services and thing-to-human services.

	Thing	Device	Human
Thing	Thing to Thing services	Thing to device services	Thing to human services
Device	Device to Thing services	Device to Device services	Device to Human services
Human	Human to Thing services	Human to Device services	Human to Human services

Table 3: Service categorization

Thing-to-thing service cannot be enabled by just using RFID tags (passive or even active) on Things. This is because even RFID tags cannot perform any actions independently of other supporting technologies like sensors. Such systems would need additional processing power to process the program logic. RFID tags with sensor services are beyond the scope of this work.

Thing-to-device services are the services a thing provides to the device. For example, a list of services a thing would provide can be stored as a URL in the tag, as the tag cannot store large amount of information. When a device reads the URL, the web service or any other RFID service can open the URL address where the services are and use those services. Note that no human interaction takes place on the device except that the device or thing is brought closer to the range of an RFID tag and reader by the human. Another example is to get a WLAN access point for a mobile device to connect to the internet when the mobile device is brought close to the RFID tag attached to the access point. When the device reads the tag, the information on the tag would provide the access point name and WEP key. The device would start connecting to the internet using the access point. A password, key or an encryption key code can also be stored on the tag. It is also possible to encrypt the stored values so that not everyone can access the information.

Thing-to-human services are the services the thing provides to the human. These services definitely must use a device to communicate to the thing. The mobile phone plays a major role for these kinds of services. For example, while shopping the user wants to know where the product is from or what the history behind it is. An extra network and back end infrastructure support is needed for supporting these kind of use cases.

3.7 Enabling the Services on the Mobile Device

A smart mobile device has interfaces in the hardware to connect new hardware devices and sensors or it can be easily made to simulate one interface [BMR07]. Smart phones usually have hardware interfaces such as USB, serial interface, or Bluetooth. Using one of the interfaces, the RFID reader can be easily connected to the phone hardware. The RFID hardware board requires a power supply and a communication interface which is used to send and receive the control signals and data to and from the mobile device.

The smart phone software platform usually provides software interfaces to connect other hardware devices or sensors with the phone. There is also a possibility that the RFID hardware can be connected to the phone's software via some UDP port. The software connection to the communication varies depending upon the implementation of the particular software device platform. The phone software platform also needs some driver services specially written to talk to the new hardware.

Modern mobile phones are equipped with dual processors or single processors. They are able to perform the functionalities of a Personal Digital Assistant (PDA) and a cell phone. They have low level hardware interfaces which are managed by operating system drivers. Operating system services are running at the lower layer level on the system drivers. Some vendors provide the access to those low level services to application developers. Nowadays in smart phone devices, there are also some vendor-specific services which are closed to the public and some open universally accepted services. An example for open services is specified by Java Specification Request (JSR) [JCP06]. JSRs are formal documents that describe proposed specifications and technologies to be added to the mobile Java platform. Vendors that have implemented the JSRs specify if their devices are supporting those services. By looking at the JSR specification for a mobile device, one can decide how to design the application services for those devices.

RFID reader hardware can be embedded in a smart phone as other hardware like GPS or Bluetooth and can be interfaced via Universal Serial Bus (USB) or other ports which the device provide. The RFID reader chip can be controlled by the RFID driver software. This can be implemented as an operating system service. The driver services should be layered very thin so that if we change the RFID hardware in the future, the whole driver software need not be rewritten. The RFID reader services use the RFID reader hardware which is connected to the smart mobile device. Basically the following elements are needed to create a RFID reader platform on the smart phone along with the software services: RFID Reader Hardware, a hardware connection between mobile device hardware and RFID reader hardware, a software driver for RFID hardware and a Communication software interface.

We can apply the SOA principle with the layered architecture to define the software services. We need to layer the services at different levels in order to reuse the services [Boh06], and to be flexible for future changes in the RFID hardware or firmware. The proposed service architecture for mobile RFID services at the device side is depicted in Figure 3.

The top layer in the architecture hosts all the types of mobile RFID application services. We have identified three types of services based on who initiates the

services. The human-initiated mobile RFID application services are the services which are designed for specific use case applications. The user starts the application services to use the RFID services along with other services in the smart mobile device. The Thing-initiated mobile RFID application services are the services which run when the mobile device reads a tag. These services are pre-registered on the device to run on specific things or events produced from things. Device-initiated mobile RFID application services are the services which are started by the mobile device itself. The service initiation here is neither by the device user nor by the thing but from other services or events from the device. For all the types of services we use EDA combined with SOA and layered architectural styles.

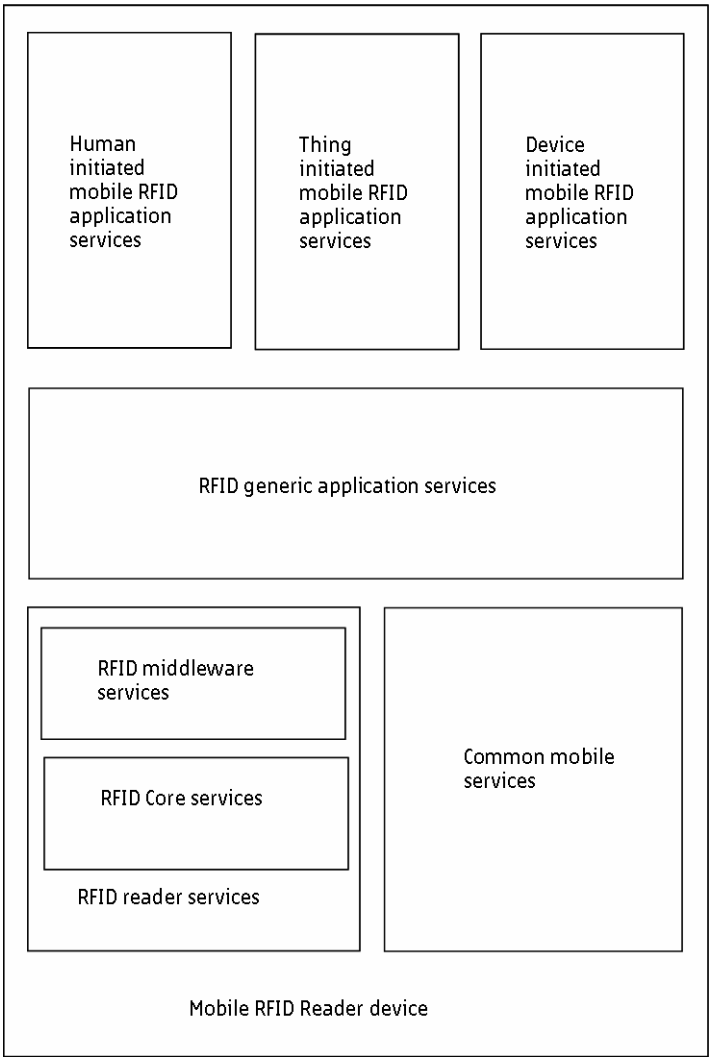

Figure 3: Mobile RFID service architecture at the device side

The RFID generic application services layer consists of services such as attaching and detaching a tag as well as reading and updating information. These services

are not specific to any application, but they can be generally used to compose specific application services. Also these services involve more than a trivial task at the lower layers. For instance, in order to perform an attaching tag service, a few services at the lower layers are needed, such as reading the ID of the tag, fetching the information to be attached, presenting the information on the UI, performing the attaching operation at the backend, updating the database, parsing the information, and verifying and displaying the error information are all needed.

Common mobile services are services which are commonly available in 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. Connectivity service enables the possibility to connect to any wireless network or device using Wireless LAN (WLAN), Bluetooth, or wibree. Networking services provide the way to handle standard protocols. Radio access services provide the way to connect the device to packet and circuit switched network over different carriers like General Packet Radio Service (GPRS), Global System for Mobile communications (GSM) or 3rd Generation (3G) radio.

RFID Reader services consist of RFID middleware services and RFID core services. RFID middleware services use RFID core services. RFID middleware services are analogous to the services that are provided by NFC devices as in [JCP05]. However, when far field RFID is considered, there are additional challenges such as the reading distance, differentiating one or more tags from multiple responses, and power consumption issues. These challenges can be solved by providing additional high level services in the middleware. Thus RFID middleware services also include high level services such as filtering, aggregating, grouping, counting, differential analysis, decoding, providing data, configuration, and power control [EPC05].

The filtering service eliminates some Electronic Product Codes (EPC) according to their identities. Also filtering eliminates EPCs except a specific object class. Aggregating service aggregates EPCs over time intervals. Aggregating eliminates duplicate reads within a certain time interval. The grouping service summarizes EPCs within a specific object class. The counting service reports the number of EPCs rather than the EPC values themselves. The differential analysis service reports the EPCs that have

been added or removed rather than all read EPCs. The decoding service decode the data into various formats. The data providing service provides data for multiple clients.

RFID core services include basic services like reading a tag, locking, writing, and killing a tag. The core services use the tag protocol to talk to the tag. For instance, reading a tag here means that the service reads a sequence of bits from a RFID tag.

3.8 Service Types

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

3.8.1 Service Solutions Based on Information Storage

RFID service information content can be stored locally on a tag or on a mobile device or somewhere in the network. We introduce two kinds of services based on the location where the service information content is kept. They are a) Stand-alone services and b) End-to-end services. Stand-alone services rely on a tag or a mobile RFID reader for information content. End-to-end services rely on the network and backend systems for information content. These two types of services can have a few solution alternatives. They will be explained in sections 3.9 and 3.10 in detail.

An RFID tag is capable of storing some information by itself, so it is possible to store information about a thing in the tag [DMS07, LeK06]. Such information can be used to provide mobile RFID application services. The main advantage of this approach is that there is no need for network access for the mobile RFID application services. But there are also drawbacks to this approach. Anyone can access the information on the tag; it thus creates a potential security risk. The security of the content in the tag can be protected by encrypting it. While there is some change in the information, the centralized update is not possible. Also tracking and monitoring the thing is difficult.

Instead of storing the information on the tag, it can be stored on the mobile device as the device memory is large enough in smart phone devices. The main advantages of this approach are that there is no need for network access and the performance of the service is better than the previous approach. Moreover the device can also store more information than the tag. However, this approach has all the other disadvantages of storing the information on the tag. They can be overcome by providing access to the network to

synchronize the information between the phone and the central database. In this case, the drawback is that the information is duplicated in two or more places.

Information about the things can also be kept in the central database which resides somewhere in the internet or intranet [DMS07]. This way the information is centralized. It is also possible to access the information about the thing from anywhere and on any device at anytime. The main advantage of this approach is that it is possible to get the central update about the information and if there is a need to evolve the data model it is possible as well. Also, the tagged things can be easily tracked and monitored and integrating the mobile RFID services to other business services is easy. The only drawback is that there is a need for quick network access. If the network fails the entire system of services won't work. The security mechanisms for distributed systems need to be applied as the network access is a security risk for attack.

3.8.2 Service Solutions Based on Service Initiation

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

HIS are designed for specific use case applications. The user manually starts an application service to use the RFID services along with or without other services in the smart mobile device. A typical example is a user trying to read information about an item in a shopping mall using her mobile RFID reader. TIS runs when a 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 a security guard near a door can be created by identifying a tag on the door by her mobile RFID reader device. Whenever she visits the door, a mobile RFID service wakes up and runs to fulfill the task. DIS is started by a mobile RFID 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 for TIS service is a scheduled service which runs on a mobile RFID reader to read the nearby tagged things.

Human Initiated Service Solution:

In this solution, the human (user) does the service initiation rather than the device or thing. The typical steps are depicted in Figure 4. The human starts a mobile RFID service application on a mobile device (1). Then the device reads a tag on a thing (2) and the tagged thing responds to the device with a tag ID (3). The device provides the service if there is no need for information from the backend. Otherwise, if information is on the backend, the mobile device contacts the backend server via the network (4) and the server responds with the necessary information (5).

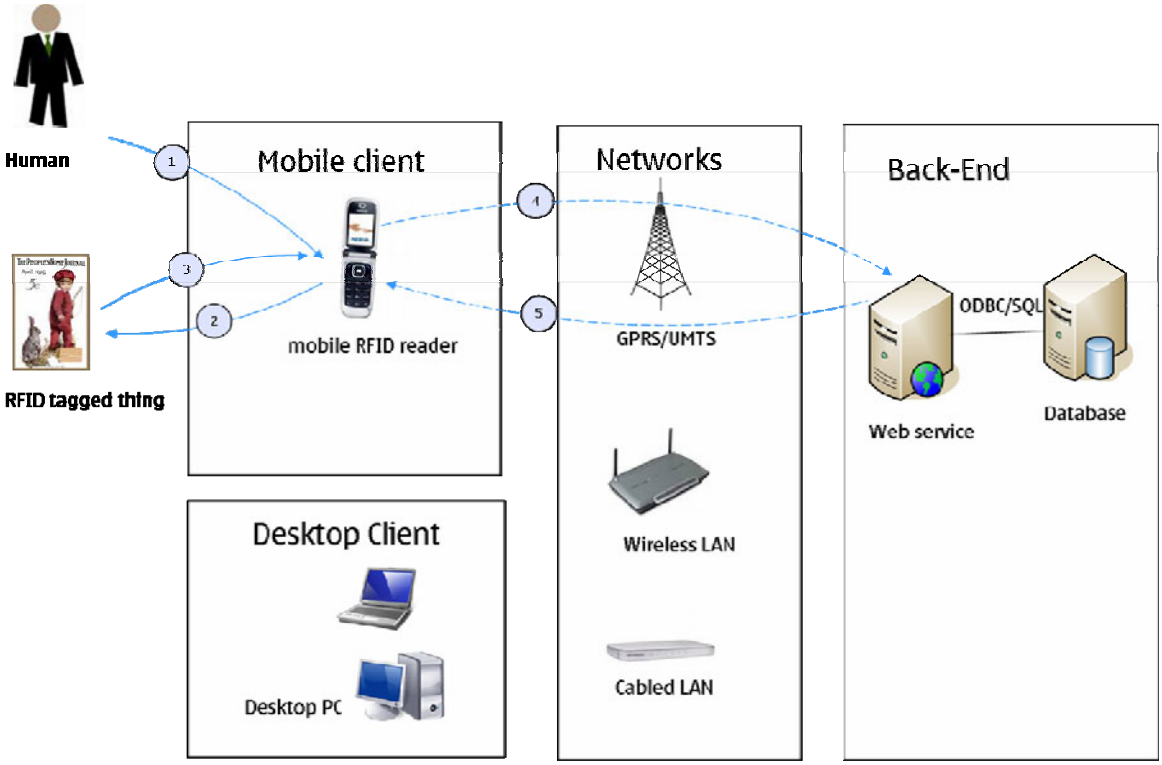

Figure 4: Human initiated service solution

The network infrastructure and the backend support are not always essential for these kinds of services. It mainly depends on where we would like to keep the information. Steps (4) and (5) can be ignored for some application services.

Thing Initiated Service Solution:

In this solution, a thing initiates the mobile RFID service on the device. The service is registered on the device registry for running during the occurrence of a specific event or events. Figure 5 depicts the scenario. A tagged thing responds to the device when

the device and the tag are coming into the reading range (1). A tag coming into the reading range of the device is recognized as an event. The mobile device supplies information for the service if it has the information. But, if information is on the backend, the device contacts a backend server via the network (2). And the server responds with the necessary information (3) which then will be displayed on the device. For these services, the EDA principle is used for service initiation and REST can be used if there is a need to contact the backend server for information.

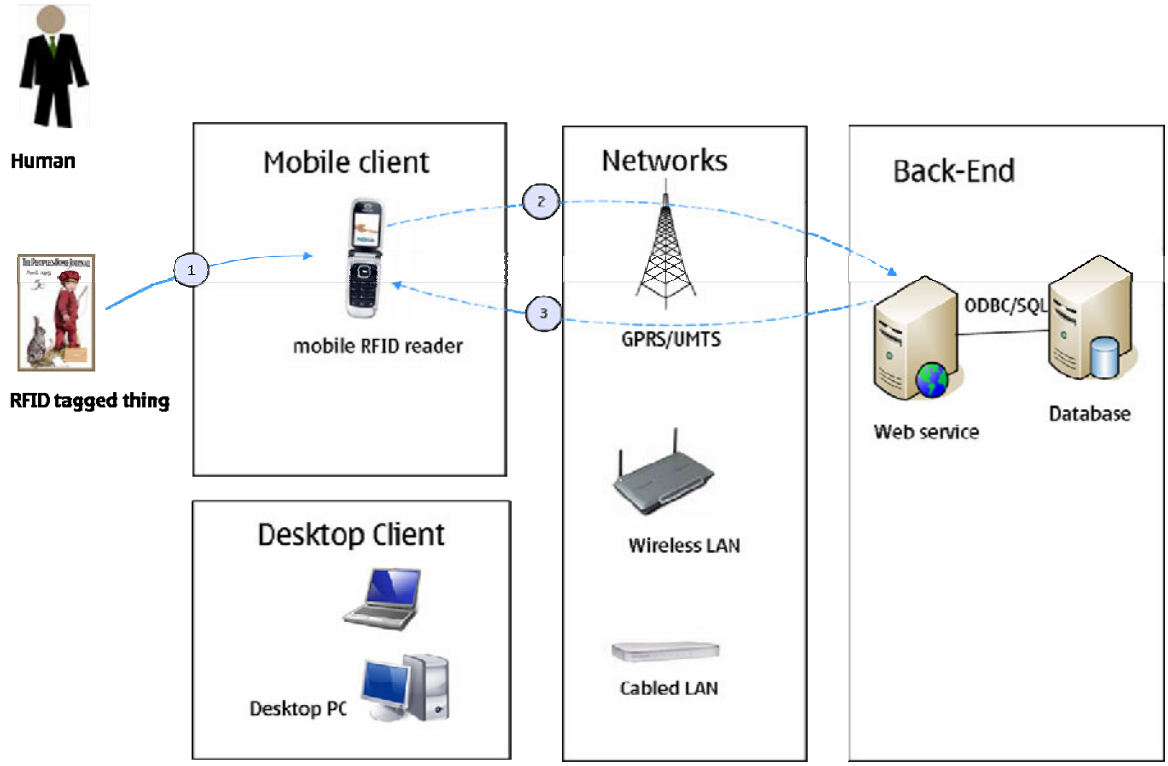

Figure 5: Thing initiated service solution

Device Initiated Service Solution:

In this service solution, a service is initiated by a mobile device or server or another computer in the network. But the service runs on the mobile RFID reader device irrespective of the source trigger device. An example is depicted in Figure 6. Step (1)a or (1)b initiates the service. In (1)a, a service is triggered by the mobile RFID reader itself. This is possible if other services, applications or sensors are used as a source trigger. On the other hand, in (1)b, a server or device in the internet can initiate a service on the mobile RFID reader device using a push technology. In either case, the mobile RFID reader device

reads a tag on a thing (2). The tag responds to the device (3). The device provides a service if it has the necessary information. Otherwise, if the information is on the backend, the device contacts a server via the network (4) to fetch information. The fetched information is finally displayed on the device. EDA, push technology, and REST principles are used for these kinds of services.

