Cascading Web Services in Mobile Environments: *Bridging Wireless and Wired Networks for Data Transactions*

Miranda Kajtazi
Bahtijar Vogel
School of Mathematics and Systems Engineering
Department of Computer Science
Växjö University

Master Thesis

Cascading Web Services in Mobile Environments: Bridging Wireless and Wired Networks for Data Transactions

Miranda Kajtazi
Bahtijar Vogel

Supervisor: Dr. Marcelo Milrad
Abstract

Cascading Web Services represent a collection of services offered in a system consisting of multiple devices and multiple interacting platform independent networks. ‘Cascading’ enables Web Services to exploit access in diverse environments without manual intervention. The aim of this thesis is to investigate how Mobile Web Services interact with multiple other Web Services by allowing the generated content to cascade. These services are demonstrated as a technical design solution, in a number of cases within the field of Learning technologies. Communication among devices is preceded using request-response commands by cascading these commands between different Web Services that are self-contained and independent on their context or state.

The system signifies a typical Service Oriented Architecture (SOA) based on a distributed system. Cascading Web Services involve multiple transport networks including Bluetooth Technology, GPRS, Wi-Fi, and Wired Networks. Whereas the protocol of this communication is to bridge Wired and Wireless networks for data transactions, specifically from a Bluetooth location-based network. A number of particular cases will be illustrated in the context of ‘educational outdoor activities’, to demonstrate how the system solution works involving users.

Keywords: Cascading Web Services, Service Oriented Architecture, Bluetooth, GPRS, Wi-Fi.
Acknowledgements

We would like to express our gratitude to our supervisor Dr. Marcelo Milrad, who encouraged, stimulated and helped us with this thesis. We also thank Dr. Milrad and all our colleagues of the “Center for Learning and Knowledge Technologies” (CeLeKT) for giving us the possibility to work on “Data Portability and Media Migration for the Mobile Internet” (MeMiMo) project. We are thankful to Arianit Kurti who gave us strong and initial ideas about the work and especially for encouraging and supporting our efforts of the thesis. Especially, we would like to thank “InterAdria” for the continuous support they provided us during our work with this thesis. We also thank our families for supporting us morally all the time, even if quite long distance separated us, and we thank all friends for spending nice times with us, even during hard working days.
# Table of contents

**ABSTRACT** .................................................................................................................................................. I

**ACKNOWLEDGEMENTS** ............................................................................................................................... II

**TABLE OF CONTENTS** .................................................................................................................................... III

**LIST OF FIGURES** ........................................................................................................................................ V

**LIST OF TABLES** ........................................................................................................................................... VI

1. **INTRODUCTION** ........................................................................................................................................... 1

   1.1 **Problem Definition** ..................................................................................................................................... 1
   1.2 **Goals and Criteria** ...................................................................................................................................... 3
   1.3 **Purpose of the Work** ................................................................................................................................... 3
   1.4 **Realization** ............................................................................................................................................... 4
   1.5 **Limitations** .............................................................................................................................................. 4
   1.6 **Outline of this Thesis** ............................................................................................................................... 5

2. **METHODOLOGICAL APPROACH** .................................................................................................................. 6

3. **SERVICE ORIENTED ARCHITECTURE: CURRENT APPROACHES AND TECHNOLOGIES** ....................... 7

   3.1 **Web Services General Description** .......................................................................................................... 7
   3.1.1 **The Standard of Web Services** .............................................................................................................. 7
   3.1.2 **The Composable Architecture with WS** ............................................................................................... 9
   3.1.3 **Mobility and Mobile Web Services** ......................................................................................................... 9
   3.1.4 **Advantages of Web Services** ................................................................................................................ 11
   3.1.5 **Drawbacks of Web Services** .................................................................................................................. 11
   3.2 **Classic/Commercial Component Systems** ............................................................................................... 12
   3.2.1 **Comparison of Web Services with CORBA** .......................................................................................... 12
   3.3 **Current Applications** .................................................................................................................................. 13
   3.3.1 **Flickr Services** ......................................................................................................................................... 13
   3.3.2 **A Ding Technology** .................................................................................................................................. 13
   3.3.3 **Parlay X Web Services** .......................................................................................................................... 14
   3.3.4 **Google Services** ..................................................................................................................................... 14
   3.3.5 **Apache Raccoon** ..................................................................................................................................... 15
   3.3.6 **Yahoo! Pipes** ......................................................................................................................................... 15
   3.4 **Summary of the Ideas Presented in This Chapter** ..................................................................................... 15

4. **CASCADING WEB SERVICES IN A DISTRIBUTED SYSTEM** ......................................................................... 16

   4.1 **Generic Architecture – Cascading Web Services Architecture** ................................................................. 16
   4.2 **Internal Architecture – Data Flow** .............................................................................................................. 17
   4.2.1 **Message Exchange** ................................................................................................................................... 18
   4.2.2 **How data content is added and updated to the database system?** ......................................................... 19
   4.3 **Web Services – Database Connection** ........................................................................................................ 19
   4.4 **Denotation of generic architecture segments** ............................................................................................ 20
   4.4.1 **Cascading Web Services – Bluetooth Technology** ............................................................................... 21
   4.4.2 **How does Bluetooth work?** .................................................................................................................. 21