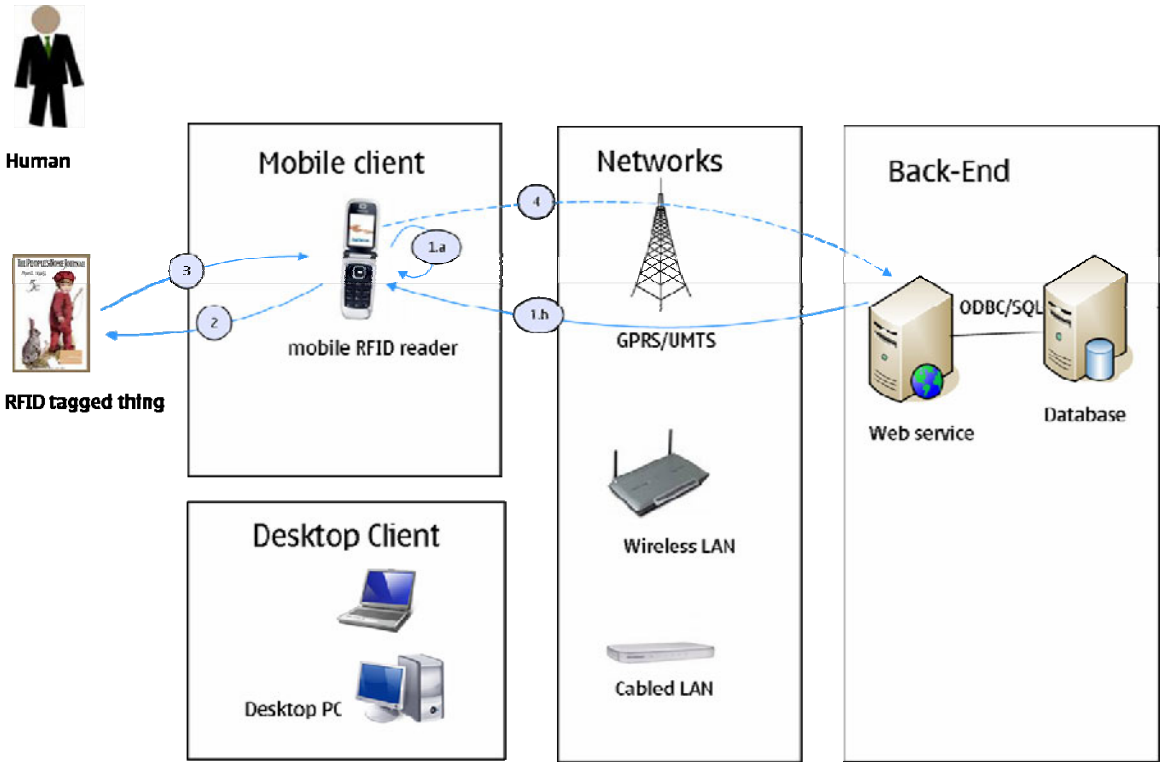

Figure 6: Device initiated service solution

3.9 Stand-Alone Service Solutions

Stand-alone service models can be classified based on the location of information content, and the location of services. We propose three models: a) content and services on mobile device b) content on tag and service on device and c) content and services on tag. The three types are illustrated in Figure 7, Figure 8, and Figure 9 respectively.

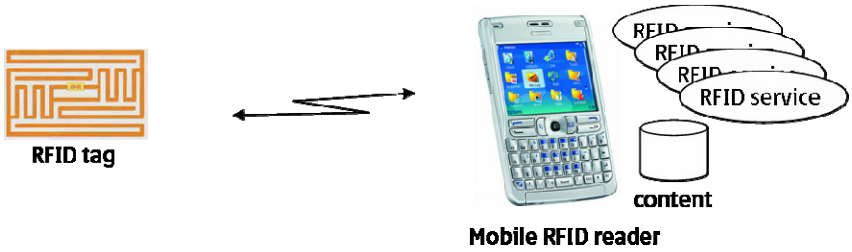

Figure 7: Content and services are kept on mobile device

The first solution can use inexpensive EPC class1 Gen2 tags. In this case, as the storage of the tag is limited, only item IDs are stored on the tags. The RFID services and content are hosted on a mobile device. When HIS is considered, the user brings a mobile RFID reader near to a tag. A RFID reader service which runs on the mobile device recognizes the tag ID and provides the appropriate information content.

Figure 8: Content on tag and services on mobile device

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

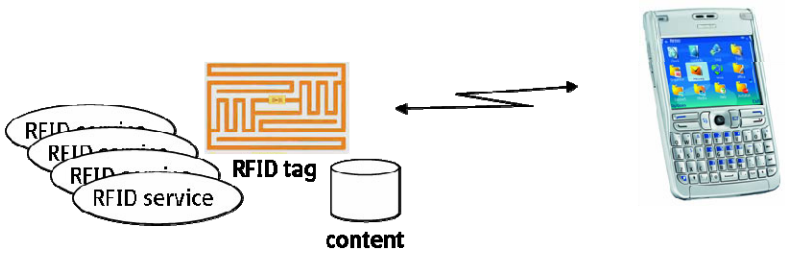

Figure 9: Content and services are kept on tag

In solution (c), in addition with information content, the RFID services are also stored on a tag. Thus it distributes not only the content but also the RFID services on things. A thing which is attached with a tag can thus provide data and services about itself. Moreover, any capable mobile RFID reader device can also use the services on the fly. But security measures must be enforced both on data and content in either direction. This is because a mobile device needs to know if it can trust the service stored on the tag and vice versa. This knowledge can be achieved by storing a certificate on a tag which provides such services. For example, a Java binary file that implemented a service can be stored on a tag. When a mobile RFID reader reads the tag, the certificate is loaded by the reader. If it is authenticated, the binary executable is then loaded on the device and installed. After that by running the service on the binary file, data on the tag will be interpreted.

Stand-alone service solutions can be implemented on an interoperable platform like J2ME or a native device platform. For example, in J2ME, an application management system (AMS) is providing the services for registering application services for events. Also by providing new JSR for UHF services one can create mobile RFID application services that can be ported to different mobile RFID readers. But it should be noted that the RFID reader services must be implemented on a device specific platform.

For all three solution models, there are some common services a mobile device should run. Such common services are security and data handler. Security service varies depending upon a solution. Data handling service interprets the data from a tag to a meaningful content.

3.10 End-to-End Service Solutions

End-to-end services use end-to-end systems to provide mobile RFID services. Elements involved in end-to-end systems are backend servers, database servers, networks, mobile RFID readers, and RFID tags. There is more than one possible alternative for network configuration. A particular alternative has to be carefully chosen after analyzing possible alternatives. This is because some major problems of the system can be solved by just selecting the proper alternative [KLY06]. Sometimes it is also helpful to add an extra network element like a gateway to solve a problem. A practical problem that was solved by selecting a proper network alternative will be shown in section 4.5.

We introduce two solutions for providing end-to-end mobile RFID services. They are a) Thin-client based and b) Rich-client based. Figure 10 and Figure 11 illustrate these solutions. A thin-client based solution uses the web browser as a platform for providing RFID services. The services which run on the web browser need some support from lower layers like transferring data from a mobile RFID reader to web browser and vice versa. We can achieve this by implementing some adaptation services which acts like a glue. The content for these services can be stored somewhere in the network.

A rich-client based solution uses a separate stand-alone application which is not network dependent. If the network is not available, still the services will work by using local content stored on a device. However, periodically data should be synchronized between the network and a device.

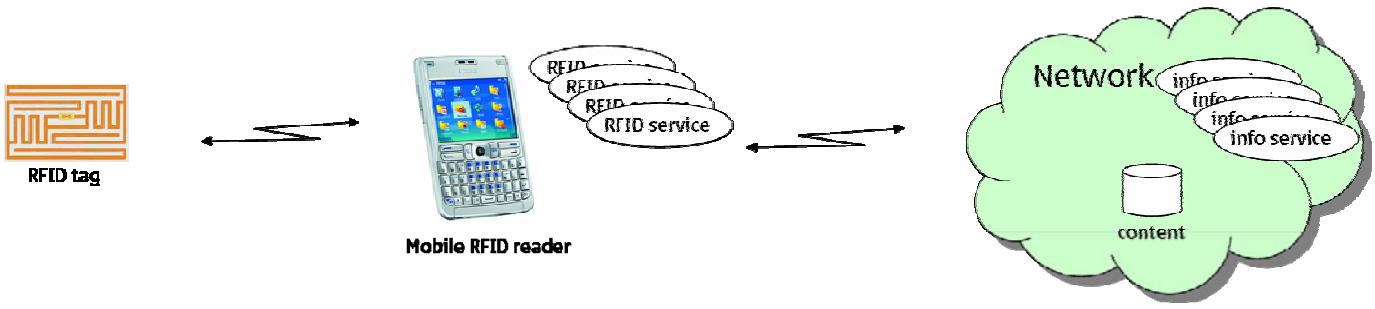

Figure 10: Thin-client based service solution

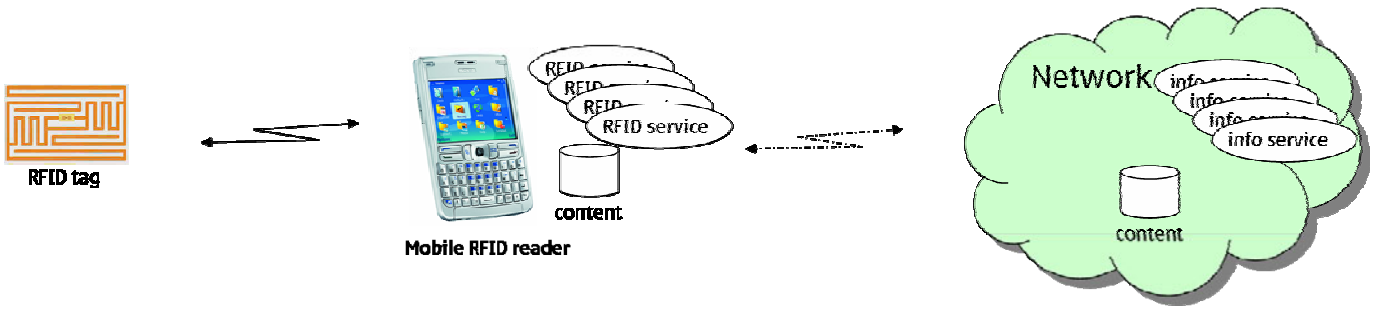

Figure 11: Rich-client based service solution

For a thin-client based service solution, on the mobile device side, we can use the RFID generic application services as depicted in section 3.7. In addition, an adaptation layer is needed between the web browser and the RFID generic application services. The mapping between the basic use case services in the RFID generic application services and the adaptation layer services are one to one. This will be explained in detail in 4.6.1. An

application specific use case can be designed to use the browser services, adaptation services and one or more services from the lower layers. Thus a thin-client based solution provides a common framework of basic mobile RFID services that can be used for specific application needs.

The rich-client based service solution also uses the RFID generic application services as depicted in section 3.7. Application specific use cases can be designed to run as separate applications or services on top of the RFID generic application services layer. Thus a stand-alone application service can be developed by combining one or more services from the RFID generic application services framework. Further explanation will be found in section 4.6.2. Rich-client service solutions can be classified into two types. They are backend dependent client service solutions and stand-alone client service solutions.

Each one of the end-to-end service solutions has its own advantages and drawbacks when compared against the quality attributes. In order to design a practical RFID service solution, quality attributes of the mobile RFID services should be taken as key factors for choosing between the two solutions. It is possible to achieve a compromise between a thin-client and rich-client service solution by designing a quasi rich-client service solution. A quasi rich-client service solution is not exactly a rich-client but acts like a rich-client. It can also be perceived as more than a thin-client. We will see a real example service solution in the next section in detail (4.6). In quasi rich-client solution, we remove the drawbacks of the thin-client and rich-client solutions.

4. Case Study - Library Item Tracking System

This section describes a case study in which we designed the service architecture and the solution for a corporate library by using RFID technology. We used RFID tags and mobile RFID readers. We applied the principles that we described in the previous section for the solution architecture. Before implementation, we analyzed several alternative solution models and walked through all the topics like the problem description, use cases, functional and quality attribute requirements, network architectural models, service architectures, alternatives, and challenges. The implemented solution was exactly intact with the solution architecture. This is proof that service solution architecture can be created and evaluated even without implementation. It also means that most of the problems or issues in the mobile RFID service solution can be solved in the architectural phase as shown here. We used the evaluation of the solution for confirming this fact. The real case study also enabled us to formulate general models and thus this section and the previous section have contributed to each other.

The Library Item Tracking System (LITS) has been developed for the corporate library to manage and track the items of a library such as books, media items, magazines, journals, and so on. LITS provides services which benefit the library, help the postman to collect and deliver items, inform the subscriber (employee) about the status of the items and interact with other legacy systems in the corporate to search and manage items, to make decisions, and to produce reports. Most of the services are using the core RFID services on the client and the server sides as well as other services from the internet and intranet. For the case study only the journal tracking and management is considered.

4.1 Problem Description

Corporate library journals have been circulated manually until the LITS was developed. We studied the basic workflows in order to capture the requirements and to understand the whole system. The basic workflows are printing the circulation list of the subscribers; sorting, distribution, and circulating the journals; and the process after circulation. We identified some problems with the basic workflow of the library system that existed and the stakeholders also knew about them. The library had tried out an RFID-based system and we also considered the lessons learned from that trial. A user experience

team in our organization had interviewed the stakeholders about the general feeling and the lessons learned. We took their input as well.

Corporate employees are the main customers of the library. They work in different locations. For example employees in location A and location B can order magazines from the intranet. The request is then sent as an email to the librarian who sits in the main library. She periodically enters the subscriber details manually into a third party system (say LNet), which is in the internet. When a new journal arrives it is checked into LNet, which will then show the list of people who have ordered it. The list is printed and stapled with the journal. The postal department worker (postman) also underlines the day-to-day time limit of circulation to make it more noticeable.

The postman sorts out the journals in the morning along with the magazines that have arrived by then. At this point the magazines are not individually sorted, only by floor and building (for example 4F). They are placed into a correct compartment on the delivery shelf.

The postman distributes the magazines twice a day. The magazines that come early in the morning are also delivered in the morning. Others are taken on the second round in the afternoon. A few magazines and newspapers go straight to the corporate head office library and to couple of coffee rooms. Employees in location B only get their magazines after location A is done with them. The postman collects the sorted mails and magazines from the delivery shelf and places them in a cart according to the distribution order (an experienced postman knows the route he is about to walk through so that he can make his route shorter). Journals are then delivered to the subscriber on top of each list.

After a subscriber has read her magazine (ideally in two days), she returns it to the out mailbox in her floor area. The postman checks the outgoing mailboxes and if the next person on the circulation list is located on the same route he is currently going through, he will take the magazine directly to her. Otherwise he takes it back to the post office and delivers it during the afternoon rounds. Subscribers should mark the return date on the slip attached with the magazine. It helps to see how long the magazine has been in the circulation, if it has been read by everyone on the list, and if it can be taken out of circulation.

After the completion of the circulation, the circulation list is checked again to see if someone was on holiday or for some other reason didn't get to read her magazine. Subscribers are never skipped completely. If the person has left a note on her mailbox about not being available for a certain amount of time, she will get the magazine after the rest of the circulation is done. When there are no more people left on the circulation list, the journal is taken out of circulation. Most of the magazines are sent to location B where they are archived in their library. Only a few of the magazines are archived in location A's own library.

Some of the problems identified in the current workflow are critical.

- Forgetting to return the magazines: If someone forgets to return a magazine no one knows who has it because the magazines are not being tracked.

- Holidays: People forget to mention they are on holiday and get a ton of magazines in their mailbox. It would be enough for a subscriber to put a post-it note on their mailbox so the postman could skip them from delivery. This problem is being reduced by not doing the magazine rotation in July which is the most popular holiday month. If the postman notices magazines piling up into someone's mailbox he will take the magazines back and carry on with the rotation. But if workers take magazines to their offices or homes and leave them there, then there's nothing the postman can do.

- Extra items: Some magazines come with an extra item like a CD or a DVD. Sometimes when a reader returns a magazine the extra item is not there anymore. There is no way of knowing who has it or what the item even was.

- Printing: LNet only allows printing names in the order they have first been typed so the order of the list can not be changed. Initially the names have been typed in so that everyone in location A comes first and the employees in location B are at the end of the list. But when a new worker in location A is added she is placed last on the list. It's not efficient to

send the magazines back and forth so the mail room worker has to check the list manually to see if there are any more location A subscribers somewhere between names.

The postal workers had taken part in a previous RFID tryout where RFID was used to track packages between different corporate offices. Their comments about the trial revealed that they felt it took time and effort to get the device to read the RFID tag. Every time, the reading took up to a few minutes, which slowed down the normal work process. The battery didn't last for more than a few minutes so the device had to be kept plugged in to the charger, and after a while that made the device heat up. After recharging, the device wouldn't always turn back on or it wouldn't start the program. Sometimes it would suddenly reboot itself and then start working. If it worked faster (like "beep") and had a better battery, they would have a positive attitude towards using it. They thought the concept itself was great. They also mentioned that during the tryout the mail was never late so the RFID really helped with that. They didn't find it difficult to carry one more object with them on the rounds. Their work outfit includes a vest with pockets where they normally keep their own mobile phones, for instance. And if they had their hands full they could always put the RFID reader in the mail cart. Right now there's no way to keep track of the magazines so in case a journal disappears it would be really helpful to know who had it last. It would also be useful to track down the missing extra items. The postal workers also think that sending a reminder e-mail to employees who have forgotten to return their journals is a very good idea. Overall if the technology was fast and reliable they would be happy to use it.

The corporate management has been looking at the costs of operating the library, particularly at how effectively the journals are circulated. Some of the journals are as expensive as 9000€. So they want an IT system to be developed for this. The management wants not only reduce the cost of the journals but also they want statistics about how effectively the journals are circulated and whether it is possible to share the journals between corporate offices which are located world wide.

At the moment, with the current library system, the policies are not clear for the librarians or corporate management or the postman. They have no way to enforce the policies manually. For instance, there is no way to know if a particular employee is not

returning her journals for a long time. Thus, the current system has a lot of loopholes and inconsistencies. There is a clear need for an automated IT solution which would make the system run more effectively.

4.2 Use Cases

A use case diagram helps us represent the whole system that we have analyzed so far. It also helps us to realize the current problems the library system has. At this phase, we do not think about the solution implementation or underlying technologies. However, we assume RFID tags, mobile RFID readers, and other software services can be used to create a solution for the new library system. We name this new system Library Item Tracking System (LITS).

The system level use cases of the LITS are shown in Figure 12, which gives a high-level overview of the system. More details of some of the high-level use cases are described in appendix A.

The stakeholders of the system are subscribers, librarian, postman, and administrator. Employees are the subscribers and the main customers of the system. The whole idea of the corporate library is to serve the employees by providing them with the journals, magazines, media, and books they need in their work. Employees sit inside or outside the corporate offices in different locations. They can connect to the corporate IT system and can also access the internet. The main services to the employees are subscription and unsubscription of the journals as well as viewing and printing history data about their previous subscriptions.

The postman is the one who physically provides normal postal services. But here in addition, he would also attach tags, sort journals, collect journals, deliver journals, and print circulation lists. He is not working for the corporate but is an external worker who may not have access to the corporate IT system.

Librarians are corporate employees who manage the library services for the corporate. The main services the system would provide for the librarian are the management of the library resources such as managing journals, managing subscribers, managing circulations and issues of each journal, and managing the orders of circulation.

The librarian can also apply the corporate library policies for the resources and customers through the LITS system.

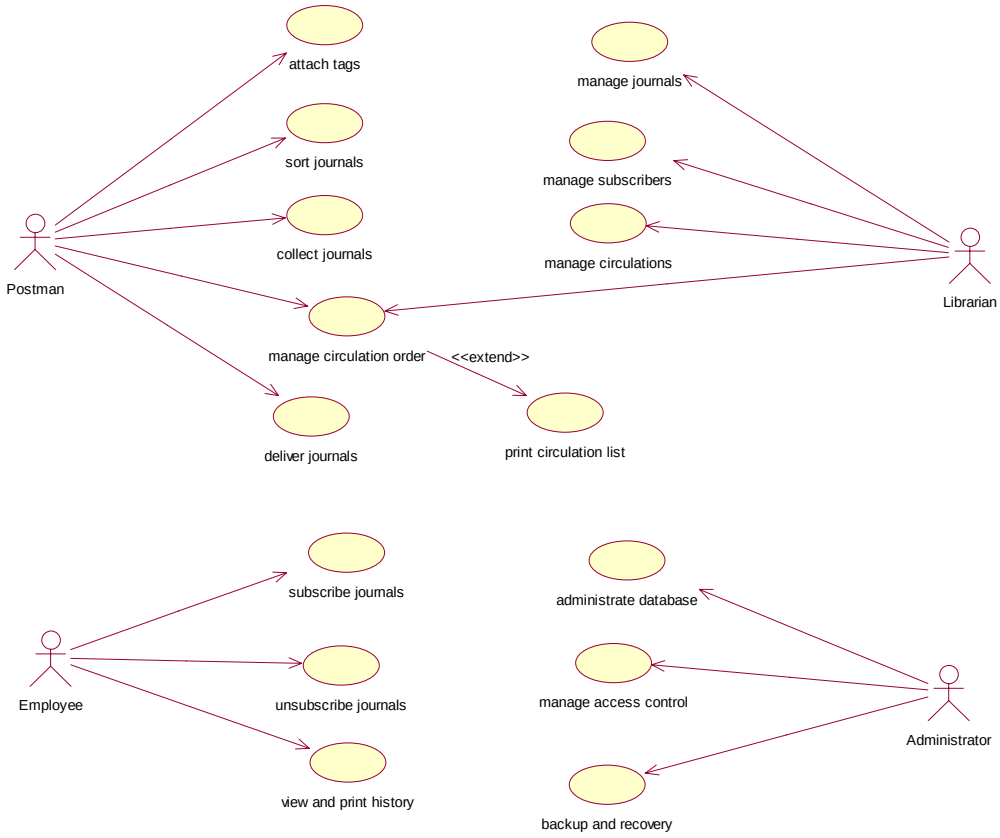

Figure 12: System level Use Cases of the LITS

Finally, in order to manage the whole LITS system technically, an administrator is needed. The administrator would administrate the database system, do backup and recovery and give access control for the users to different services of the system.

4.2.1 An Example Use Case – Attach Tag

We describe one of the use cases in detail to give an operational view of the system (Figure 13). The numbers in the sequence diagram represent the sequence of steps during the execution of the use case.

The attach tag use case is used for attaching a newly arrived library journal with an RFID tag. This use case is significant because it explains the scenario for binding a

physical object with its equivalent digital information. The actor of this use case is the postman. An RFID tag is physically attached by the postman. In addition, the postman also updates the LITS system with the information about the attached journal item. This must be done item by item.

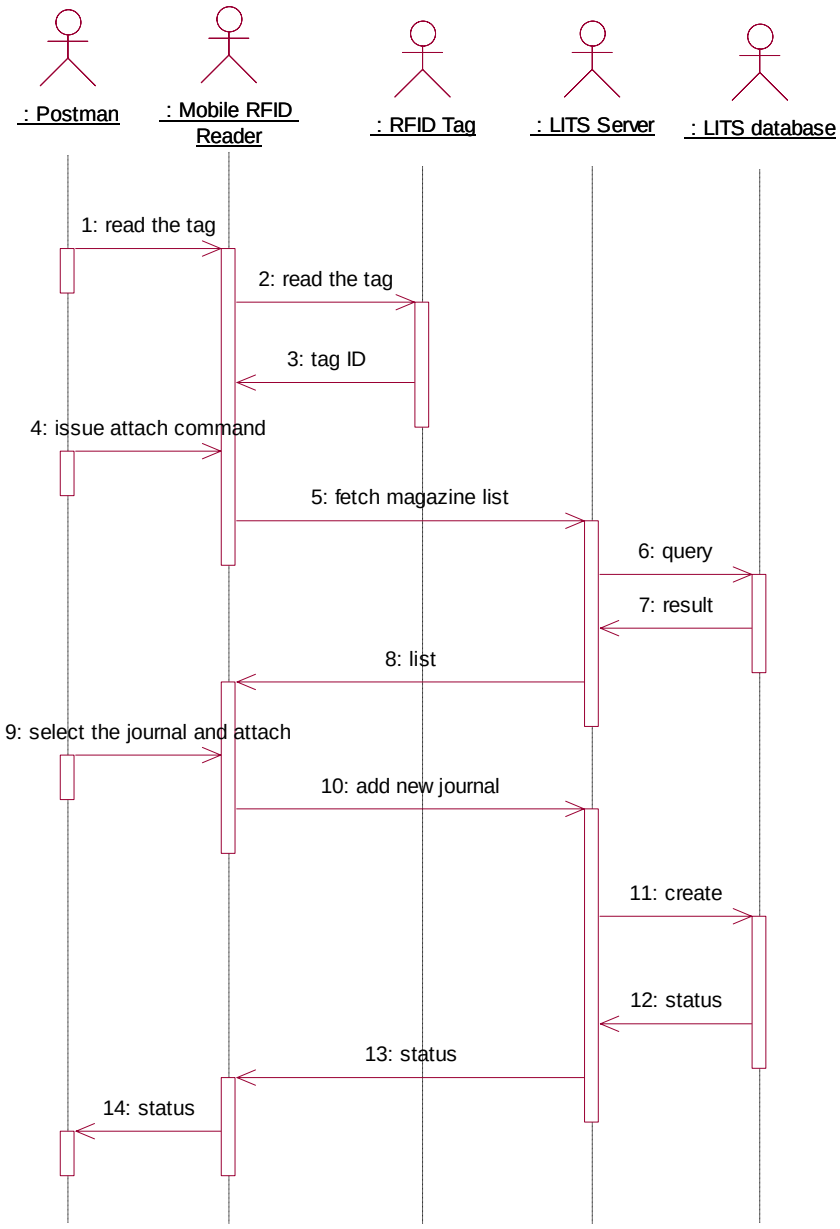

Figure 13: Attach Tag - sequence diagram

This use case is used only when some new journals are added to the library. New journals don't arrive very often, so the frequency of this use case is once in a month or so. There are also some preconditions for this use case. The LITS client needs to be installed on the mobile RFID reader so that postman can use it during attachment. The mobile RFID reader must also be configured to connect to the network. At the backend side, the LITS server and database should be up and running. Finally the postman must have access to the LITS system.

To start this use case, the postman starts the LITS client in the mobile RFID reader and brings the mobile RFID reader next to the new RFID tag. The mobile RFID reader reads the tag and it answers back to the mobile RFID reader. Now, LITS client has the ID of the read tag. Then the postman physically attaches this tag to the new journal and issues the command to attach. The LITS client asks the LITS server about the list of journals that can be attached. The LITS server queries the LITS database and gets the list of journals that can be attached. The LITS server sends the list to the client. The postman selects the journal from the list and issues a command via UI menu item. The LITS client sends the information to the LITS server for attaching. The LITS server creates a new entry for the tag and magazine entries in the LITS database. The LITS database informs back the status to the LITS server, which forwards the status to the LITS client. The LITS client displays the status of the operation on the mobile RFID reader's screen. If the attachment procedure fails, it will inform that as well. This procedure will be continued for all the newly arrived journals. The other use cases of the LITS system are described in Appendix A.

4.3 Quality Attribute Requirements

In addition to the functional requirements, it is important to include the quality attribute requirements of the system [BCK98, BaK99]. The quality attributes are also called non-functional requirements. The non-functional requirements are researched by interviewing the stakeholders of the existing library system. In addition, a general model for providing an architectural solution is used [BBC00, OMB05, BEL03, Off02].

The quality attribute requirements cover the overall system qualities and the attributes. Here, we only focus on the RFID services of the LITS system. The RFID services that we identified from the use case analysis are attach tag, sort journals, collect

journal, deliver journal, skip subscriber, extend subscription period, and detach tag. These services are taken as the input for finding the system constraints and attributes.

4.3.1 Runtime System Qualities

The runtime system qualities in this subsection are the measurements that we would like to have in the RFID services while the LITS system executes. The considered runtime qualities are functionality, performance, security, availability, usability, and interoperability [BKL95]. The following paragraphs explain these attributes in detail.

Functionality is the ability of the LITS system to do the work it is intended to do. Here we concentrate especially on the RFID services part of the system. If an RFID service is invoked to perform a task or a set of tasks by the user, it should work as described in the use cases and it should maintain the consistency and integrity of the other services of the LITS system. Basically, during the system's runtime, the functional requirements should be fulfilled as expected.

Performance of all RFID services is very critical. The response time of the services determines the success of the whole system. The postman expects that the mobile RFID reader device should respond in one or two seconds. If the response time is more than two seconds, it would frustrate the postman. In practice, the time delay includes the time for the tasks such as scanning or reading a tag, processing and getting information from the server. Some of the RFID services may also have to use more than one database access request which has to travel on the network. This is a challenging issue for performance. For example, the sorting service would have to respond so quickly that the postman can just touch the tag and immediately see the subscriber and her location to sort out the journal in his moving cart. The radio network has to respond quickly with the content when the tag info is queried from the device. Also, when a tag is touched the device should read it quickly. When more than one postman is involved with the system, the RFID services should still function with high speed without delays.

Security is a very sensitive area when RFID is involved in a system. When people hear about RFID services the security and privacy question immediately pops up. The RFID services provided by LITS should not be available for every stakeholder of the system. Only the postal staff should be able to access the services with the supplied username and password. The system should deny access from any unauthorized users.

Since it would be logical to integrate other corporate services with the LITS system, this might open some security holes. The LITS system should not allow outsiders access to any corporate services or devices. It should comply with all the corporate security polices and eliminate any security vulnerabilities. It should avoid any deployment of unnecessary WLAN access which could create security risks. If any other network is used out of the corporate, it should use secured protocols for providing RFID services. In case the mobile RFID reader device is stolen, the RFID services should not be available. In addition, normal general security measures, such as keeping wireless networks password protected and using transport and content level security in wired networks, should be applied.

When availability is considered, the RFID services should be available for the postmen who access the LITS system from different geographical locations. If we would provide RFID services on mobile devices, it would imply a need for the whole LITS system to be in operation all the working days. However, it is not a mission critical system if the backend server or database crash or go down. If there is a failure of the backend server or database, the library system won't suffer that much. It is enough to bring the system up in couple of days in those failures. Such failures should not happen often.

In case of failure of the server there should be backups to bring the system into its previous state. Even if the network or backend server fails, the RFID services should run incorrectly. And in case the postman loses the device or the device crashes, the services should still run without much hassle. The backend server and database services should be accessible for the mobile RFID reader device even from different countries. In case of mobile network failure, the services should be available on the web for verification.

The usability of RFID services should be easy for both non-technical postal staff and technical users. The RFID services should be easy to use and be self-evident. They should not take too much time to learn. Without the need to spend a lot of time on user manual to learn or for extensive training, one is able to use them. The UI menu text and the other screen texts should be self descriptive and provide enough hints for the users about the current status or the ongoing operations of the services. The RFID services should help the postal staff to do their work more effectively than before. They should not feel in anyway that the RFID services are bottle necks in their normal daily work.

The RFID service application which runs on the mobile device should not have a lot of navigations of UI screens. It should be very simple even to operate, sometimes with out any user key presses. For example, in case of sorting the journals it should provide the services without the user typing anything on the screen at all. The postman also should not have to type the user name and password into the system many times a day. The menu commands should be consistent in a way that the system should always display the type of commands on the same side of the screens so that the user would not accidentally press the other button when he is in a hurry. The UI style of all RFID service screens should be consistent. Also the RFID service application(s) should be placed in the right place on the phone main screen/folder with proper icons for every service. The text and the background color of the UI should have enough contrast in order to read the text the screen displays so quickly.

The RFID service applications should handle the errors and exceptions properly. The common user typo errors should be avoided by the UI by providing proper formatted text types for the UI fields or controls on the screen. The user data should be validated before any process is started and must be informed to the user in a very friendly manner. If the services encounter some errors, it should be conveyed to the user in a very friendly readable manner. The errors should not cause any interference to the normal operations. The services should be able to be operated when the application is restarted without any inconsistencies. The errors in the services should not cause the whole operating system of the device to crash.

Overall the users of RFID services should feel comfortable with using the services and be satisfied with them. The application or applications which provide RFID services to the users should not let the user wait long while the service is being executed or while the network is being contacted. If an operation takes a long time to execute, the user should be informed about the estimates or progress on the screen. There should be always be a timeout so that the user would not have to wait for indefinite periods.

The mobile RFID services are provided on the mobile RFID reader device. The postal staff will also use this device as their personal phone device to make calls, receive calls, to get email, and for other tasks. So, the RFID services provided on the device should not any way affect any other phone or device functionalities or services and

they should not crash the device in any situation. While using the RFID service if there is an incoming call, the service should provide possibility to attend the call. After the call is over, the service should continue the functionality as expected. An incoming call shouldn't crash the device or corrupt data.