5. **RATIONALE AND SCENARIO – BASED DESIGN OF OUR IMPLEMENTATION** .............................................. 23

   5.1 **Case #1 – Scenario** ....................................................................................................................................... 23
   5.1.1 **Flow of Events** ......................................................................................................................................... 24
   5.1.2 **Sequence Diagram for sending and receiving pictures** ........................................................................ 27
   5.2 **Case #2 – Scenario** ....................................................................................................................................... 27
   5.2.1 **Flow of Events** ......................................................................................................................................... 28
   5.2.2 **Sequence Diagram for Cascading Web Services [lookUp function]** ...................................................... 29
   5.3 **Case #3 – Scenario** ....................................................................................................................................... 29
   5.3.1 **Flow of Events** ......................................................................................................................................... 31
   5.3.2 **Sequence Diagram for Publishing a Picture on Internet [flickr]** ............................................................ 31
5.3.3 Sequence Diagram for Retrieving the Published Picture from flickr ................................................................. 32
5.3.4 Image on map – automatic geo-tagging .................................................................................................................. 32
5.4 Advantages of Cascading Web Services Based on Scenarios ..................................................................................... 33
6. TECHNICAL IMPLEMENTATION ............................................................................................................................... 35
   6.1 Software Components - Interoperation .................................................................................................................. 35
   6.2 Development Tools Overview ................................................................................................................................ 36
      6.2.1 .Net Web Services ....................................................................................................................................... 37
   6.3 Research Design – Artifact .................................................................................................................................... 37
   6.4 Inter-ACT – General Class Diagram .................................................................................................................... 37
      6.4.1 Brief Description of Design Classes - Thin Client ......................................................................................... 38
      6.4.2 Web Service [location based] ....................................................................................................................... 39
      6.4.3 Web Service [Cascade] .................................................................................................................................. 40
      6.4.4 Class Analysis [Web-based Application] ..................................................................................................... 41
7. CONCLUSIONS AND FUTURE CHALLENGES ............................................................................................................. 43
   7.1 Results ...................................................................................................................................................................... 43
       7.1.1 Reflections ....................................................................................................................................................... 44
   7.2 Future Challenges ...................................................................................................................................................... 44
   7.3 Conclusions ............................................................................................................................................................. 46
GLOSSARY ........................................................................................................................................................................ 47
REFERENCES ................................................................................................................................................................. 48
APPENDICES ..................................................................................................................................................................... 51
   APPENDIX A – Demo [Thin Client and Web-based Application] ................................................................................. 51
   APPENDIX B - Web Services in a Location-based Environment .................................................................................... 57
   APPENDIX C – Web Services ....................................................................................................................................... 63
   APPENDIX D – Thin Client .......................................................................................................................................... 66
   APPENDIX E – Database .............................................................................................................................................. 69
TABLE OF ENDNOTES ...................................................................................................................................................... 71
List of Figures

Figure 2.1: Method adapted based on "MAYETIC CYCLE" (Holl et al., 2006) ........................................ 6
Figure 3.1: SOA and the Standard of Web Services, Source: (Löwe, 2006) ........................................... 8
Figure 3.2: Data flow/traffic ................................................. 9
Figure 4.1: SOA using Cascading Web Services .......................................................... 17
Figure 4.2: Message Exchange (Text and image) .......................................................... 18
Figure 4.3: Internal Architecture – Data Flow .......................................................... 19
Figure 4.4: User – Web Services – Database Interaction .................................................. 20
Figure 4.5: Bluetooth Personal Area Network, Source: (Kammer et al., 2002) ......................... 22
Figure 5.1: Field group taking a picture of a tree (AMULETS) .................................................... 25
Figure 5.2: Indoor group generating content for the received picture (AMULETS) .................. 25
Figure 5.3: Generic Architecture Segments – Block 1 & 2 ..................................................... 26
Figure 5.4: WS Transmitting message between two applications ............................................ 26
Figure 5.5: Sequence - sending-receiving picture with CWS .................................................... 27
Figure 5.6: Requesting; responding from multiple locations ................................................... 28
Figure 5.7: Searching via CWS – Block 1, 2 & 4 ............................................................ 29
Figure 5.8: Sequence - Lookup function ............................................................... 29
Figure 5.9: Publishing content to Flickr – Blocks 1, 2 & 4 .................................................... 30
Figure 5.10: Calling the Flickr service for publishing a picture ............................................. 31
Figure 5.11: Sequence - publishing a picture in Flickr ......................................................... 31
Figure 5.12: Sequence - retrieving a picture from Flickr to Bluetooth device .......................... 32
Figure 5.13: Publishing image on Flickr from a Bluetooth device ........................................ 32
Figure 5.14: Image on flickr map; geo-tagging ................................................................. 33
Figure 6.1: Image on flickr map; geo-tagging ................................................................. 36
Figure 6.2: Class Analysis [Thin Client] ............................................................. 38
Figure 6.3: Class Analysis [Location based web service] ..................................................... 39
Figure 6.4: Class Analysis [Cascading Web Services] ......................................................... 40
Figure 6.5: Class Analysis [Cascading Web Services] ......................................................... 41
Figure 7.1: Bluetooth – GPRS networks; Nokia Mobile Server ............................................ 45
Figure 7.2: Future Challenge; Location – based network ..................................................... 45
Figure A.1: Login and main screenshots ................................................................. 51
Figure A.2: Messages received ................................................................. 51
Figure A.3: Send activity and picture ............................................................. 52
Figure A.4: Receive activity/picture – Search in multi services ............................................. 52
Figure A.5: Publish and retrieve a picture from Flickr ...................................................... 53
Figure A.6: The main page of the web-based application ...................................................... 53
Figure A.7: Search feature ................................................................. 54
Figure A.8: Creating an activity ................................................................. 54
Figure A.9: Sending a picture ................................................................. 55
Figure A.10: Viewing details of an activity ............................................................... 55
Figure A.11: Publishing a picture in Flickr ............................................................... 56
Figure A.12: Retrieving a picture from Flickr ............................................................... 56
List of Tables

TABLE 1: REQUIREMENTS FOR USER AUTHENTICATION ................................................................. 24
TABLE 2: REQUIREMENT FOR THE DATABASE CONTENT ......................................................... 28
TABLE 3: REQUIREMENTS FOR CWS TO BRIDGE WIRELESS AND WIRED NETWORKS ............... 30
TABLE 4: COMPARISON – CWS & AMULETS BIOLOGY ............................................................... 34
TABLE 5: DESIGN CLASSES [THIN CLIENT] ............................................................................. 39
TABLE 6: DESIGN CLASSES – WEB SERVICE [LOCATION-BASED] ............................................ 40
TABLE 7: DESIGN CLASSES – CASCADING WEB SERVICES ..................................................... 41
TABLE 8: DESIGN CLASSES – WEB-BASED APPLICATION ....................................................... 42
1. Introduction

Electronic information and Communication Technologies have developed very rapidly in recent years. The Web Services (WS) paradigm is currently considered one of the most promising technologies for developing applications in distributed and heterogeneous environments. (Pilioura et al., 2005). Multiple transport platforms of networks correlated with Web Services support intermediate processes, delivery of information, easier and rapid access to database systems. Cascading Web Services (CWS) intend to examine intelligence in a distributed system. Where devices are present and ubiquitous in the environment, by integrating computation and enabling people to move and interact with the system at the same time (Autili et al., 2006).

Ubiquitous computing revolutionized the way of interaction and data transfer between users, by putting information into environment and integrating computation into our everyday life, everywhere and anywhere (Trifonova 2003: Pashtan 2005). Mobile technology that is encompassed by Ubicomp is a new area of modern technology. It supports and handles applications with Web Services, resembled to large applications that run on powerful and traditional desktop computers (Lipponen, 2004).