Interoperability is the ability to exchange information and use the information that has been exchanged by other systems. LITS system should be interoperable because in the future, it may need to use some of the corporate services while providing RFID services. Other systems in the corporate will also have to be able to make use of the RFID services provided by the LITS system.

The RFID services may have to be implemented as composite services by combining some services from different lower levels. The services that we need to use from the mobile RFID reader device should be generalized and layered so that if later some new reader chip arrives in the market only the changes that affect that particular part of the service would have to be replaced. General services should be formulated at the middle layers so that irrespective of the tag types the services at the lower layer would be reused. The key point here is to design the services into multiple layers and to provide adequate abstraction on each layer. The higher layer services should be implemented independently of some telecom operator, device manufacturer, software platform, or language reference details. The interfaces the service would provide should have enough room for extension in the future and should be language and platform independent as much as possible.

The web services provided should be interoperable and universally accessible irrespective of the platform or language that the accessing device or workstation uses. The backend services should also define and provide an interface to integrate the services with any existing or upcoming business processes at a high abstraction level. Interoperability of web services across platforms, applications, and programming languages of the LITS system should be thought of.

4.3.2 Non-Runtime System Qualities

Non-runtime system qualities cannot be measured as the system executes. Here, modifiability, portability, reusability, integrity, and testability are considered as such attributes.

Modifiability is the degree to which a system or component facilitates the incorporation of changes, once the nature of the desired change has been determined. The mobile RFID services of LITS should be designed to accommodate future anticipated changes. The changes can be expected in the back-end server technology, database, network related parameters and services. On the mobile RFID reader side, the RFID reader hardware chip, smart phone's performance (processing speed and memory capacity), and energy consumption will improve. The RFID tags will also change in a way that they can store larger amounts of data. There can also be new tag types manufactured in the future.

When portability is considered, LITS system should be easily portable so that it would run under different computing environments. The environment types can be either hardware or software or combination of both.

Reusability provides the possibility to perceive the system components from the perspective of reusable parts. At the same time, reusability means defining the general components so that some of the newly developed parts of the system can be reused later for similar kind of services or functionalities. The mobile RFID services developed for the LITS system should be reusable for other RFID applications. The software services developed at different levels should easily be reusable when there is a change in the hardware part of the device. The server side services should also be reusable.

Integration of the system is done after developing all the parts of the system. The RFID services should be developed in a way that the separately developed components of the system will work correctly together after integration. The interfaces for the services at different levels should be defined carefully.

Testability is verifying the system for correctness. The RFID services should be tested before the trials. Some test services should be created while development.

4.4 System Overview

Now, we have understood the functional and non-functional requirements of the mobile RFID services for the LITS system. Next we outline the solution architecture by following a couple of architectural models. The chosen models for our case study are the network architecture model and the service architecture model. The network architecture

model gives the network configuration perspective of the LITS system. The service architecture model gives the service perspective.

The LITS system can be outlined as shown in Figure 14. In the center, we have the devices and devices like mobile RFID reader devices, RFID tags, server and database computers, client workstations and laptops. They are connected to different networks like internet, intranet, WLAN, and UMTS networks. Finally, we can see various software applications that run on the networks and devices described as abstract services. Our challenge is to find out a solution for our problem by reusing, composing, and designing the existing and new services on the elements that we see in the figure. In the next sections we explore some possibilities for arriving at a satisfying solution.

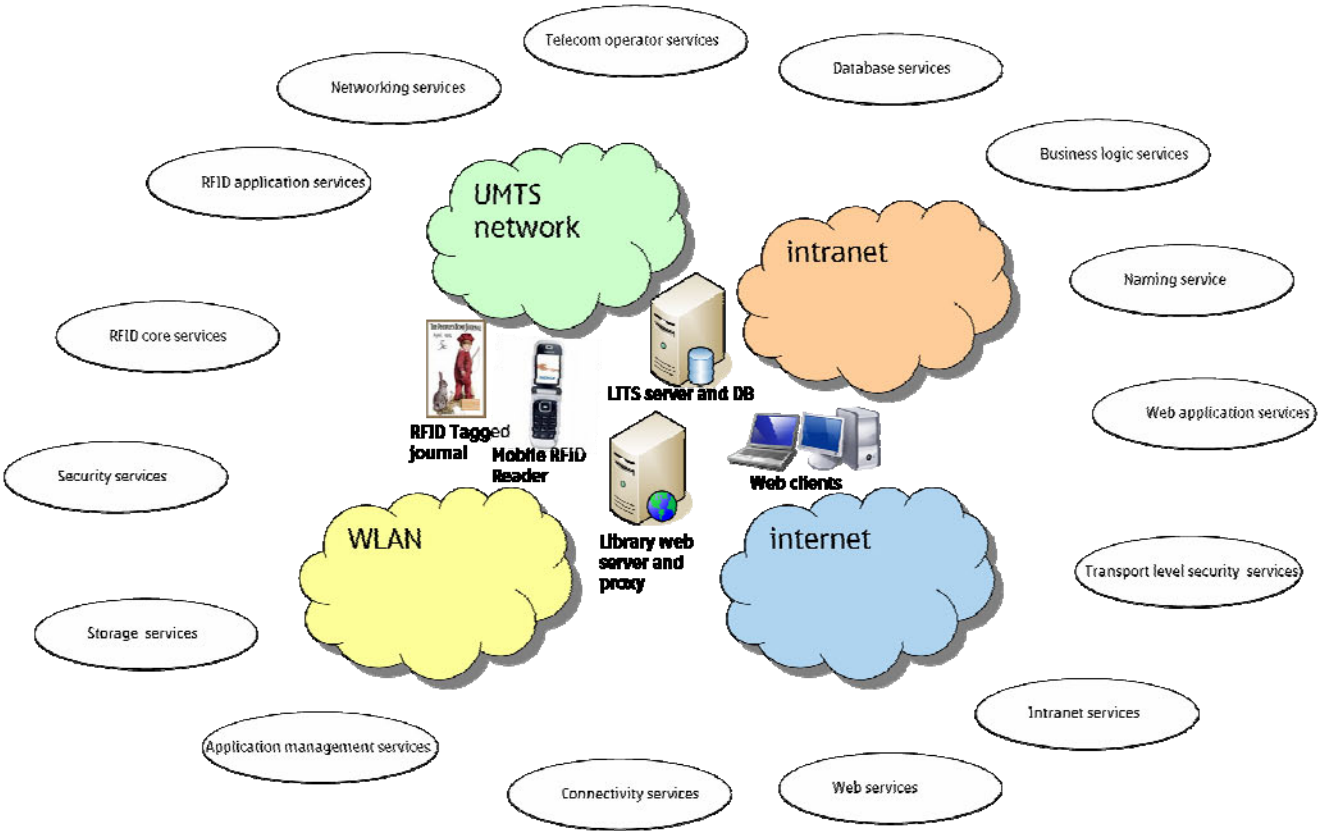

Figure 14 : LITS system overview - devices, network and services

4.5 Network Configurations

The LITS system must use a distributed service architecture model as the services are spread across different networks and geographical locations. The basic elements of the network are backend server, database server, intranet, internet, mobile network, WLAN, smart mobile RFID reader device, and RFID tags.

A backend server is used to run the backend application and web services. A database server is used to host the database services. The intranet provides all the corporate communication services and where all the employees and library personnel are connected to. The internet is connected through a proxy and firewall to intranet. Mobile network provides the packet data services and covers the corporate buildings. Intranet and internet WLAN is used to extend the wireless mobility. The corporate employees (subscribers and librarians of LITS) have the access to internet through intranet. They can also access the intranet services from internet via VPN when they are outside the corporate premises. Employees can also access the intranet services on their smart mobile devices. The postman doesn't have access to the corporate intranet at all. But in his office, he can access the internet.

Depending up on where the server is placed in the network the access to the services for the actors varies. For example, if the server is placed in the intranet, the LITS services cannot be accessed on the postman's workstation, however the subscribers and librarian can access the services. But if the server is connected to the internet, the corporate information about the subscriber could not be accessed by the server. These alternatives give us a starting point when evaluating different network architecture models for the LITS system.

We have different alternatives at different levels. We could use WLAN or UMTS network for wireless data access, and we could use internet or intranet for web access. Each alternatives has its constraints and advantages on every of the alternatives. After analysis, we have identified four possible configurations that are explained in detail in the following subsections.

4.5.1 Internet - UMTS Based Alternative

Figure 15 depicts the internet – UMTS based alternative. The LITS server and database is connected to the public internet. The web clients are connected also to the public internet. The mobile network is connected to the public internet via the gateway. The mobile RFID reader can access the backend server via a gateway.

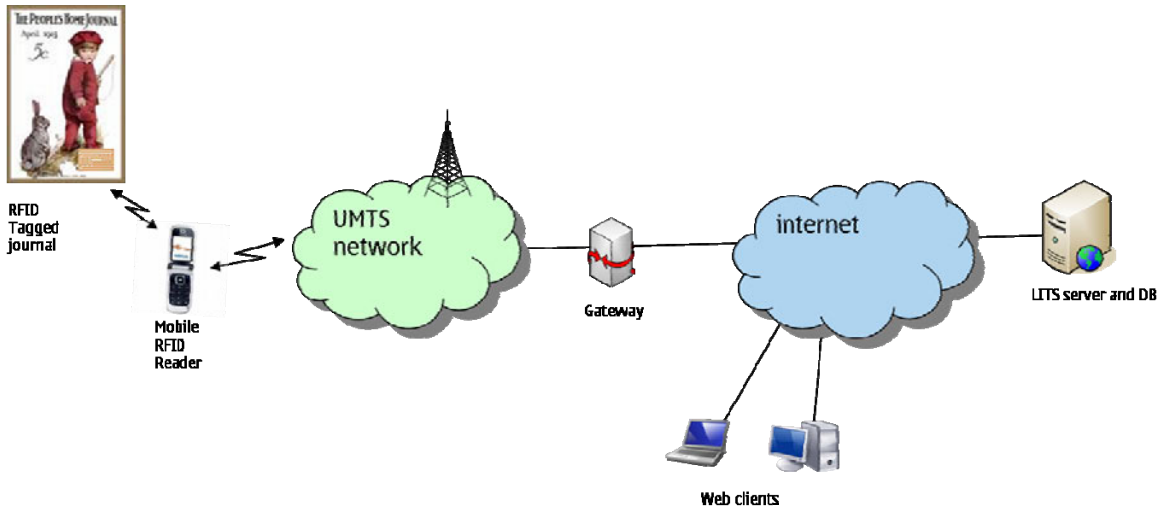

Figure 15 : Internet - UMTS based network architecture model

The mobile RFID services can be accessed on the device from the backend services all the way from the mobile network to the internet. The drawback of this model is that the corporate information services cannot be accessed and used by the LITS services. In other words, LITS backend server cannot connect to the corporate intranet as it is placed outside of it.

4.5.2 Intranet - UMTS Based Alternative

The network configuration of this alternative is same as the previous one except that instead of the internet, the corporate intranet is used. The configuration of this model is depicted in Figure 16.

The mobile RFID services can be accessed via the intranet on the mobile RFID reader device by running some VPN tunnel services. The main drawback of this alternative is that the postman has no access to the LITS services because he has no access to the intranet.

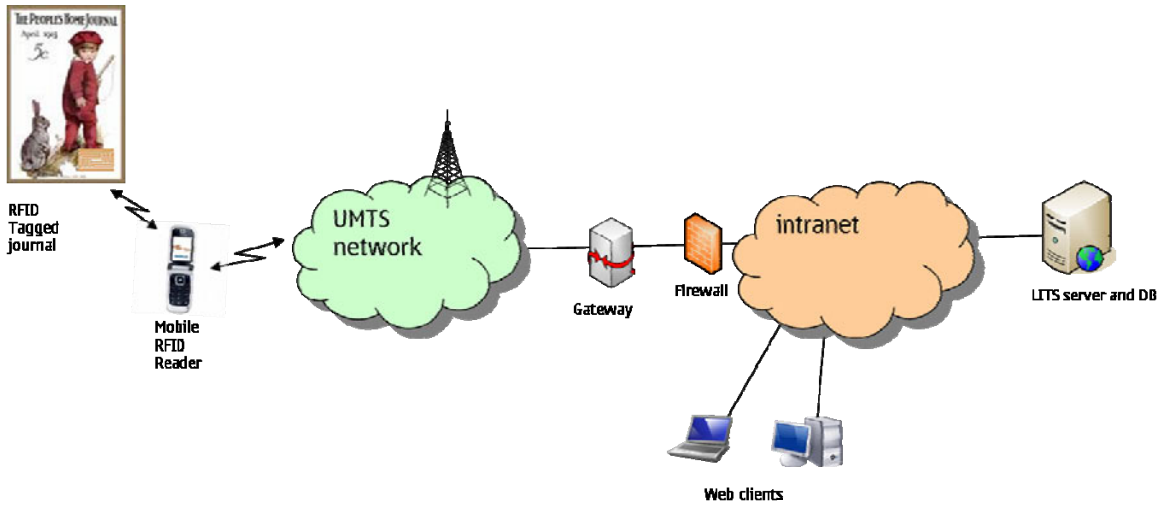

Figure 16 : Intranet - UMTS based network architecture model

4.5.3 Internet - WLAN Based Alternative

In this alternative, the WLAN wireless network is used instead of UMTS network to connect to the internet. The LITS server and database are connected to the internet and the mobile RFID reader device accesses the backend services through the WLAN which is connected to the internet as shown in Figure 17.

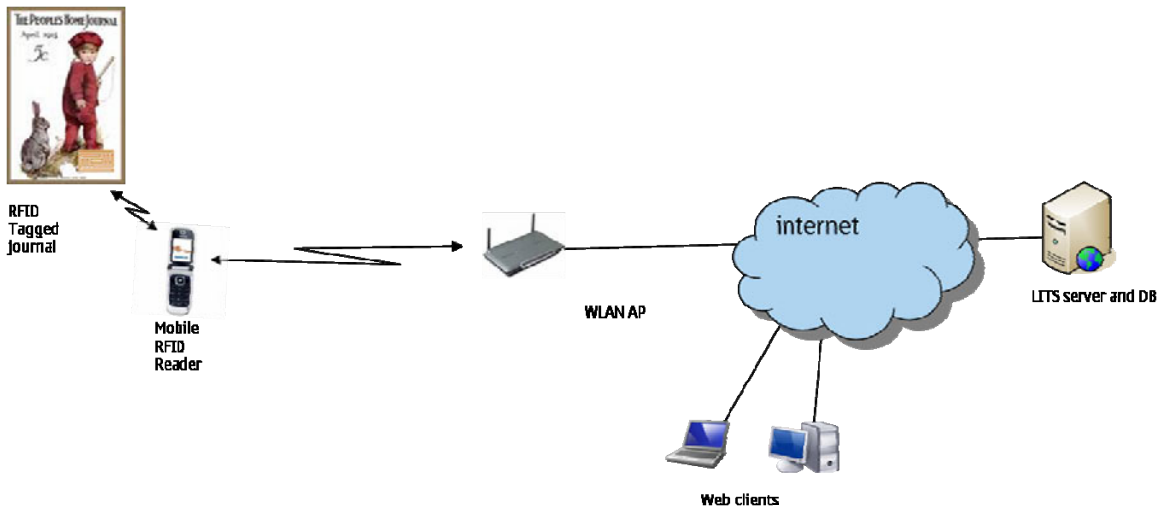

Figure 17 : Internet - WLAN based network architecture model

The drawback of this model is the same as with internet – UMTS based alternative. In addition, the WLAN network is exposed to security vulnerabilities. Furthermore, not all the corporate offices are covered by the public WLAN access points.

4.5.4 Internet & Intranet - UMTS Based Combination

We combine some of the elements and networks in the previous alternatives to come up with a hybrid alternative. Here, we bring another new network element to overcome the drawbacks that we saw in the previous models. It is a server device which is called the Library web server and proxy. It runs the proxy service and the web service. It is connected to the intranet as shown in Figure 18.

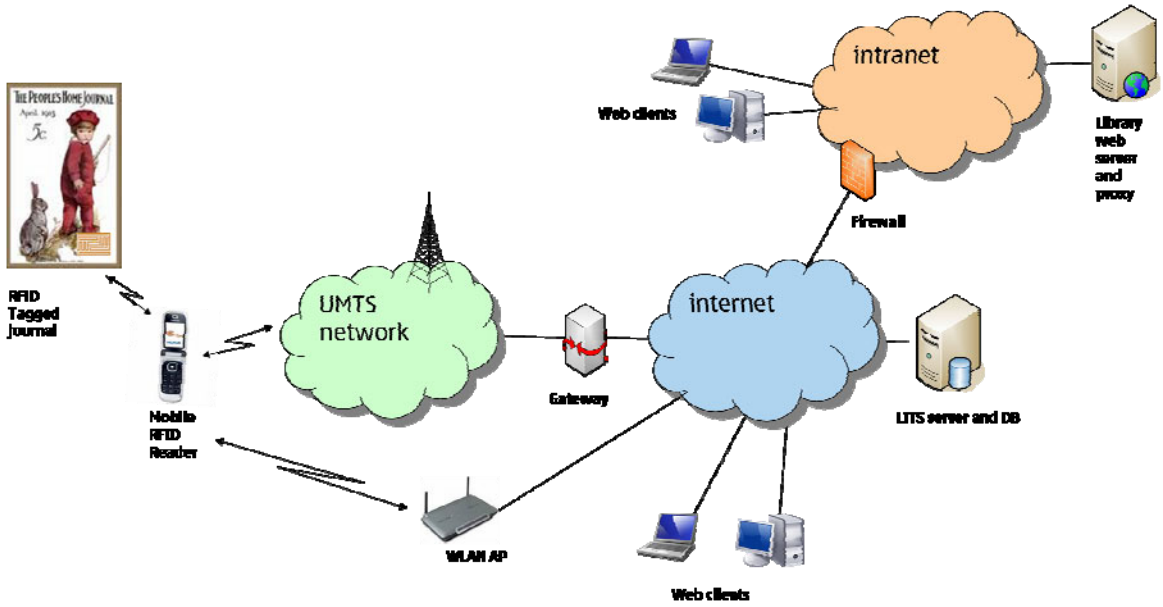

Figure 18 : Internet/intranet - UMTS/WLAN network architecture model

The library web server is connected to the corporate intranet as a proxy to provide the library web services to the subscribers who are also in the same network. The library web server can access the corporate services within the network. The LITS service and database is in a backend server which is connected to the internet. The backend service and database can be accessed via the library web server from the intranet. At the same time, postman who doesn't have the access to the corporate intranet could access the library services from his device which is connected to the same internet network.