Communication processes between the mobile world can become more interesting in contrast with traditional static and wired communication, possessing innovative features such as providing the possibility to sequentially use request-response commands through Web services over interoperable multiple networks. Designing such a communication, enables data transmission to be manipulated from wireless to wired networks (JANET, 2006). Existing services, traditional static and wired network systems are distinctive, such as: Bluetooth, Wi-Fi, GPRS and Wired Networks, which de facto do not cooperate with each other at the same time within one system for involving the user to interact with it (JANET, 2006). While with cascading Web services we represent a new technological innovation, creating a huge environment that involves all network platforms to be interconnected. This interconnection goes through communication channels by transmitting data and essentially bridging the wireless with wired networks.

Transmission of information and access to services with this system is ubiquitous. This impacts full user mobility by using wireless devices connected with wired devices. It offers the capability to communicate through distributed systems and also through mobile ad-hoc networks (Xiang et al., 2006). Implementing such a network infrastructure of heterogeneous networks (with connection of mobile devices) can be schematically referred to “pure connection” which means that the mobile device is always connected to the internet by a diversity of technological ways, thus having access to the WWW and other services available for small devices, i.e. through WAP, GPRS, UMTS, Bluetooth, etc. (Trifonova, 2003). The main purpose of this thesis is to reveal interoperability in heterogeneous network architectures, where data transactions are allowed to move from wireless to wired networks, through generated Web services that cascade from one platform transport to the other one.

1.1 Problem Definition

Conforming Web services today are developing many other different techniques based on standards of XML. They are used for different purposes that facilitate the intercommunication between wide ranges of services offered through networks. Many researchers still explore how Web services can be used as open standard applications in distributed and heterogeneous environments. Web services are in focus today and are continually being explored to find different possibilities that they offer. To determine a data interoperability environment, we designed and implemented a component-based
system, which can be easily integrated with other systems that encourages collaboration with diverse network environments. Web services technology offers interoperability and is a flexible technique for data manipulation in diverse networks and systems. WS is a subject which is being developed very rapidly and is being transformed to futuristic shapes that increases high performance and high usage.

User interaction with the system of integrated cascading Web services is transparent to trigger information from one Web service to the other. This technique is performed by using a wrapper Web service which creates an automatic transmission of information, designing the system to interact with users without manual intervention. In line with the main ideas of our thesis (bridging heterogeneous networks through Web services: wireless and wired networks), Pilioura and his colleagues (2005) authored a research paper discovering Web services that support users of wireless devices. According to this paper, the authors state that users of wireless devices, who are also connected to a wireless network, use different Web services that offer personalized services for them. Under their investigation, they show that wireless carriers offer services that allow a variety of information to be presented, such as: weather, stock quotes, news, traffic, or sports updates to be pushed to wireless terminals (Pilioura et al., 2005). This static provision of services, although useful, can not efficiently address the needs of users of wireless devices who want to dynamically decide on the services they would like to consume depending on their individual preferences and content. Our goal is to find and develop an innovative technical solution that creates multiple Web services which interacts and exchanges data automatically with each other. We go further and provide more advantages by simply using Web services technology, by providing a dynamic cascading algorithm for Web services and creating a huge space where information flows from one place to the other.

According to Aziz and his colleagues (Aziz et al., 2006), which primarily describes “Intelligent Wireless Web Services for Construction” provides specific data based on users needs, by giving them flexibility to choose the existing information. The focus of this paper is to exploit the convergence and synergy between different technologies, which has become possible to deliver intelligent wireless Web support, in a particular case, to mobile construction workers. The key point within this paper is to explain how to bridge the gap between the physical world of construction operations and the virtual world enabled by the digital ICT infrastructure.

The idea of our thesis, fully introduces these above mentioned issues that have been explored earlier, whereas our domain problem is to potentially create a wide network of many sub-networks that are interrelated based on different activities performed by users. Comparing to these above mentioned efforts, we introduce a new inventive idea for creating a system of Web services that transfers sequential information. One of our margins within this project is that with our services we allow users to choose or to create the content they want to view; we give the possibility to get the information from any available database and from any available network. No matter if the user uses wireless or wired networks, he/she can still communicate by cascading from one platform to the other.

The basis of this thesis is to generate Web services that operate independently. Appended to WS, the general architecture as a distributed system is particularly taken into consideration. This architecture comprises several important segments that introduce the synergy of the network, exclusively containing the Bluetooth environment that can be directly connected with a wired network, GPRS, Wi-Fi, or any other network. Data flow is another important issue for which we attempt to create a specific communication between request-response commands, which are responsible for the data flow. In the coming Chapters, these issues are discussed separately in order to
accomplish a full understandable interpretation of Cascading Web Services and their performance.

1.2 Goals and Criteria
As we have mentioned earlier, this project is developed with the focus on Mobile Technology. The overall goal of the project is to design and implement Web Services which are adaptable for Mobile Devices. The purpose is to generate Mobile Web Services and access these services from a Bluetooth enabled mobile device by ‘Cascading’ these services between each other.

One of the goals is to implement a technical solution of multiple Web services that interact and exchange data automatically. Another goal is to provide more advantages using Web services technology. The purpose is to provide a dynamic cascading algorithm for Web services by creating a huge space where information flows from one place to the other. The criterion for this goal is to provide the data flow from one web service to another one. The next goal is to incorporate multiple communication network platforms in one system including: Bluetooth, GPRS, Wi-Fi, and Wired Networks where data should pass flexibly between these communication networks. The criterion for this goal is simply to use Cascading Web Services by bridging these transport protocols according to the transmission of data. Another goal is to implement Cascading Web Services (CWS) in a Bluetooth Mobile Device. The purpose is to allow users of limited networks (Bluetooth) to be in interaction with wider ranges of networks. The criteria will be to access CWS, firstly from a location-based server by continuing to GPRS, Wi-Fi or any Wired Network. CWS will be used in the context of Learning Technologies allowing groups of students (indoor and outdoor groups) to exchange data beginning from location-based devices. The criterion for this goal is to exchange different activities or generated contents.

The integration of different Web Services (APIs) from different web based applications is another goal of this thesis. We integrate Flickr services that allow mobile users to publish the generated content from a Bluetooth mobile device to the internet. This procedure incorporates multiple functions for the same data flow: publishing the picture on internet, and mapping the picture (geo-tagging) to the exact position where the picture was taken. These ideas are illustrated by combining CWS and accessing Flickr API and publishing the content on internet and on the map. In this particular case we don’t use GPS (Global Positioning System) to provide the location of the picture. However, we simulated this process by simply putting manual longitude and latitude data for the location of the picture. The practical outcome of the project is to implement two different applications: one mobile based and one web based. Multiple Web Services are also implemented to perform activities between two applications by cascading information from one platform network to the other.

1.3 Purpose of the work
The purpose of this thesis is to explore and design a system that uses multiple Web services in order to allow data transactions between different devices. Primarily, the intention is to allow data transaction to be performed through mobile devices. These devices are connected through Bluetooth, Wi-Fi, GPRS and wired networks. The target of developing this system is to create a technical software solution that can retrieve data content from any device and through different types of communication platforms (Bluetooth, GPRS, Wi-Fi, TCP/IP); this process includes collecting and transmitting data content through Cascading Web Services.

We attempt to create independent communication environments, each having a web server that allows its slaves to manipulate with data as Two-Tier architecture, in order to
have a distributed system divided in some networks. Each network (i.e. Bluetooth location-based) enables the local access of the database and the Web service from the first tier. In fact, the retrieval process of this procedure is much faster considering that this is the first level for accessing one Web service. Otherwise, a user might need to cascade through multiple Web services for finding the proper response. These independent communication platforms (i.e. Bluetooth only, or GPRS only etc.), are interconnected together and shape an N-Tier architecture which allows communication between devices through all transport services. This is based on a Service Oriented Architecture designed as a distributed system.

Web Services offer an infrastructure to interconnect different platforms without boundaries (Pingali & Stodghill, 2006). Combining and implementing Web services within a diversity of communication platforms (Bluetooth, GPRS, Wi-Fi, TCP/IP) is a challenging task that determines the trigger behavior of collecting and storing data content. Thus, our research question can be described as follows:

“How will it be possible to conceptualize, design and implement a system architecture based on Cascading Web Services that can provide services independent of the user’s location and devices?”

In Chapter 4, we describe in more details our initial thoughts regarding the possible solutions to find some answers to the question we are investigating.

1.4 Realization
Accessing Web Services and cascading them from a Bluetooth mobile device is the main objective of this thesis to be developed. This process elicits the procedure of data flow that occurs from an existing service to another service. In one way, the collaboration of data between services starts from a Bluetooth device that by using the HTTP layer it connects with the local Web service. This local service triggers another Web service that resides in another network (i.e. GPRS), where the data transaction is performed. In another way, the Bluetooth device that connects with its local service, is capable to cascade in other existing services (i.e. API’s of flickr, google…), by delivering and receiving different generated content. In fact, this relation is the most important part which is implemented. The second issue after Web services is the generated content. The purpose of this content is to test Web services. The focus of the test is the proportion of different existing content, which contains different data types determining the user to know how to look for data and where to find it. It is generated using MS SQL. This database is chosen, considering its flexibility and its support for data manipulation, control of data and easy transfer activities. The architecture of diverse platform networks is an important matter. Hence, we analyze each part of it to see how we implement the software system and how we bridge wireless networks with wired networks. For details related to blocks of the architecture, Chapter 4 presents this architecture in details in relation with scenarios.

1.5 Limitations
Regarding the technical solution of the project, we have been concentrated on developing Web services that cascade from one another. Multiple databases have also been designed to test how data are exchanged between services. Since the main issue of this thesis is to develop and integrate Web services in multiple platform independent networks, some database issues are not considered:
Each database developed contains information different from the other, but they are not complex and they don't offer advanced features, although they are only used to test cascading Web services.

The search option in the application allows the user to search in multiple databases. The feature of search is accomplished by cascading from one Web service to the other. But in fact, the limitation stands in the point that Web services do not create roots to search between different systems, depending on the content of data. Therefore searching can cause delays until the response is sent back to the user.

Implementation of the system varies depending on resources that we have to integrate the general architecture in real settings. It can be integrated within two transport networks (two web servers consisting Web services and databases) or more.

Fast delivery of information (ex. making the system enough intelligent to know where to search and from where to bring back the result in the fasters manner) to users between different networks is not discussed in this thesis, whereas this point is discussed in the section of the Future Challenge.

Furthermore, we give a number of cases where we show the possibilities of user interaction with the system. But, we will not test the system by involving the user to interact with it. Instead, we test the system on our own and we present a number of cases in relation to the “AMULETS” research project that is conducted at CeLeKT.

1.6 Outline of this thesis
Chapter 2 introduces the main idea behind this thesis, and the state of the art technologies by describing some important issues related to Web services and their standard. It also gives a brief description of other related technologies to Web services. Chapter 3 elicits the architecture of the software system we are suggesting in order to tackle some of the problems discussed in Chapter 2. The architecture is explained based on a Service Oriented Architecture of a distributed system. Chapter 4 presents a number of scenarios related to the generic architecture by signifying general requirements for the software. Each scenario described the flow of events according to the activity by integrating blocks of the generic architecture. Chapter 5 describes the technology used to implement the proposed software solution. It also gives a solid description of generated classes and the inner structure of the Thin Client application, Web services and the Web-based Application. Chapter 6 summarizes the fundamental idea and the results that were obtained based on the work conducted in this thesis. The implementation of the application is also described shortly, by illustrating some of the main developed parts concerning the architecture presented in Chapter 3.
2. Methodological Approach

The development process of the software implementation and the design research presented in this thesis is methodologically based on the cyclic method adapted from the “Mayeutic Cycle” which in this thesis corresponds to the synthetic and analytical phase of a software life cycle (Holl et al., 2006). The Mayeutic Cycle is an approach that helps us to structure the development procedure of our project. In Figure 1 each arrow represents a main phase of the software development process. While the points of the arrows represent the results built from each phase.

![Mayeutic Cycle diagram]

*Figure 2.1: Method adapted based on “Mayeutic Cycle” (Holl et al., 2006).*

Following the rules of this software development cycle, we design an artifact that is useful in different areas. Implementation of this artifact is done according to the diagrams and scenarios, which show the future usage and implementation of the product in one application area where it can be deployed. As a result of this project, we realized that Cascading Web Services lead us to a wide range of applications. Therefore our product can be used in different application areas, since we are building the software that has a multi-perspective view. The deployment of the product can be done for example in medicine, in astronomy, in education and in many other areas. Considering that this thesis is conducted in conjunction with an ongoing project at “CeLeKT”, we explain the system design of the product, according to education purposes (Chapter 4 – Scenarios), because our target is to construct and conceptualize the project for education. Scenarios presented in Chapter 4, are parts of the AMULETS project. AMULETS stands for “Advanced Mobile and Ubiquitous Learning Environments for Teachers and Students”. It supports new ways to integrate outdoors learning in the classroom, using such as: positioning technologies, contextual metadata, digital maps, etc. (Kurti et al., 2007).
3. Service Oriented Architecture: Current Approaches and Technologies

Recently, different efforts and approaches have designed services based on SOA that are becoming more and more utilized rather than using traditional all-in-one systems that narrow the user’s flexibility for making the choice of the content, service and the interaction with a system. The purpose of working and investigating how to integrate different aspects of Web services is to contribute to new series of applications that involve Web services in SOA, which today are evolutionary. This Chapter provides an in depth explanation of Web Services, their performance and their concurrency in the market. We created Cascading Web services by providing special functionalities that allow them to be accessed through Bluetooth location-based devices. The purpose is to integrate limited range networks to be interoperable with wide networks in the same infrastructure. On the one hand we provide a solution where wireless networks can be bridged with wired networks. In this context, “Cascading” is a state-of-the-art technology, which was explored during the development phase of the software system. The effort to make this technology practicable and usable was to bring a new technique that can possibly improve the communication process between diverse networks. Based on this effort, we succeeded in proving the theoretical interpretation of cascading Web services which is elicited clearly in Chapter 3.