When it comes to RFID services at the mobile device side, the postman has a couple of options for wireless radio access network. He could either use the WLAN radio to connect to the LITS services on the internet or he could use UMTS radio network. WLAN access points are protected by some security measures like WEP key.

4.6 Service Architectures

In the previous sections, we have seen the overview of the LITS system and the different network configuration scenarios. Now, it is time to look at the whole system from the services point of view. We perceive the LITS system as a composition of software services [Fow03]. The software services run on different hardware and network infrastructures and use their basic services. We don't go into the details of what kind of services every network infrastructure provides; instead we focus on how the LITS services can be structured. However in order to architect the whole system effectively, it is important to have basic knowledge of the services and interfaces to the services provided by the underlying network infrastructure, operating system, and application software platforms.

From the LITS system overview, we conclude the basic logical service elements for of the LITS from the services point of view. The LITS system has four elements or logical blocks that are logically connected to each other as shown in Figure 19. They are the RFID services on the mobile device, services for the web client, server and database backend services, and communication services. To come to this logical conclusion, we have also used the knowledge and reasoning gained by looking at the use cases, the available network infrastructure and the latest trends in the technologies.

Providing RFID services on the mobile devices need some analysis about the latest development in the mobile RFID services technology area. Lately, RFID services have been researched and developed for various use cases on different geographical locations. RFID reader devices have been used in logistics for a long time, but a mobile phone with an RFID reader is not commonly used. Some device manufacturers have launched NFC devices few years ago.

Mobile RFID reader phones are being researched and developed by various vendors and mobile device manufacturers across the globe. These devices are able to not only use one type of tag like NFC but also able to recognize any kind of RFID tags like EPC type tags. The smart mobile devices have already the capacity to run advanced services on them. Nowadays it is very common to run advanced applications like emails, web browsers, widgets, web based applications, games, music players, and radio on the smart phones. If these devices can also integrate the RFID reader hardware, it is possible to

have a RFID services platform on these devices. This helps us to combine the smart phone services with RFID reader services on a single device. But that is not the end of the story. In order to make real RFID service applications, we also need the network infrastructure and the backend database support.

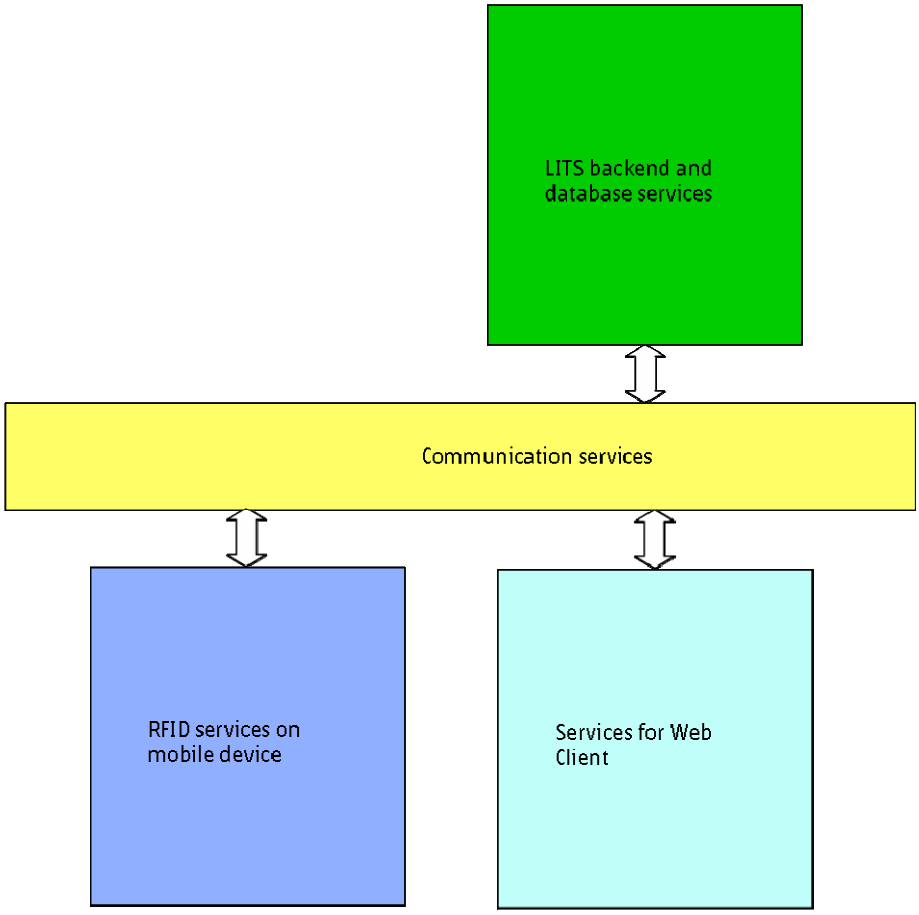

Figure 19 : Basic logical service blocks of LITS

The backend server and database services (see Figure 20) include database services, business logic services, authentication service, security service, logging service, and email service. On the top of them the LITS web services and mobile services are built up. Database services are used for handling data such as data storage, retrieval, maintenance, and backup. Business logic services include in addition with the LITS specific logic services, services for processing the incoming data from the clients such as parsing, maintaining a session, and error handling, and providing the contents in different formats for different clients (mobile and web clients). Authentication service is used to verify different clients. Security service is used to enforce the security policy. Logging is

used to keep the records. Email service is used to send emails to the subscribers about the reminders. Backend server services can be combined from the basic services with web services and new services.

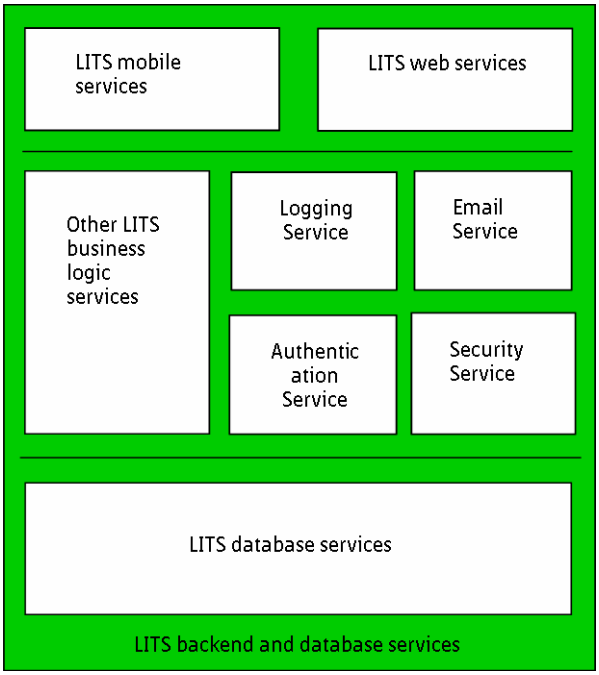

Figure 20 : LITS backend and database services

The application logic of the LITS system can be implemented as a set of software services which could be reused by other software services. There should be enough attention paid to the content of the service and the interface it provides. The content should be a standard one so that any clients outside the system could interpret the data easily. The interface should be a standard one too. There should be enough room for the content and interface formats which would allow us to extend any new features with LITS. It also helps to see the whole LITS system as a set of services to compose with other services to create new application services or to integrate with any other corporate workflow systems. For example, the services could be located by their web URLs so that they can be accessed anywhere from the world.

The web client services are the services which are needed for the client devices to access the LITS services provided by the server across different network. The client devices can be either in the intranet or internet. Usually, the client device would be a laptop, a workstation or a smart mobile phone. The client services should be capable of interpreting the services that the server provides. Interpreting the services to the end users

includes providing UI services, parsing the contents, providing the security to the contents while communication, encrypting and decrypting the content, and providing session support. A standard web browser would be a typical example of accessing web client services from the network. The standard browsers usually include all the mentioned client services in them.

Communication services provide possibilities for communication between the services which are separated and running across different geographical locations. Details of underlying protocols are unnecessary at this point as we clearly know that the possibility to communicate between high level application services is a proven concept. Communication services make the system more flexible to organize the services across different locations of the corporate and enable the service mobility. Communication services can be described as a service bus which could be used as a medium to connect the other services which are spread across the geographical location. Communication services can be used to connect a smart phone to a PC to transfer data between them or can it can be perceived as a medium to access the web services over the air while the device is moving, or the normal web services that are used between two connected PCs in the internet.

Now that we have figured out the logical block structure for the service architecture, we need to find out the actual application services for the LITS system in order to place them inside these blocks. For that we take the use cases as the input. The use cases are once again analyzed from the service point of view and explored a bit more. We came up with some RFID generic use cases from LITS application specific use cases as shown in Table 4.

We may need one or more services for every use case and same service can be used for multiple use cases. After working it out we found the correlation as listed in the table. From the LITS application specific services, we found the generic application services. Some of the generic application services are repeating. We normalize the generic use cases to create generic services to make it as a common denominator for any RFID applications. We have also found out the commonality in the parameters the services use and the patterns they repeat.

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 the item	Attach info	Attach tag	{service_name, reader_id, tag_id, info_field1, info_field2, …, info_fieldn}	unique
2.	Sort journals (by reading the subscriber info)	Reading info from the item	Read subscriber	Read info	{service_name, reader_id, tag_id, info_fieldm}	unique
3.	Collect journals	Updating the info	Collect journal	Update info	{service_name, reader_id, tag_id, status_fieldm}	unique
4.	Deliver journals	Updating the info	Deliver journal	Update info	{service_name, reader_id, tag_id, status_fieldm}	repeated
5.	Skip Subscriber	Updating the info	Skip subscriber	Update info	{service_name, reader_id, tag_id, status_fieldm}	repeated
6.	Extend subscription period	Updating the info	Extend subscriber	Update info	{service_name, reader_id, tag_id, status_fieldm}	repeated
7.	Detach tag	Detaching info from the item	Detach info	Detach tag	{service_name, reader_id, tag_id}	unique

Table 4 : Service identification from use cases

We apply the client/server architecture to further structure the LITS service architecture. From the four logical blocks that we have seen, we bring our focus more towards Mobile RFID services. So, let us analyze the service solutions from the mobile

RFID services point of view. The mobile device is the one which is used by the end user to access the RFID services. In that respect the mobile device can be perceived as a client node.

We classify the client service solutions based on the services on different levels especially at the mobile device side. The client service solutions can be classified into two classes according to the way the services are composed using the presentation services. They are a thin-client based service solution and a rich-client based service solution.

Before going into the details of these two classes, let us see the set of common services needed for both of the solution models. The common services include RFID application services, RFID reader services, and common mobile services. The elements of common services are depicted in Figure 21. The top layer provides the presentation services for the user along with communication services and others. We will look into details of this layer in the following subsections. The RFID application services layer consists of the basic RFID application services such as attaching a tag, detaching a tag, reading info and updating info. These services are not specific to LITS but they can be generally used to compose application services. For example a shopping application could easily be created on a mobile device by combining these general application services.

Common mobile services are the services which are available in any smart phone devices on the market. Some of the services are connectivity, networking, security, radio, file handling, multithreading, and multimedia handling. Connectivity services provide the possibility to connect to any wireless networks or devices using WLAN, blue tooth, and wibree. Networking services provide the way to handle standard protocols. Radio services provide the way to connect the device to packet and circuit switched network over different carriers like GPRS, GSM or 3G radio.

There are low level hardware interfaces in mobile devices which are managed by the operating system drivers. The operating system drivers can be perceived as operating system services. Some vendors provide access to low level services via API (Application Programming Interface). Nowadays in any leading smart phone device, there are also some vendor specific services which are closed for the public. In addition there are also some open but universally accepted services. For example, Sun's JSR numbers are

used to specify such services [JCP06]. Vendors who implement the JSRs specify the kind of services their device supports.

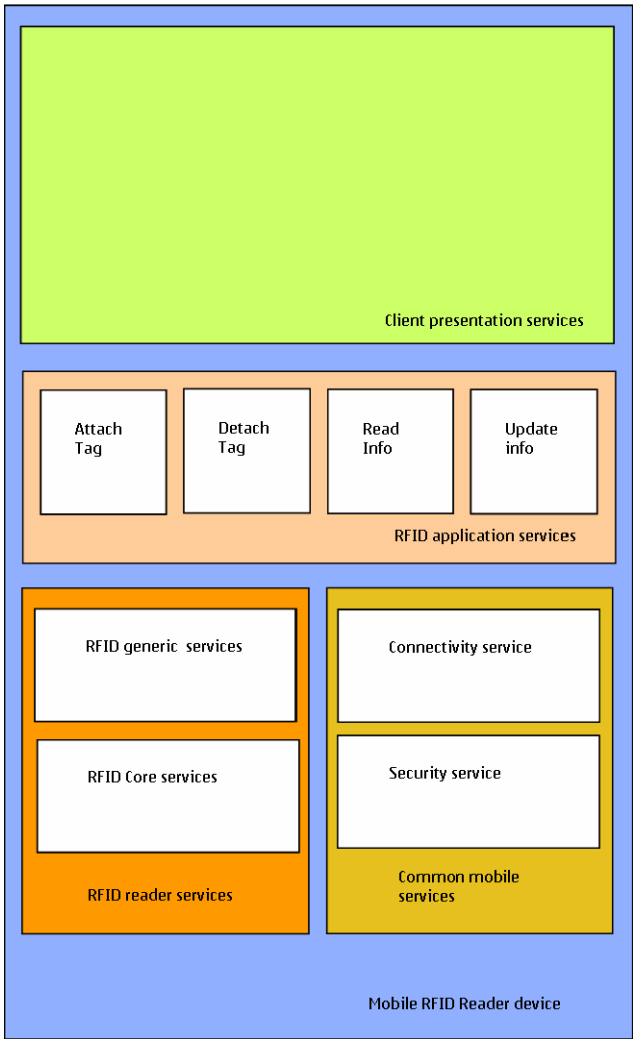

Figure 21 : Common services for all the solutions

RFID reader hardware device can be embedded in a smart phone as other hardware like GPS or blue tooth and can be interfaced via USB or other ports. An RFID reader chip can be controlled by RFID driver software. This can be implemented as an operating system service. RFID driver service should be layered very thin so that if we change the RFID hardware in the future, the whole driver software need not be rewritten. RFID reader services use RFID reader hardware which is connected to a smart mobile device.

RFID Reader services consist of RFID generic services and RFID core services. RFID generic services use RFID core services. RFID generic services are high

level services such as filtering, aggregating, grouping, counting, differential analysis, decoding data, providing data for multiple clients, configuration, and power control [EPC05]. Filtering involves eliminating some EPCs according to their identities, such as eliminating all but EPCs for a specific object class. Aggregating means eliminating duplicate reads within the interval. Grouping function summarizes EPCs within a specific object class. Counting means reporting the number of EPCs rather than the EPC values themselves. Differential analysis means reporting which EPCs have been added or removed rather than all EPCs read. Decoding implies decoding the data into various formats. RFID core services include basic services like reading a tag, locking, writing, and killing a tag. The core services use the tag protocol to talk to the tag.

4.6.1 Thin-Client Based Service Solution

Thin-client based service solution follows a solution model like web application model. Web application is an application software that runs on the server device which is accessed and used via a general web browser over the Internet or an intranet to perform a task or tasks. Web application uses the application logic along with web services at the server, the client, and the communication services as the medium for communication between services. The concept here is that a general minimal processing application would run on the client device, which is capable to process user inputs, store minimum amount of data, cache minimum amount of data, parse standard data content, and able to communicate to the server which resides across the internet or intranet over a wireless network.

Figure 22 shows service architecture of a thin-client based service solution. It is not a typical layered architecture. Services in the top layer can use the services in the bottom layer in general; however the top layer service can also bypass services in the middle layer to access bottom layers.

Presentation services have general presentation services part and application specific adaptation services part. The top layer provides a thin general presentation services for the user. Services in the presentation layer use common mobile services such as connectivity, networking, security, and parsing directly. For example, a general web browser runs on a mobile device is a thin-client which provides general presentation

services. Presentation services can be either started by an user via normal key inputs or it can be triggered automatically due to other hardware events like detection of an RFID tag.

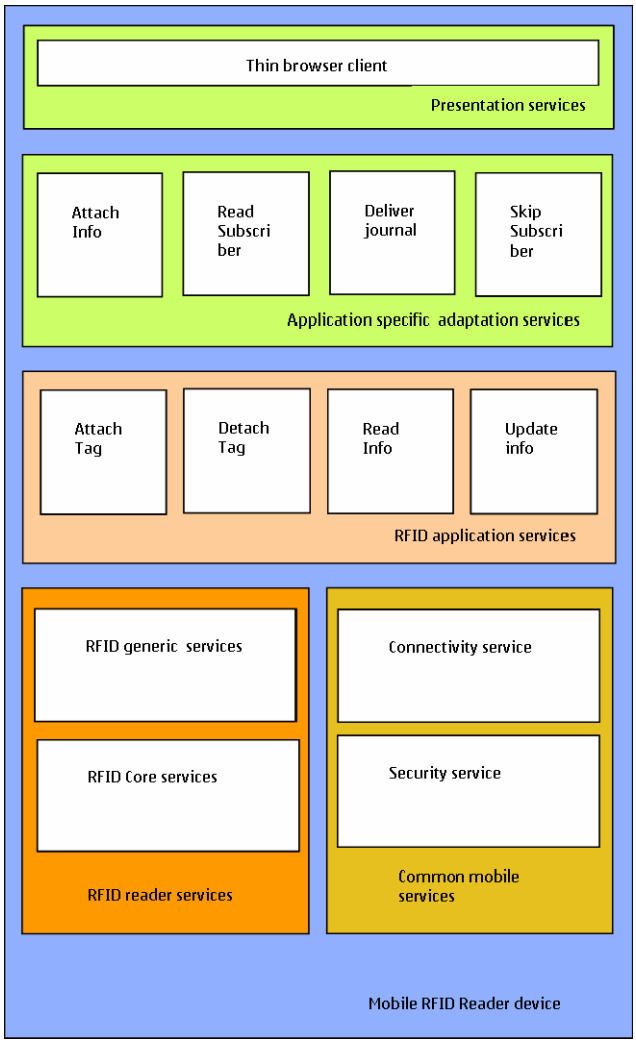

Figure 22 : Thin-client based service solution

The next layer consists of application specific adaptation services. These are the services that glue RFID application services to presentation services. Presentation services can be initiated or provided by application specific adaptation services. It can be a pull or push based one. In addition, these services are very specific (maps to the use cases). They are very thin as well. They just do the tasks like transferring data they get from lower layer to a thin-client or supplying the configured or self contained data needed for a thin-client.

4.6.2 Rich-Client Based Service Solution

Rich-client or fat client service solution relies on the local resources to execute the services. It doesn't always rely on the communication services or backend server and database services run on different devices. We classify the rich-client based services in to two sets. They are backend dependent client service solution and stand-alone client service solution.

Backend Dependent Client Service Solution:

Backend dependent client service solution is same as thin-client based service solution except that here the service solution is not built to run on common presentation services like web browser, instead they are build as specific software application services. Services for the backend dependent client services are shown in Figure 23. The backend dependent client service solution can be implemented as a single mobile application or a number of small applications. The backend dependent client service solution uses RFID application services and implements the minimal set of services as provided by the general presentation services needed to run the applications.

An application is composed of services such as user interface (UI), parsing, security, communication, processing and exception and error handling. An UI service does the tasks to display the information to the user in a user friendly manner, catch user key inputs and validating input data. Parsing service parses the data it gets from server and also it composes data that it sends to server in a specific format. Security service takes care of security issues involved in communication and access control at the client side. Communication service performs the communication between the client and server and organizes data it receives or sends over a network. Processing service does data processing tasks when it gets RFID data from RFID application services, and also it does some application logic part. For example, it tries to optimize the communication between a server and a client by doing some check before communication and by storing some repetitive data an user needs.

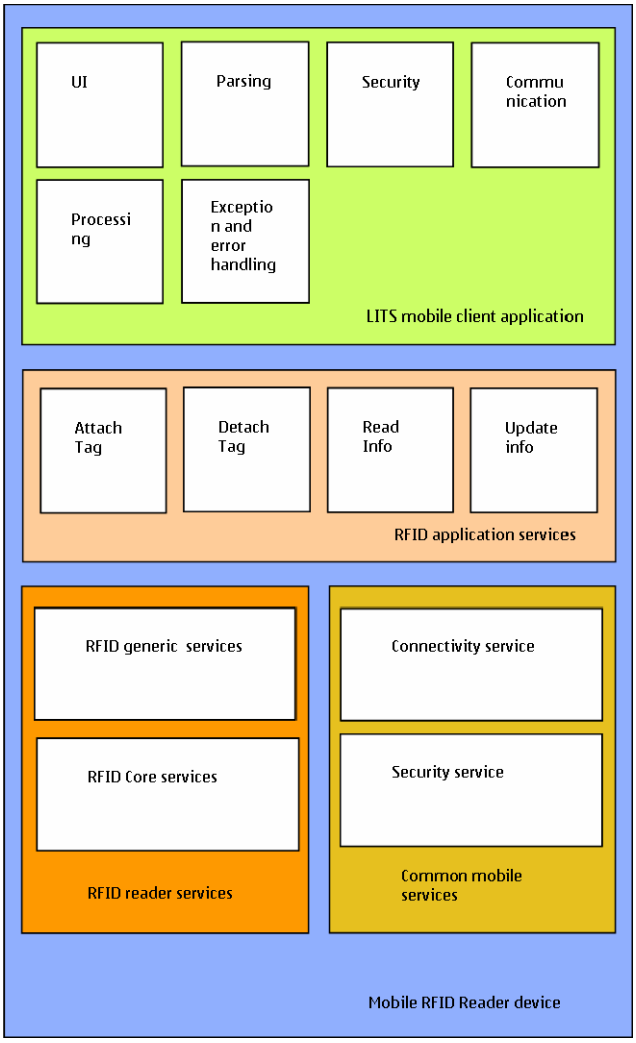

Figure 23 : Backend dependent client services

Stand-Alone Client Service Solution:

Stand-alone service solution doesn't completely rely on any backend server and data base services or communication services. In the absence of communication services, these services still run as expected. But periodically they need to be connected to backend server and database services to maintain data integrity and quality of services. A stand-alone service solution can be also implemented as few software applications or one single application that run on the mobile RFID reader device. It can also be configured to behave like a backend dependent client service solution which completely relies on the backend and database services. Figure 24 shows the included services of this solution.

In addition to the services that we have seen in the backend dependent client services, there are two more services needed to make a service solution stand-alone. They

are the local storage services and synchronization services. Local storage service runs on a mobile RFID device that would maintain RFID data locally rather than relying backend database services. Local storage service tasks include storage and retrieval of RFID data using local resources and local services from the mobile service platform [JCP06]. Synchronization service is used when there is a need for synchronizing data between a mobile RFID reader device and a backend server database. This may happen once or twice a day in practice. After this, local RFID data and database backend data are both up-to-date in backend as well as in the device.

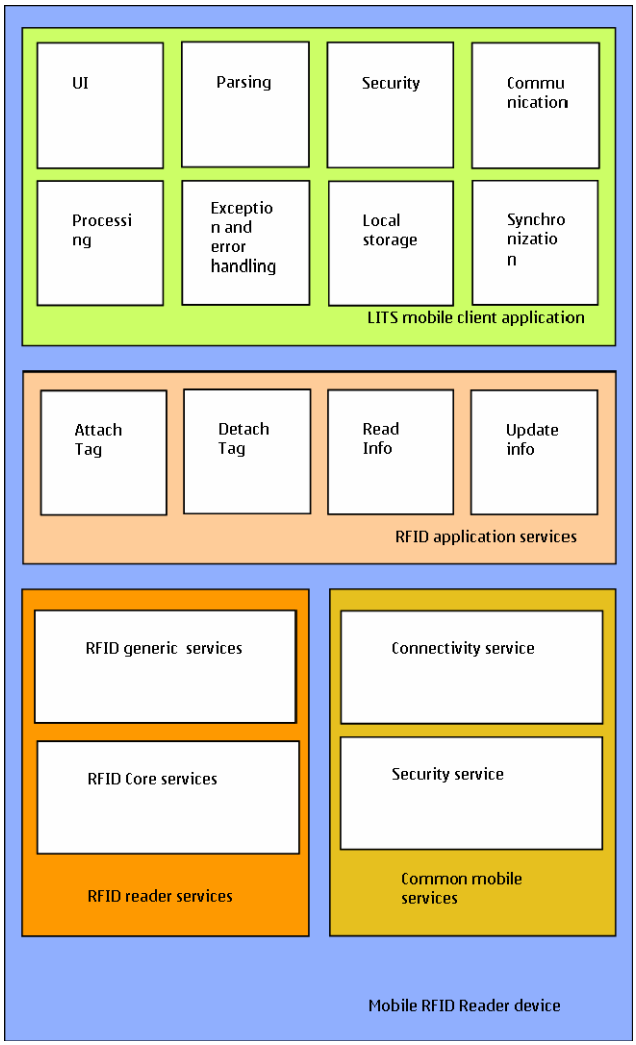

Figure 24 : Stand-alone client services

There can be also a configuration service which could be run on a mobile RFID reader device to configure other services. By configuring services, a service solution can benefit the flexibility of making use of alternatives.

4.7 Design and Implementation of the Selected Solution

The architecture models we discussed in the previous sections provide us with options to choose the architectural solution for the LITS system. We use the Internet & intranet - UMTS based combination alternative (refer section 4.5.4) for our network configuration and use the service architecture model (refer Figure 19) for our mobile RFID service architecture. At the server side, we use the architecture as described in Figure 20. For mobile RFID client side services, we use a "quasi rich-client service solution" like backend dependent client service solution (refer section 4.6.2). The reason for this conclusion is described below.

We chose the internet & intranet - UMTS based combination alternative (section 4.5.4) in order to make use of both the internet and intranet services to communicate between client and server devices on these networks. This also helps to utilize the mobile network in an efficient manner. A mobile RFID device could use either an UMTS network or a WLAN network to communicate with backend services. By keeping the web services inside the intranet we serve subscribers the best inside the intranet, and also at the same time, we don't open any security holes to the outside world. We keep the LITS server and database in the demilitarized zone of the corporate extranet network to connect to the internet. Thus we protect it behind a firewall and use the security shield services provided by the corporate.

The LITS server provides its services on two channels. One is the web services it provides on the internet for web clients. The postal staff and the librarian access the LITS web services from their office workstations. The web services are protected by user authentication and the content is served over HTTPS. The other channel is the mobile RFID services the LITS server provides for its mobile RFID clients. The postal staffs access the mobile RFID services on their mobile phones via the mobile operator network. They must subscribe to the operator services for the mobile operator network services. The mobile RFID services provided via the mobile network are protected by authentication and the content is served over HTTPS. The library web server and proxy provide the services

to the subscribers on the intranet. Thus the selected network architecture model provides the appropriate services for the RFID solution, the right security model and a good cost to benefit ratio.

When it comes to the mobile RFID client service solution, we selected the backend dependent client service solution (refer section 4.6.2), because we have considered the following factors

- In the future, it is easy to extend the backend dependent clients service solution to stand-alone client service solution (please compare Figure 23 and Figure 24 ; we need only few more services and they can be glued very easily).
- The thin-client based service solution is heavily dependent on backend server database and network. Hence, if there is a single point of failure in the network or server it affects the whole RFID service provisioning.
- The thin-client based service solution increases the network communication cost and security risks.
- The amount of effort that we need to put to develop the thin-client based solution is almost half of the amount that we need for the rich-client.
- In the thin-client service solution, gluing the services with the browser would create some noticeable effect to the user and it takes the user's attention for every RFID service access.

Technology wise we made some analysis about the recent trends in the related area and selected some of them as listed in Table 5.

The backend server can run some web server like Apache web server software which would provide web services. Logging, security, and authentication services come along with the Apache server and can be easily configured for our need. For email service, we can use some freely available email service. MySQL can be used for database services. Other LITS services can be designed and implemented using PHP. The HTTPS can also be configured in Apache. The business logic of the LITS should be designed in a way that it can be easily extended for the future and can be reused.

Technological elements	Options	Selected, options for the solution	Reasoning
RFID Tags	Active, Semiactive and Passive tags: NFC, EPC Class I Gen II	EPC Class I Gen II (passive tag)	Passive tag needs no power. Can be read from a distance, are widely used, and EPC reader can be integrated with the mobile device with WLAN and UMTS radio.
Reader	NFC, UHF, UHF reader integrated with mobile device	UHF reader integrated with mobile device	UHF Reader is selected for reading EPC tag type. However, an integrated reader with NFC, UHF, and barcode technologies could also be used.
Reader device connectivity	ZigBee/Wibree, Bluetooth, WLAN, UMTS	WLAN, UMTS	Mobile devices with WLAN and UMTS are commonly available and can easily use the backbone network in a cost effective manner.
Protocols	HTTP, HTTPS	HTTPS	Provide security at the transport layer level using TLS.
Content exchange	XML, Plain text	Plain text	To reduce the processing at the mobile device and to fasten the RFID service response time.
Mobile Client	Browser, AJAX, J2ME, Widgets, Mashup	J2ME	We chose Backend Dependent Client services form our architectural analysis.
Server side	PHP, J2EE	PHP	For faster development of services.
Backend Database	MySQL, Hash table, XML	MySQL	Open source and relational database.
Architecture technology model	Web service, SOAP, REST, EPC	REST and web service combination	Tailored to fit the application needs and well suitable for mobile device.

Table 5 : Technologies for the solution

All of the mobile RFID device level services can be designed and implemented using J2ME technology except the driver which talks to the UHF RFID reader. It can be integrated with the smart mobile device via USB or some other interface. The driver can be implemented in C language and can provide a Java interface to the RFID core services in the form of some J2ME APIs. The generic RFID services can also be designed using J2ME and can provide APIs to the upper layers. RFID application services can be designed as J2ME libraries. The RFID application services and reader services can be combined and implemented as a middleware library. If this middleware is implemented in J2ME it can be reused. This middleware should also be designed to provide room for other item level identification technologies like NFC or barcode. LITS mobile client application can be designed as a single MIDP application or as a set of MIDP applications.

We have implemented the solution and used it successfully. The trials conducted have been very successful. The RFID service response was quick and the users were glad to use the services. Based on the experiences, we have also reused the service framework and model to implement a shopping application demo in a couple of weeks time and it was also very successful. We have also implemented a solution for NFC tags and it also went well.

4.8 Challenges

The main challenges we faced was at the system analysis and conceptual architectural solution part. It took a lot of time to understand the current library system in use and we had to wait until a user experience team (a small team in our laboratory) made some user interaction studies to figure out what exactly the customers want. Otherwise we also had some challenges in finding how to provide access to the postal staff that doesn't have access to the corporate intranet. At the same time, we needed also to use the corporate information for identifying the subscribers without exposing the information to the outside world. We also had to figure out how to use the existing library system's services and corporate services and workflows that are spread out across different geographical locations. When it came to the wireless network access, we had to solve issues related to how, when and to whom to provide the WLAN access and UMTS access. Besides technical challenges we had also the cost and schedules. We had to design and implement not only the RFID service but the whole system within certain budget and time. We

decided to use J2ME technology, MySQL, and PHP even though they were all new for the team members. Fortunately, we learned them quickly to use them for our implementation. The RFID services had also to be designed to be reusable and modifiable. For example, the services should be atomic and be independent of the other layers, services, platforms, hardware, and database. It was a big challenge. The solution also needed to be easily portable to other platforms like s60.

4.9 Evaluation of the Solution

In this section we evaluated the implemented mobile RFID service solution of the LITS system. The whole LITS system has been very successful in providing all the RFID services to its mobile clients and web clients as expected. Thus it met all the functional requirements (refer section 4.2). 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. We evaluate the solution by considering the functional and non-functional requirements (refer section 4.3) of the system [KKC00]. When the functional requirements of the RFID services are concerned, we found out that all of them were met. All the expected RFID services were fully functional without any defects. The non-functional requirements such as runtime qualities, non-runtime qualities, business qualities, and architectural qualities are evaluated in Table 6, Table 7, Table 8, and Table 9 respectively.

The evaluation tables show how the solution fits to the problem. The first column lists the serial number of the quality attributes in every table. The second column of the tables list the name of the quality attribute we evaluate against the RFID services. The third columns “evaluation results” explain the findings. These findings are based on some tests we conducted to measure the services. The last columns in the tables represent the grading. Grading can be one among the four possible values namely excellent, good, fair or poor. Excellent is the highest possible grade, which means that the RFID services are exceeding the value more than what is normally expected. Good means that the services are performing as expected. Fair means that there is a room for improvement. A fair value also means that the RFID services are not performing according to the quality attribute value as expected in all circumstances. Poor means there is really something lacking with the service with respect to the expected quality value.