3.1 Web Services General Description

Web services offer a new and growing standard for building distributed network applications. They are available services that are offered via the internet that use standardized eXtensible Mark-up Language (XML - specification for meaningful structure of data) messaging system which is used to encode all communications to a Web service (Bell et al., 2007). Web services are simple standards that work independently, regardless of communication platforms, systems and any programming language or operating system. These services can be accessed from different devices, networks or operating systems. Here’s a simple descriptive scenario that explains the sequence of events when a Web service is invoked:

The system sends an XML request to one of the existing services (always connecting to the first tier, then cascading for retrieving further information). The service is called by a given URL that uses SOAP (standard for exchanging XML-based messaging system) over HTTP. It processes the request and returns an XML response to the potential user(s). With this scenario, Web Services are superlative in the context of transparency and synergy. Based on the standard, they are created to work without taking in consideration the difference that exists in diverse infrastructures.

3.1.1 The Standard of Web Services

Web services are widely used for communication between several applications. They are created as components that can be easily attached with one another, and can be easily integrated with diverse technologies (Assmann, 2003; Szyperski, 1999). The standardization of Web services based on XML technologies is loosely-coupled and allows cooperation between all services. They are known as dynamic technologies, communicating through SOAP and they are built on top of open standards including: TCP/IP, HTTP, XML, JAVA. This standard based technology presented in Figure 3.1 lets us create and identify effective strategies and professional development for the software solution.
Blocks of the Figure 3.1 are explained one-by-one to give a brief overview for each process:

1. WSDL (Web Services Description Language) published in a registry of UDDI (Universal Description, Discovery and Integration XML-based protocol).
2. Clients make a query to locate the services.
3. The clients are referred to WSDL documents.
4. WSDL provides data to interact with Web Services.
5. A SOAP request is sent by the client.
6. Web Services responses with a SOAP message.

The concept of cascading Web services, which is clearly described in the problem definition of section 1.1, is dependent on the standard of Web services (W3C). Adapting the inner existing structure of pre-defined Web services standard, we developed a software solution that is inventive in a way. First of all, we explore possibilities of creating new ad-hoc protocols of Web services, which can also be shared and published easily. According to this, we implicate that today, whenever a new idea of Web services flourishes, it immediately signifies a new innovative technique. Whereas in these circumstances the information that circulates within a new system, still passes through different channels and moves without borders. However, in order to make advancements for the data transmission of today’s technology, with Web services and cascading procedure, we try to find a more suitable method. Besides data transmission, this method can link every kind of transport platform and every kind of programming language. Succeeding to implement this method, the theory of interoperability which in a way describes that any software can be coherent with any hardware, is clearly stated in “ISO/IEC 2382-01” standard that says:

“The capability to communicate, executes programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units.”

Based on this standard, our proposed technology can also communicate at the same time regardless the distance and the transport platforms where the data flow is evolved. As a matter of fact, triggering data can be proceeded even in a larger loaded system with many crossroads. To fully understand what is said above, Figure 3.2 is designed to show that information can move in opposite directions passing without intersection in crowded traffics of huge systems.
The figure above describes the transmission of data between different networks that go across a workstation/server. The figure intends to describe these movements with one workstation/server. This means that multiple workstations/servers can be interconnected to perform the same function.

This figure also describes how the data flow appears in a crowded traffic. In Figure 3.2 we present an incoming data to the workstation/server with a black line, and an outgoing data with a gray line. Each incoming data knows its destination and therefore the figure presents different turns of data. But it is clearly seen that there isn’t any collision, although the traffic looks very crowded. With this represented form of data communication and transmission, we explain the possible solution that is built in our system too, which actually performs the same type of data flow.

3.1.2 The Composable Architecture with WS

The generic architecture is the basis of our idea for this thesis for it influenced the development phase of the application, and the structure of it. This architecture presents capacities of wireless networks that can be bridged with wired networks as it will be later presented in Chapter 5. It is designed as a Service Oriented Architecture, purposely to introduce our collection and cooperation of services, with no limits! The architecture is made up of a range of separate proportions that differ in the aspect of data transfer. One segment of the architecture is dependent on the physical cables, while the other part truly relies on the radio-frequency spectrum for data transmission.

The architecture itself introduces a highly composable system that provides recombinant components (WS). They can be used or selected by the user to satisfy the users’ specific request. Verification of this recombinant system is based on these components that can be used independently, considering their status of being self-contained, which is the essential characteristic that makes Web services and the generic architecture to be composable at the same time.

3.1.3 Mobility and Mobile Web Services

New generations of networks support services with “pure connection”, having the capability to provide on-line services without interruptions. Mobile networks are extended and connected to provisions of wired internet to allow the user to access data services from anywhere and from everywhere (Pashtan, 2005). The extension to wired internet happens from wireless networks where mobile users are connected. Nowadays, lots of Web services are available for mobile users provided from internet. Such
services are developed by different companies or organizations that give the user the possibility of using mobile Web services. Existing on-line services can be categorized in:

- General Information
- Travel
- Entertainment
- e-commerce
- Connectivity

For each category there are a number of available services, which today reflect the mobility of users and the wide Web of mobile Web services. Wireless and wired transport networks provide these services, giving the user the possibility to use search engines to find the desired content. As an example of these services, Google has created a wireless search service, purposely to offer services for mobile users (Pashtan, 2005). Several research groups in the world have proposed different ideas related to mobile Web services. Some discuss the architecture and the integration of Web service, some develop frameworks that could possibly support transmission of information through a location-based network, and others give ideas and scenarios explaining how a mobile Web services environment can be managed.

Based on the results presented by Wonsuk Lee (2006) and his colleagues in the paper “Intermediary based Architectures of Mobile Web Services” we could identify some of the challenges related to the development of mobile services. This article stresses out the adaptation of mobile Web services with standards of data transmission through Web services. They argue that legacy technologies and existing standards need to be expended to fit in for mobile Web services. Considering the challenge of our project we explain how we provide Cascading Web services by creating mobile ad-hoc networks and infrastructure of heterogeneous networks. This corresponds with the idea presented in the paper of Wonsuk Lee and his colleagues, where they mention that a mobile Web service (MWS) consists of two types of networks such as those introduced in our thesis.

An important matter presented in the paper, is the intermediary of MWS. Lee’s goal is to demonstrate the integration of MWS and other mobile applications. With intermediary, they explain the architecture of MWS designed with SOAP/HTTP request-response handling that returns results to the mobile device user in a supported format.

The worked carried out by Auletta and his colleagues (2006), allows us to understand different challenges when using Web services through a Bluetooth location-based network. Their intention was to propose a framework that allows applications to directly interface Web services environment, without restrictions and in a very transparent way. Their proposal was to generate an application written in JAVA that invokes Web services from mobile devices connected with a Bluetooth network. Their first experiment using their framework measured the discovered delay that happened when a Slave wanted to find and to connect to the Master. However they have improved their framework, integrating a cache mechanism in order to reduce the delay. Furthermore, they have performed other tests to evaluate the framework in general, and they concluded that their experiments of the framework correctly use different communication modalities that are provided by the Bluetooth specification, making their framework small enough which can be supported by mobile devices. In the context of service-oriented computing Dorn and Dustar (2007) provide a specific scenario, based on the user-related context (i.e. location, activity), service-related context (i.e. system capabilities) and task-related context (i.e. team, roles). Their scenario is depicting an algorithm for “scheduling meetings”. They represent Web services as
abstract services that can be implemented in a distributed or a central fashion. Based on their foundation, such an algorithm of using Web services has enough logic, although it requires having access from other devices and other Web services that can fetch information from different locations. The purpose of this scenario is to present the hierarchical context information highlighting the retrieval process from the nodes connected in the server (Dorn & Dustar, 2007).

In support of the ideas explored in this thesis, the above mentioned papers enable us to examine our contribution to understanding mobile Web services and Web services in general. We realize that our intention of developing cascading Web services can support different activities in the context of user collaboration with the system by opening a new environment where users interact with the system. This environment is heterogeneous, but synergized based on the flow of information effected by users.

3.1.4 Advantages of Web Services

Web Services interoperability and ubiquity are advantages that not only make Web Services unique for delivery of information, but for providing the service itself. The purpose of this thesis is to interpret Cascading Web Services intensity by numerating their advantages:

- Web Services use open standards which enables the communication to be done between different platforms.
- Web Services are developed as a component-based software solutions, which means they are reusable and can be attached at any other system.
- Programming Web Services is a flexible task. They can be built in existing infrastructures and existing resources that don’t need huge changes.
- Publishing Web Services enables users to locate the service description by showing how the user should interact with it.
- Objects that are geographically far from one another can be linked together, by bridging and allowing them to communicate via Web Services that pass information sequentially.
- Communication is done virtually and directly at any time and any place.

3.1.5 Drawbacks of Web Services

As every existing technology, Web Services have some drawbacks that should be discussed. In fact, they do have some limitations, although for our purpose, Web services don’t limit us to pre-defined margins, since we get to overcome such limitations by simply bridging every transport platform, operating system or programming language. However, some of the issues related to limitations of Web services are as following:

- Availability – on-line applications might not be available when the user needs to access.
- Special Requirements – an available application might not offer all requirements of its users. Some users require more features then those that are available in a generated Web service.
- HTTP – does not guarantee the delivery of each information.
- Performance – when the data flow proceeds between multiple servers, the system infrastructure sometimes gets overloaded by consuming the bandwidth and time.
3.2 Classic/Commercial Component Systems

Today there are a number of technologies which provide functionalities of being self-contained software components, reusable, interoperable and flexible. Below we are presenting some existing well-known component systems:

CORBA \(^{xiii}\) is an independent software component that different applications use it to interoperate in networks. In general, CORBA is a suitable standard for being usable on different environments, including operating systems and programming languages.

COM \(^{xiv}\) is Microsoft’s technology that allows multiple software components to communicate. COM is widely used to develop re-usable software components based on Microsoft’s infrastructure.

EJB \(^{xv}\) is specifically developed for the JAVA platform. It has a server-side component architecture, which is used to develop distributed, transactional, secure and portable applications.

Web Services \(^{xvi}\) are standards that work independently, which interoperate in different network platforms, build on any programming language or operating system. They are based on XML standards including: SOAP which is an XML-based message system, exchanging content through HTTP; UDDI which is an XML-based protocol that provides a distributed directory enabling businesses to discover different services and to describe their own services in a uniformed way by putting them on Internet; and WSDL which is the standard for describing Web services.

.NET \(^{xvii}\) is a technology integrated in the platform of Microsoft. It is used to connect information, systems and different devices through software.

3.2.1 Comparison of Web Services with CORBA

Analyzing the technology that is chosen to implement Web services, we realize that many argue when it comes to choosing a technology that has lots of concurrency in the market today. Why Web services is a question that randomly comes to a mind, when someone knows that similar to Web services there are a lot of other comparable technologies. Discussing this issue, we make a comparison of WS with CORBA technology, which stays almost at the same level where Web services are today. To clarify the usage of Web services in this project, below are discussed some points that show how WS fit for this application and why we didn’t choose CORBA, for instance.

First of all, based on Web services we understand that the design of our project is composed according to the flexibility that is offered for message exchange between multiple transport networks and multiple types of devices.

In contrast with Web services, CORBA is limited when it comes to Thin Clients and especially with Mobile Environments (Gokhale et al., 2002 ). For mobility, CORBA is tightly coupled between the client and the server. While in a mobile environment the user always keeps moving. This means that the network might change when a user changes the location. For this typical activity CORBA is not suitable, and this is because of tight coupling between the client and the server (Gokhale et al., 2002). Comparing to CORBA, Web services are ideal for Thin Clients. They use SOAP messages which are well-suited for our case, since with SOAP the intention is only to transfer content, and Web services don’t require high memory capacity or special libraries that might be needed in Thin Clients to support SOAP. In contrast to Web services, if we would use CORBA, each Thin Client needs to support ORB libraries.
requiring a faster processor and a larger memory. In a way, this is generally investing in a device to make it work and perform better (Gokhale et al., 2002).

3.3 Current Applications
In the sub-sections below, we emphasize some state of the art projects and ongoing efforts for SOA that are closely related to our thesis and that are important for some features of our thesis.