Serial No.	Runtime Quality Parameters	Evaluation Results	Grading (Excellent, Good, Fair, Poor)
1.	Performance	The RFID services respond quickly on the mobile RFID device without significant delay. It took averagely around 1.5 seconds to respond. It doesn't hang or crash the phone.	Good
2.	Security	The services are protected well from any security vulnerabilities. It provides transport layer level security (HTTPS) and content level security (authentication). While testing, it was proved. The WLAN access points are protected by WEP keys.	Excellent
3.	Availability	The RFID services are always available when UMTS network is used. However there was some service interruption when WLAN is used as the WLAN coverage is not good in all the office areas. Failure of the communication service can cause the mobile RFID services to halt.	Fair
4.	Usability	The postal staff easily learned to use the services. They were easily used to adapt the system into their normal workflow. They are very happy about the sorting use case, because it did not need any user input at all; and the response was so quick. There were few times the services popped some errors that the users did not understand.	Good
5.	Interoperability	The interoperability was verified against web services and it was excellent. The services were tested on different platforms at the client web services and server side. New applications could easily use of the existing system.	Excellent

Table 6 : Evaluation of the RFID services against runtime quality attributes

As we can see from the tables that most of the quality parameters are evaluated as "excellent". But some of them are "good" and one of them is "fair". The fair value for the availability parameter can be improved if we had used stand-alone client service solution architecture for the mobile RFID reader device. It is because in stand-alone client service solution, the mobile RFID services are not network dependent and can be stand-alone during the failures of network services, backend services or database

services. The stand-alone service solution will also improve the performance and usability of the RFID services. So by using a quasi rich-client solution we achieved most of the desired quality attribute values for mobile RFID services.

Serial No.	Parameters	Evaluation Results	Grading (Excellent, Good, Fair, Poor)
1.	Modifiability	For other application the same services were successfully used without much modification.	Excellent
2.	Portability	The RFID services were ported on NFC phone with less effort.	Excellent
3.	Reusability	The same services were easily ported to NFC phone as well as most of the services were reused to create another (shopping) application. Only certain part of the system needed modification.	Excellent
4.	Integrability	It was very easy to integrate the system as it was using layered architecture with Java APIs and web services. The services had been developed independently.	Excellent
5.	Testability	It was easy to test the web services. But hard to test the middleware or lower level services and needed extra effort. However the layered solution made it easier to test the services by simulations.	Good

Table 7: Evaluation of the RFID services against non-runtime quality attributes

Serial No.	Parameters	Evaluation Results	Grading (Excellent, Good, Fair, Poor)
1.	Cost and Schedule	The cost of the system was less than expected. It used most of the available services from different systems, platforms and networks.	Excellent
2.	Marketability	The system can be competitive with the existing systems in the market, because this uses the mobile phone as a reader device. It can employ NFC or any other mobile device with RFID UHF reader.	Good
3.	Appropriateness for Organization	This system is really appropriate for the organization as there is a huge need to track the items in the organization. The stakeholders don't need any special skills to use the system.	Excellent

Table 8: Evaluation of the RFID services against business quality attributes

Serial No.	Parameters	Evaluation Results	Grading (Excellent, Good, Fair, Poor)
1.	Conceptual Integrity	The RFID services are architected by creating the overall LITS system structure (backend, database, network, and device) and its services. Thus it is composed from a number of small architectural structures.	Excellent
2.	Correctness	The mobile RFID service architecture satisfies all the functional and non-functional requirements of the proposed system.	Excellent

Table 9: Evaluation of the RFID services against architecture quality attributes

5. Conclusion

The first section introduced the outline of the problems, solutions, and scope of this thesis. It explained the needs and opportunities to create mobile Radio Frequency Identification (RFID) service solutions to realize the vision of the Internet of Things (IOT). It also described why such service solutions should use mobile phones as mediators. In addition, it named the principles and methods used in rest of the thesis.

Section two gave an overview about the technical background information on IOT, RFID systems, mobile services and mobile RFID services. It explained the terms and concepts used in rest of the thesis by giving some examples. Furthermore, it described the types of services, types of RFID, and smart mobile device architecture. Thus section two provided the basic knowledge to understand the architectural solutions described in the following sections.

Section three explored some mobile RFID service architecture solutions from a theoretical point of view. This section started by explaining the basic use cases for creating RFID services. Then it analyzed the state-of-the-art architectural styles like REST, SOA, EDA, and Push & pull from the perspective of applying them on mobile RFID service solutions. It also presented the basic mobile hardware and software platforms for providing the RFID services. Further, it introduced a layered service architecture platform for the mobile RFID reader device. Then it classified the RFID service types and explained the rationale behind the classification. Three important actors: humans, devices, and things and their involvement with mobile RFID services were also described there. In addition, it introduced a new concept called a “Quasi-Rich-Client Service Solution” for achieving the best quality attributes. Generally, this section showed how a service solution for mobile RFID services should be designed at various levels and the main factors to be considered during the design phase.

Section four described a case study called LITS, a library item tracking system. It showed how LITS used mobile RFID services in practice. It also explained how such a system could be designed by describing the problem, use cases and quality attributes. Further, it described the system overview in terms of services. It explored alternatives of network configurations and service solutions for mobile RFID readers. It explained how the service solution is selected from a set of alternatives. Moreover, it also

suggested various implementation technologies. And finally it evaluated the solution against different quality attributes.

It was found that mobile RFID 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. On the mobile RFID reader side, there were various issues that needed to be clarified in the solution architecture. This included the future hardware and software changes in the mobile device and changes in the RFID technologies. End-to-end RFID service solutions needed to consider more than one solution to solve the challenges in the mobile RFID reader, network, and server areas. We also 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.

The thesis also showed how the conventional ways of building a software product from use cases can be changed in terms of building software services. In order to do this, during the architectural phase, the functional requirements have been grouped into use cases and then the common denominators were grouped as services. The case study showed that the use cases can be explored to find the services. These services have been further exploded into fine-grained services. It is also evident that in addition to the functional requirements, the non-functional requirements should also be always considered while designing the service solutions. The non-functional requirements reveal the qualities of the mobile RFID services.

Many interesting research areas are open for further exploration in relation to the thesis subject. The end-to-end solutions we considered were assuming only point-to-point RFID service access. RFID services provided by multiple points to a single mobile reader device could be further explored. Other research possibilities include studying different possibilities of data formats which can be used between the mobile client and different servers; finding how the quality attributes affect the data formats used in different service solutions; figuring out how to bring mobile RFID services to the normal user; and how to convince the user to find information about everyday items. Work is also needed in areas such as how to create interoperable standards on RFID tag data structures, network

components and protocols. In addition, RFID security and privacy issues need to be studied further in order to bring these services to the normal user.

Research is also needed to evaluate the exact effect of RFID on fixed and mobile networks. It is very obvious that if people would start using RFID services in everyday life, network traffic would greatly increase. It is necessary to find out when this would happen and how much impact it would have. Further examination should be done on how RFID services will be provided with Next Generation Networks (NGN). There is also the issue of RFID-IPv6 mapping. Studies are needed on how RFID technologies can be integrated into existing and future smart spaces and smart environments. Moreover, in this thesis we focused mainly on RFID to identify dummy objects. But in the future, there will be services that smart devices can provide in a smart environment. Such smart environments will have smart services and smart identification technologies. In that case the user will not have to worry about choosing or identifying any objects or services. Overall, we believe that future research in the area will bring interesting times for mobile users with plenty of heterogeneous smart services that live in ubiquitous smart environments.

References

AFZ97 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.

BaK99 Bass, L. and Kazman, R., Architecture-Based Development. *Technical Report, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, 1999, CMU/SEI-99-TR-007.

BBC00 Bachmann, F., Bass, L., Chastek, G., Donohoe, P., and Peruzzi, F., The Architecture Based Design Method. *Technical Report, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, January 2000, CMU/SEI-2000-TR-001.

BCK98 Bass, L., Clements, P., and Kazman, P., *Software Architecture in Practice*. Addison-Wesley, 1998.

BDK02 Bhide, M., Deolasse, P., Katker, A., Panchgupte, A., Ramamritham, K., and Shenoy, P., Adaptive Push Pull: Disseminating Dynamic Web Data. *IEEE Transactions on Computers*, 51,6, 2002, 652-668.

BEL03 Barbacci, M. R., Ellison, R., Lattanze, A. J., Stafford, J. A., Weinstock, C. B., and Wood, W. G., Quality Attribute Workshops. *Technical Report, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, 2003, Third Edition, CMU/SEI-2003-TR-016.

BFM06 Booth, P., Frisch, P.H., and Miodownik, S., Application of RFID in an Integrated Healthcare Environment. In: *Proceedings of EMBS'06, the 28th IEEE International Conference of the Engineering in Medicine and Biology Society*, New York City, USA, August 2006, IEEE Press, 117-120.

BiI05 Birari, S.M. and Iyer, S., Mitigating the Reader Collision Problem in RFID Networks with Mobile Readers. In: *Proceedings of the 2005 13th IEEE International Conference on Networks jointly held with the 2005 7th IEEE Malaysia International Conference on Communications*, Kuala Lumpur, Malaysia, November 2005, IEEE Press, 1, 6pp.

BKL95 Barbacci, M., Klein, M., Longstaff, T., and Weinstock, C., Quality Attributes. *Technical Report, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, 1995, CMU/SEI-95-TR-021.

BMR07 Ballagas, R., Memon, F., Reiners, R., and Borchers, J., iStuff Mobile: Rapidly Prototyping New Mobile Phone Interfaces for Ubiquitous Computing. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, San Jose, CA, USA, 2007, ACM Press, 1107-1116.

Boh06 Bohn, J., Prototypical Implementation of Location-Aware Services based on Super-Distributed RFID Tags. In: *Proceedings of the 19th International Conference on Architecture of Computing Systems (ARCS'06)*, Frankfurt am Main, Germany, 2006, Number 3894 in LNCS, Springer-Verlag 69-83.

BrD07 Breslin, J. and Decker, S., The Future of Social Networks on the Internet: The Need for Semantics, *IEEE Internet Computing*, 11, 6, 2007, 86-90.

CCC07 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.

CuJ02 Cugola, G. and Jacobsen, H. A., Using publish/subscribe middleware for mobile systems. In: *Proceedings of the ACM SIGMOBILE Mobile Computing and Communications Review*, 6,4, October 2002, 25–33.

DMS07 Diekmann, T., Melski, A., and Schumann, M., Data-on-Network vs. Data-on-Tag: Managing Data in Complex RFID Environments. In: *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS)*, January 2007, IEEE Computer Society Press, 224a-224a.

EPC05 EPCglobal, *The EPCglobal Architecture Framework*. http://www.epcglobalinc.org/standards/architecture/, 2005.

FGM97 Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and BernersLee, T., *Hypertext Transfer Protocol - HTTP 1.1*. Internet Engineering Task Force, January 1997. RFC2616, http://www.w3.org/Protocols/rfc2616/rfc2616.html

Fie00 Fielding, R., Architectural Styles and the Design of Network-based Software Architectures. *Doctoral Dissertation*, Department of Computer Science, University of California, Irvine, 2000.

FiT00 Fielding, R. T. and Taylor, R. N., Principled design of the modern Web architecture. In: *Proceedings of the 22nd international Conference on Software Engineering*, Limerick, Ireland, June 2000, ICSE '00. ACM, 407-416.

Fli06 Fling, B., 10 Things I Learned at Mobile 2.0, http://www.blueflavor.com/blog/mobile/10_things_i_learned_at_mobile_20.php & http://mobile2event.com/, November 2006.

FLR07 Floerkemeier, C., Lampe, M., and Roduner, C., Facilitating RFID Development with the Accada Prototyping Platform. In: *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07)*, White Plains, NY, USA, March 2007, IEEE Computer Society Press, 495-500.

Fow03 Fowler, M., *Patterns of Enterprise Application Architecture*. Addison-Wesley Professional, 2003.

FRD07 Falke, O., Rukzio, E., Dietz, U., Holleis, P., and Schmidt, A., Mobile Services for Near Field Communication. *Technical Report, Ludwig-Maximilians-Universität* (LMU) Munich, Germany, March 2007, LMU-MI-2007-1, ISSN 1862-5207.

FrZ98 Franklin, M. and Zdonik, S., Data in Your Face: Push Technology in Perspective. In: *Proceedings of the 1998 ACM SIGMOD international Conference on Management of Data*, Seattle, Washington, USA, June 1998, ACM Press, 516-519.

FuM06 Fuller, T. and Morgan, S., Data Replication as an Enterprise SOA Antipattern. *Microsoft Architect Journal*, 2006, 8. http://msdn2.microsoft.com/en-us/arcjournal/bb245678.aspx

Ger99 Gershenfeld, N., *When Things Start to Think*. Henry Holt and Co., Inc., 1999.

GKC04 Gershenfeld, N., Krikorian, R., and Cohen, D., The Internet of Things. *Scientific American*, 291, 4, October 2004, 76-81.

Goo07 Google Inc., Google Services for Mobile Devices, http://www.google.com/mobile/, http://code.google.com/android/, 2007.

GuJ05 Gupta, A. and Jode, M. D., Extending the Reach of MIDlets: How MIDlets Can Access Native Services, Version: 1.1, Symbian Limited, June 2005.

Han05 Hanson, J., JavaWorld, Event-driven Services in SOA. http://www.javaworld.com/javaworld/jw-01-2005/jw-0131-soa.html, 2005.

HPL06 Han, M. K., Paik, I. W., Lee, B. H., and Hong, J. P. 2006. A Framework for Seamless Information Retrieval between an EPC Network and a Mobile RFID Network. In: *Proceedings of the Sixth IEEE international Conference on Computer and information Technology (CIT'06)*, Seoul, Korea, September 2006, IEEE Computer Society Press, 98-98.

IbH04 Ibach, P. and Horbank, M., Highly Available Location-Based Services in Mobile Environments. In: *Proceedings of the First International Service Availability Symposium, ISAS 2004*, Munich, Germany, May 2004, Lecture Notes in Computer Science, Springer, 3335/2005, 134-147.

Iso07 Isode, IMAP IDLE: The Best Approach for 'Push' Email, http://www.isode.com/whitepapers/imap-idle.html, 2007.

ITU05 International Telecommunication Union (ITU) Reports, The Internet of Things, November 2005.

JCP05 Java Community Process, JSR 257: Contactless Communication API, Version 0.85, Public Review draft. http://www.jcp.org/en/jsr/detail?id=257, October 2005.

JCP06 Java Community Process, JSR 248: Mobile Service Architecture, Version 1.0, Final Release. http://www.jcp.org/en/jsr/detail?id=248, December 2006.

KCK06 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.

KKC00 Kazman, R., Klein, M., and Clements, P., ATAM: Method for Architecture Evaluation. *Technical Report, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, 2000, CMU/SEI-2000-TR-004.

KLY06 Kim, Y.W., Lee, J.S., Yoo, S.K., and Kim, H.J., A Network Reference Model for B2C RFID Applications, In: *Proceedings of 8th IEEE International Conference of Advanced Communication Technology (IEEE ICACT-2006)*, Phoenix Park, Republic of Korea, February 2006, 4-4.

LeK06 Lee, J.S. and Kim, H.J., RFID Code Structure and Tag Data Structure for Mobile RFID Services in Korea. In: *Proceedings of 8th IEEE International Conference of Advanced Communication Technology (IEEE ICACT-2006)*, Phoenix Park, Republic of Korea, February 2006, 3-3.

LeT06 Legner, C. and Thiesse, F., RFID-based maintenance at Frankfurt airport. *IEEE Pervasive Computing*, 5,1, 2006, 34-39.

LNY06 Liu, F., Ning, H., Yang, H., Xu, Z., and Cong, Y., RFID-based EPC System and Information Services in Intelligent Transportation System. *The 2006 6th International Conference on ITS Telecommunications Proceedings*, Chengdu, China, June 2006, 26-28.

MLM06 MacKenzie, C. M., Laskey, K., McCabe, F., Brown, P. F., and Metz, R., *OASIS - Reference Model for Service Oriented Architecture 1.0*, Committee Specification 1, August 2006.

Mic05 Michael, M. P., Energy Awareness for Mobile Devices. *Research Seminar on Energy Awareness*, University of Helsinki, Finland, 2005.

Mic07 Microsoft Inc., Microsoft Mobile Services. http://mobile.msn.com, 2007.

Nok02 Nokia Corporation, White Paper: Mobile Terminal Software Technologies for the Future. http://www.nokia.com/NOKIA_COM_1/About_Nokia/Press/White_Papers/pdf_files/MobileTerminalSw_wpaper_net.pdf, 2002.

Nok04 Nokia Corporation, Press Release: Nokia Unveils the world's first NFC product, Nokia NFC shell for Nokia 3220 phone. http://press.nokia.com/PR/200411/966879_5.html, November 2004.

Nok06 Nokia Corporation, Press Release: It's what computers have become - the new Nokia N95. http://www.nokia.com/A4136002?newsid=1077775, September 2006.

Nok07 Nokia Corporation, Press Release: Nokia 6131 NFC phone taps into mobile payment, ticketing and local sharing. http://www.nokia.com/A4136001?newsid=1096858, January 2007.

NoS07 Nokia Corporation, Nokia S60 Applications downloads. http://www.s60.com/life/application/displayApplications.do, 2007.

NSA07 Nokia Corporation, Nokia Software Applications for Phones. http://europe.nokia.com/A4144902, 2007.

NSM05 Nakanishi, K., Setozaki, M., Ma, J., and Huang, R., A Java-Based RFID Service Framework with Semantic Data Binding Between Real and Cyber Spaces. *The Second International Symposium on Ubiquitous Intelligence and Smart Worlds (UISW 2005), in conjunction with the IFIP International Conference on Embedded And Ubiquitous Computing (EUC'2005)*, Nagasaki, Japan, December 2005, 365-374.

Off02 Offutt, J., Quality Attributes of Web Software Applications. IEEE Software, 19, 2, 2002, 25-32.

OMB05 O'Brien, L., Merson, P., and Bass, L., Quality Attributes and Service-Oriented Architectures. *Technical Note, Software Engineering Institute, Carnegie Mellon University*, Pittsburgh, PA, USA, September 2005, CMU/SEI-2005-TN-014.

PaP07 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.