3.3.1 Flickr Services
Flickr is a digital photo sharing Web site that uses Web services. It recently has released a number of API services that can be used by the third party. We integrated these services in our system, considering the procedure for publishing and retrieving content from online sources (in our case flickr’s Website). Flickr is a Web 2.0 application, which doesn’t just provide online services from its Web site, but also allows users to integrate its Web services in other applications, by using the flickr system for repository. The aim of flickr is to provide interconnectivity to users by allowing users to publish pictures related to the content of the pictures. Some of its most well-known features are: meta-tagging and geo-tagging. They allow its users to publish pictures by generating a specific content and at the same time by positioning the picture on a map. Except these main features, flickr allows its users to tag photos for RSS feeds. For example, a user publishes a picture with clouds, he/she tags this picture with the tag “clouds”, other interested users in the same picture that also see the RSS, generate other tags for the same picture, such as “weather” xviii. Based on these efforts of flickr developers, we use one of their recent released API, which enables us to combine their services with our own generated services. Using these existing flickr APIs xix we can create the content on a Bluetooth location-based network and we can publish the same content on flickr services, by geotagging this content, and by putting it on the map at the exact position where the content was generated. The same available flickr service allows the user to retrieve any content from a Bluetooth location-based network or any other network, by cascading from a flickr service to our services.

3.3.2 Adnota Technology
Adnota is a relatively new innovative technology that was founded in 2003 xx. This technology is a solution suited for personal information on mobile devices. Recently, Adnota is updated with new features, such as providing location for information and expanding the EXIF tags. It is based on the Symbian OS, which has several products designed for different purposes. Adnota’s Key features are as described as follow:

- Easy recording of Voice Memos
- Multimedia Integration
- Location of Information
- Voice Retrieval and Search Functions
- Messaging
- Categorization
- Dictation Oriented Interface

These key features are followed by different sub-products of Adnota’s technology. The most innovative feature of Adnota is the voice search engine. It is similar to other
engines (i.e. Google Search Engine) for some reasons: feature extraction, end point
detection, special distortion measurements and time alignment. This special feature is
integrated in Mobile Devices based on Symbian OS, which searches on different
databases, by bringing the result in an ordered list, firstly, showing the best match of the
search result.

As we can see, both ADnota and Flickr have some similar functions on which we
base the development of applications. We analyzed how ADnota works, although we
have integrated a Flickr API to expand the horizon of cascading Web services, since
ADnota does not provide us with APIs yet.

3.3.3 Parlay X Web Services
Parlay X Web Services are developed in order to enable development of telephony
applications and “telecom-enable” IT applications. Their intention is to develop
applications for the next generation of network applications. Parlay X Web Services are
known as simple and powerful services that can be used to generate innovative
applications (The Parlay Group, 2002). These services have telecommunication
capabilities, providing functionalities to develop telephone applications. Their message
exchange is based on XML technologies. They can be developed using common Web
services techniques for development. They are not specific or limited to one network,
but they provide the functionality to interact with more than one network. Generally,
Parlay X Web Services are interfaces for applications that rely on the standard
infrastructure of Web Services. They don’t provide the implementation of
Authorization, Authentication, and Accounting (AAA) (The Parlay Group, 2002).

These services have some specific features which correspond to our thesis. We have
investigated the infrastructure of Parlay X Web Services to understand how they
communicate between applications and in the network. We realized that they can be
developed in multiple programming languages by using Web Services or CORBA
transports, interacting with the Parlay Gateway, which is complex and can exploit the
Parlay API. Following these Parlay features, we introduce them in our system
architecture, involving a number of interoperable networks and a number of Web
services that can be developed in multiple programming languages.

3.3.4 Google Services
Google Web services are developed as a continuance of Google Search Engine. The
purpose of Google Web services is to allow users to access their existing content from
users system, without going to the Web site of Google. These Web services are quite
essential that help the user to represent and extract data in the form that the user
needs, rather than in the form that Google provides (Mueller, 2004). Google have
developed many APIs, from which we distinguish some of the most used APIs that also
correspond with our system. Google Maps and the Google Search Engine, are also
provided in the form of the API, to allow the user embed these services in their own
application.

Google Maps is a particular service which we have analyzed, according to the
domain problem of our thesis. This service is similar to the idea of Yahoo Maps used by
flickr application, therefore we have investigated both to see similarities and
differences, and we came to the conclusion that both offer the same possibility to meta-
tagging and geo-tagging functionalities.
3.3.5 Apache Raccoon

Recently Nokia invented a new technology integrated in mobile devices by creating a Mobile Web Server in a Nokia device (S60) based on Apache Raccoon. It has advanced features which allow the user to do content generation and put it in a Web-site on the mobile phone. Nokia created a Web-application that prompts the phone owner to take a picture which subsequently is returned as a JPG. That is, on a personal device the Website can be interactive.

We have investigated this new technology, to see what kind of possibilities it offers to integrate it as a server that would have function as same as other Web-servers where Web services and databases reside. We wanted to bring a new solution where a hand-held mobile device acts as a Web-server, allowing other hand-held devices to connect via a Web service placed in this mobile web server. The problem was that the Apache Raccoon is limited only to acting as a server for publishing a picture on the Nokia’s Web-site. Yet, this server does not provide high quality features, such as creating a Web service and storing it in Apache Raccoon.

3.3.6 Yahoo! Pipes

Yahoo! Pipes is a free online Web-based application which offers services to use its Pipes that collect and mix structured data, from more than one resource, such as feeds. It is nominated as a high-tech innovative application, especially for its simplicity and the interoperability. Pipes combine a sophisticated development environment with features such as the ability to clone or share the web services you produce. Pipes are used to build services that can easily be integrated in any application, without having to write a line of code. One of its features is to make your own Web services public, to use others Web services or to offer your own Web services to others. We have examined Yahoo! Pipes to understand their role in being interoperable for the data flow of structured content of multiple web-based applications.

3.4 Summary of the ideas presented in this chapter

In this chapter we introduced the Web services paradigm and the state of the art technology regarding Cascading Web services. We have explained different component-based systems related to our thesis and the component itself which makes our project flexible and more innovative. We also gave a brief description of a number of web based applications and the possibilities they offer for data exchange based on APIs using web services. In the coming chapter we describe the generic architecture of the system we have designed and implemented, building it based on a Service Oriented Architecture.
4. Cascading Web Services in a Distributed System

This Chapter presents Cascading Web Services (CWS) architecture which is proposed and designed to fulfill the purpose of solving the defined problem for this thesis in the previous Chapters. We also describe the data flow where CWS plays the most important role. The system design we are exploring is based on a Service Oriented Architecture (SOA) in a distributed system that includes multiple Web services performing data transactions between diversity of devices that are connected through Bluetooth, Wi-Fi, GPRS and wired networks. The system is used to potentially create a technical solution that can retrieve data content from any device and through different types of communication platforms (Bluetooth, GPRS, Wi-Fi, TCP/IP). The function of Cascading Web services, in diverse network architecture is about collecting and transmitting data content between multiple users of heterogeneous networks.