PaH05 Papazoglou, M.P. and van den Heuvel, W.J., Web Services Management: A Survey. *IEEE Internet Computing*, 9, 6, 2005, 58-64.

PeL03 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.

PHJ02 Podnar, I., Hauswirth, M., and Jazayeri, M., Mobile Push: Delivering Content to Mobile Users. In: *Proceedings of the International Workshop on Distributed Event-Based Systems in conjunction with the 22nd International*

Conference on Distributed Computing Systems, July 2002, IEEE Computer Society, 563-570.

RFI06 RFID Journal, TwinLinx Proposes to Marry NFC and EPC. http://www.rfidjournal.com/article/view/2908 & http://www.twinlinx.com, December 2006.

RLC06 Rukzio, E., Leichtenstern, K., Callaghan, V., Holleis, P., Schmidt, A., and Chin, J., An Experimental Comparison of Physical Mobile Interaction Techniques: Touching, Pointing and Scanning. In: *Proceedings of the 8th International Conference on Ubiquitous Computing - UbiComp '06*, Springer-Verlag 2006, 87-104.

Ruk06 Rukzio, E., Physical Mobile Interactions: Mobile Devices as Pervasive Mediators for Interactions with the Real World. *Doctoral Dissertation*, Faculty for Mathematics, Computer Sience and Statistics. University of Munich, 2006.

Sei05 Seidler, C., RFID Opportunities for mobile telecommunication services. *ITU-T Lighthouse Technical Paper*, 2005

Sli03 Sliwa, C., Event-Driven Architecture Poised for Wide Adoption, Computerworld, May 12, 2003.

Sun04 Sun Developer Network, Introduction to J2ME Web Services. http://developers.sun.com/mobility/apis/articles/wsa, April 2004.

Sun05 Sun Developer Network, Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI). http://java.sun.com/developer/technicalArticles/WebServices/soa, April 2005.

Sun06 Sun Developer Network, RESTful Web services. http://java.sun.com/developer/technicalArticles/WebServices/restful, August 2006.

TPT05 Thompson, C. W., Pazandak, P., and Tennant, H. R., Talk to Your Semantic Web. *IEEE Internet Computing*, 9, 6, 2005, 75–78.

Wan06 Want, R., An Introduction to RFID Technology. *IEEE Pervasive Computing*, 5, 1, 2006, 25-33.

Wei05 Weinstein, R., RFID: A Technical Overview and Its Application to The Enterprise. *IT Professional*, 7, 3, 2005, 27-33.

WiD06 Wikman, J., Dosa, F., Providing HTTP Access to Web Servers Running on Mobile Phones. *Nokia Research Center*, Helsinki, Finland, http://research.nokia.com, May 2006.

Yah07 Yahoo Inc., Yahoo Mobile Services. http://mobile.yahoo.com, 2007.

YLK07 Yoo, S., Lee, J., Kim, Y., and Kim, H., An Integrated Mobile RFID Service Architecture Between B2B and B2C Networks. In: *Proceedings of the 9th International Conference on Advanced Communication Technology*, Phoenix Park, Republic of Korea, February 2007, 1, 90 – 93.

ZCD04 Zeeshan, A., Chimay, A., Darshan, R., Patricia, C., and Dino, B., Semantic Web Based Services for Intelligent Mobile Construction Collaboration. *Journal of Information Technology in Construction*, ITcon, Special Issue Mobile Computing in Construction, http://www.itcon.org/2004/26, 9, 2004, 367-379.

Appendix A: LITS Use Cases

This section describes some of the important use cases for the mobile RFID services that are used in the LITS system.

1. Attach Tag

The attach tag use case is used for attaching a newly arrived library journal with a RFID tag. It is described in Table 10. The Description in the table lists the steps of the ongoing operations during the execution of this use case. The numbers in the table are in correspondence with the numbers represented in the sequence diagram, which is depicted in Figure 25.

Summary	The post man attaches RFID tags with new journals and updates the system
Actor(s)	Post man
Frequency	New journals don't arrive very often. This use case is used only when some new journals are added to the library. The frequency of this use case may be once in a month or so.
Preconditions	LITS client is installed on the mobile RFID reader. Mobile RFID reader is configured to connect to the network. LITS Server and database is up and running. Post man has the access to use the LITS.
Description	1. Post man brings the mobile RFID reader next to the tag 2. Mobile RFID reader reads the tag 3. Tag answers back to the mobile RFID reader 4. Post man physically attaches a new RFID tag with the new journal and issues the command to attach 5. LITS client asks the LITS server about the list of journals that can be attached 6. LITS server queries the LITS database 7. LITS database answers with the list 8. LITS server sends the list to the client 9. Postman selects the journal from the list and issue the command 10. LITS client sends the information to the LITS server for attaching 11. LITS server creates a new entry for the new tag and magazine entries in the LITS database 12. LITS database informs back the status to the LITS server 13. LITS server informs the status to LITS client 14. LITS client displays the status of the operation.
Exceptions	The network environment must be properly configured. [There might appear some network errors].

Post conditions	All new journals arrived have been entered into the LITS Tags are assigned properly to the corresponding journals and the correct information is updated into the database
Illustrations	Sequence diagram is shown in Figure 25

Table 10 : Attach tag use case description

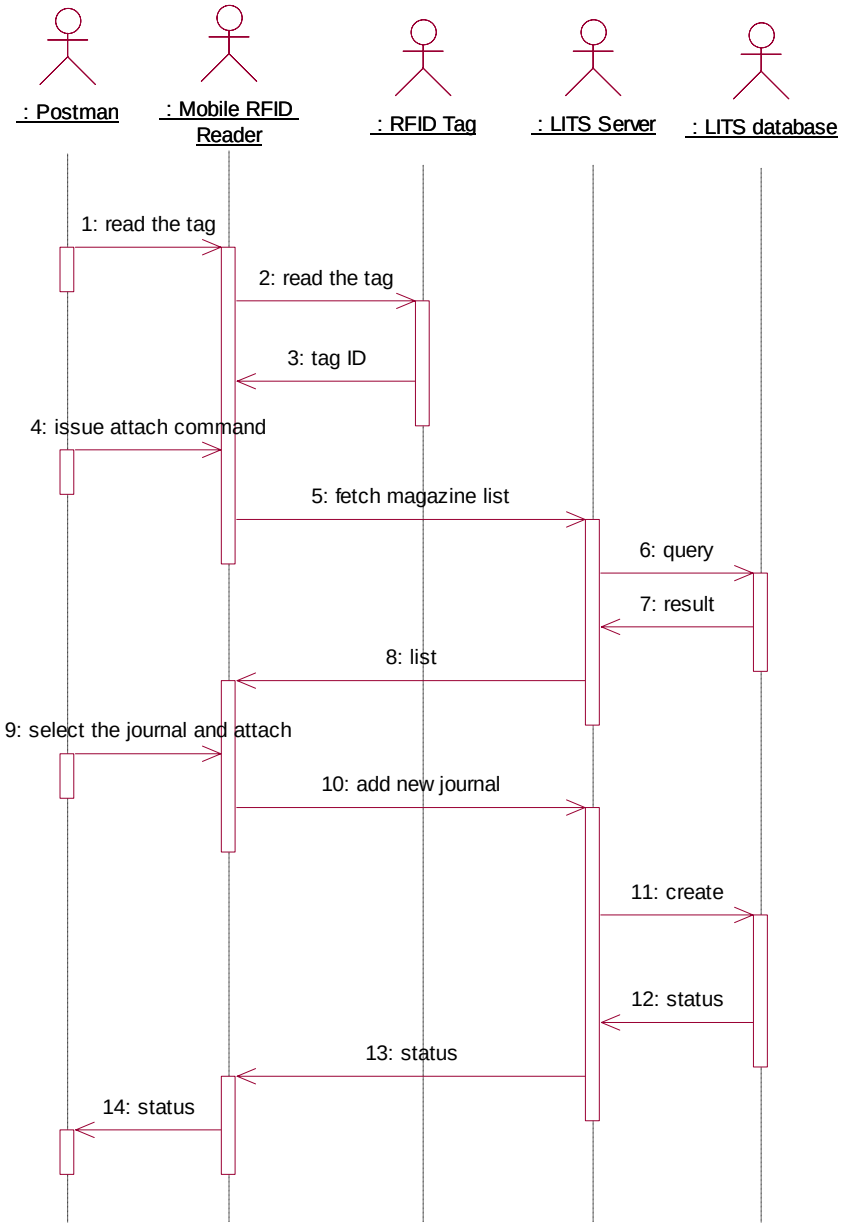

Figure 25: Attach Tag sequence diagram

2. Sort Journals

The sort journals use case is described in Table 11. The Description in the table lists the steps of the ongoing operations during the execution of the use case. The numbers in the table are in correspondence with the numbers represented in the sequence diagram depicted in Figure 26.

Summary	The post man sorts the journals in his office before he distributes on each floor
Actor(s)	Post man
Frequency	The sorting happens at least twice a day
Preconditions	LITS client is installed on the mobile RFID reader. Mobile RFID reader is configured to connect to the network. LITS Server and database is up and running. Post man has the access to use the LITS.
Description	1. Postman wants to sort all the journals to be distributed. He starts to read the journals 2. mobile RFID reader scans for the tag 3. tag replies with its id 4. mobile RFID reader asks the next subscriber for the journal from the LITS server 5. LITS server queries the LITS database about the next subscriber of the journal with the given tag ID 6. LITS database replies to the LITS server the next subscriber info which contains the name and the location 7. LITS server sends the subscriber info to the mobile RFID reader 8. Postman sees the subscriber's location and sorts the journal accordingly in his moving cart
Exceptions	The network environment must be properly configured. [There might appear some network errors].
Post conditions	All the journals to be delivered are sorted out according to the location and the postman's traverse path.
Illustrations	Sequence diagram is shown in Figure 26

Table 11 : Sort journals use case description

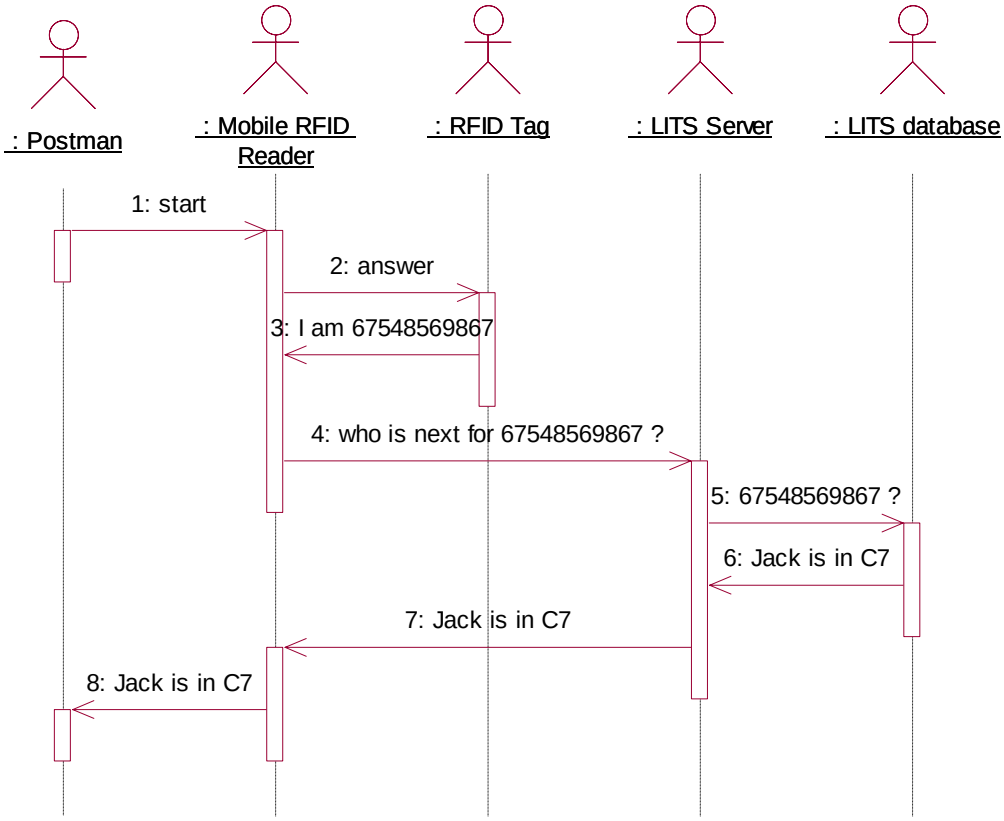

Figure 26: Sort journals sequence diagram

3. Collect Journals

The sort journals use case is described in Table 12. The Description in the table lists the steps of the ongoing operations during the execution of the use case. These numbers are in correspondence with the numbers represented in the sequence diagram depicted in Figure 27.

Summary	Post man collects the journals from the employee's postbox from each floor before delivering them. While collecting the journal, if the next subscriber is found in the same floor, the journal will be delivered immediately.
Actor(s)	Post man
Frequency	The collection and delivery happens at least twice a day
Preconditions	LITS client is installed on the mobile RFID reader. Mobile RFID reader is configured to connect to the network. LITS Server and database is up and running. Post man has the access to use the LITS.

	Journals are in circulation.
Description	1. Post man brings the mobile RFID reader next to the tag 2. mobile RFID reader scans for the tag 3. Tag replies back to the mobile RFID reader with its id 4. Mobile RFID reader contacts the LITS server and asks about the next subscriber 5. LITS server queries the LITS database 6. LITS database answers the next subscriber info which includes the name, location and the status 7. LITS server sends the info to the Mobile RFID reader 8. Mobile RFID reader displays the information 9. Postman instructs the LITS client if the next subscriber is found in the same postbox area. If the next subscriber is on vacation, he would be skipped. Otherwise the next subscriber is the "library archive" 10. Mobile RFID reader requests the LITS server to update the status 11. LITS server updates the state entry in the LITS database 12. LITS database informs back the status of the operation to the LITS server 13. LITS server informs the status to Mobile RFID reader 14. Mobile RFID reader displays the status of the operation.
Exceptions	The network environment must be properly configured. [There might appear some network errors].
Post conditions	All returned journals are collected and the status is updated in the LITS database
Illustrations	Sequence diagram is shown in Figure 27

Table 12 : Collect journals use case description

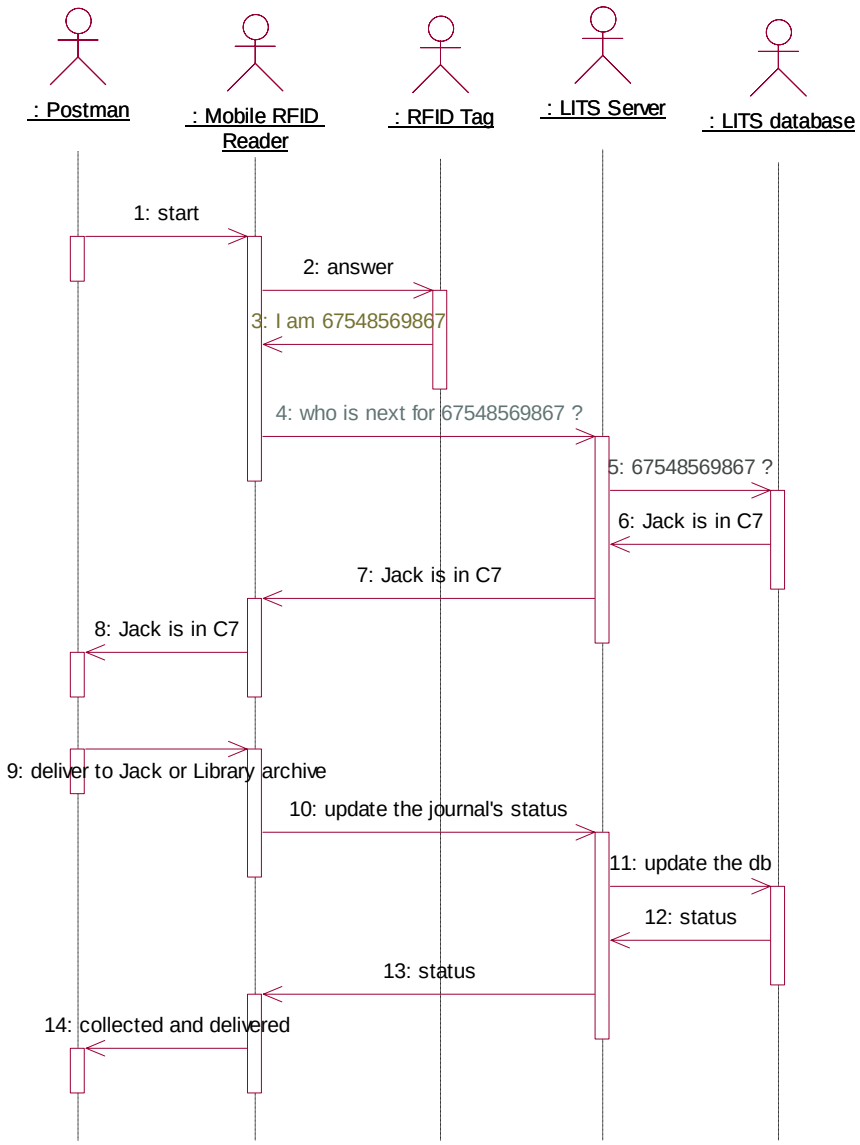

Figure 27: Collect journals sequence diagram

4. Deliver journals

This is same as the previous use case. The difference is that the journal state has to be updated to “delivered” state instead of “collected” state.