We decided to choose this approach, where multiple Web services play the key role for the composable architecture of SOA, comparing to other solutions studied in Chapter 2, and we understand different usages of SOA for different approaches, although our system differs in the aspect of providing multiple Web services in heterogeneous networks at the same time. Independent communication environments have personal web servers that allow their slaves to manipulate with data as Two-Tier architecture. Interconnected transport platforms present a distributed system divided in some networks such as location based (Bluetooth) so the user can access the database and Web services locally from the first tier which in fact the retrieval process is much faster considering that this is the first level for accessing one Web service. Otherwise, a user can cascade through Web services for finding the proper response.

These independent communication platforms (i.e. Bluetooth only, or GPRS only etc.) designed in Figure 4.1, are interconnected together that shape an N-Tier architecture. This allows communication between devices through all transport networks based on Cascading Web Services. Interconnection of networks is based on Service Oriented Architecture (Erradi et al., 2006). In general, Web services offer an infrastructure to interconnect different platforms without boundaries. Operation of Web Services is platform independent based on XML as the main feature for communication in distributed systems (Pingali & Stodghill, 2006). Combining and implementing Web Services within a diversity of communication platforms (Bluetooth, GPRS, Wi-Fi, TCP/IP) determines the trigger behavior of collecting and storing data content.

4.1 Generic Architecture – Cascading Web Services Architecture

The relationship and principles of the system architecture play an important role for data manipulation. Therefore we proposed a system architecture design illustrated in Figure 4.1 that shows all communication services for access control, used for controlling transmission of data specifically with mobile devices. This architecture design explains the communication of devices using request-response commands to the web server. It follows these commands cascading from one Web service to the other Web service.

Our system design is based on an architecture level explaining the capacities of wireless network that can be bridged with wired networks through internet. Data transmission works by collecting and storing all the necessary information in databases. Data transactions are done directly by users that store and get data from the databases such as text and images. Communication process, concerning the collaboration of networks for data transmission which is presented in this section is also shown in Figure 4.1. This is an architecture which represents transport layer and session layer, involved for proceeding data transactions.
According to the figure above, the intention of exploring and designing such architecture is to show a group or collection of services that communicate with each other that are self-contained and independent on their context or state. This figure represents a typical Service Oriented Architecture that works within distributed systems. This kind of communication between different networks and devices facilitates and opens new opportunities for data transaction. Cascading Web Services are introduced by involving multiple transport networks including: Bluetooth Technology, GPRS, Wi-Fi, and Wired Networks. The architecture is divided in blocks based on the transport network and the presence of devices. This separation is purposely done to easily understand which parts of the system are used to test the applications as presented in Chapter 4.

Bluetooth connection corresponds to a typical “location based” network allowing the creation of PAN without cables or wires. GPRS as a mobile data service available to users of GSM, transmits - receives data in packets. While TCP/IP is a very suitable communication protocol used here to connect hosts in internet through wire or Wi-Fi. All these network technologies create interoperability that use exchanged information from one network to the other via Web Services residing in web servers having the data placed in databases. Indeed, this is bridging the gap of non-proper communication and data transmission between such networks.

4.2 Internal Architecture – Data Flow
Application logic, distributed data processing and data flow, primarily rely on a connection-based, Two-Tier architecture. Integration of multiple Web Services that cascade for the data flow is N-Tier architecture of the distributed system; data interaction is platform independent; transactions are scalable and support different data content. The Database System will be designed and modeled as a Structure System by using this technique:
Data Flow Modeling which is used to identify and document all movements in the system: activities for data transportation of one database to the other, sending and receiving operations into the system and the general data flow comprehending Web services movements.

4.2.1 Message Exchange

XML data is the defined format of message exchange for sharing information to all corresponding sub systems of the infrastructure with built in Web services. With cascading Web services the pattern of message exchange is by using request-response commands. These commands can be systematically identified by all devices connected with the system. This is the nature of Web services and their capability to provide the same interest (transmitting information) of different groups, meaning that their standard is well-defined, as mentioned in section 2.2.1. As XML is the format used to exchange messages, the system is capable to transfer different file formats which are different and rather more complex, than just simple text files. In our application, we can exchange images of different extensions. Transmission of an image data type is converted in XML text format and is sent as an XML file from a service to a service.

This implicates that transactions between users, pass through XML. Although the conversion of a complex file format for message exchange is encoded to binary, in the client side. Binary is a digital coding system that uses two symbols (0 and 1) to represent different structures of data. Purposely, we use binary message exchange format to avoid problems that might appear due to variants of data types. Binary represents the machine-readable form, which is the reason for involving binary transfers for various data, considering different devices and different transport platforms. Figure 4.2 represents transactions of different file formats from a device to a device in multiple transport platforms.

Figure 4. 2: Message Exchange (text and image)
4.2.2 How data content is added and updated to the database system?

Considering the general architecture of the system, Two-Tier architecture demonstrates one node connected to the distributed system. This connection refers to a typical Client/Server architecture where user interaction with the application sends and receives information within one tier. This interaction with the system allows adding, retrieving or updating the data content. Data can be added and updated in the first database the user connects (Client/Server – GPRS). When the same data is requested by another user of a distinctive network (Bluetooth), cascading Web services look for the registered data in one of the potential databases and respond with an answer.

Inter-process communication of this distributed system handles all transactions proceeded by the user interacting with the system. The data flow network as it is described in Figure 4.3, executes processes sent by the user. Transactions are performed among Web services that automatically retrieve data and return it to the user. The retrieval process of Web services creates a data flow between databases, which interact with each other. This operation creates an environment of multiple independent platform networks that work together as a distributed system where the data flow plays an important role for the overall communication.

4.3 Web Services – Database Connection

To describe the inter-relation between Web services and the database, a scenario is presented below, to fully understand this issue. Our infrastructure presents many platform independent systems, each having a web server. The Web service and the database are placed in each web server. Database system provides all kinds of information in different formats. Users that want to access the content can easily fetch all information needed by simply connecting with the application and accessing the existing services. This lets each user to interact with the system with no limits.

Let’s imagine we have several users who use Wi-Fi technology with Smart Phones and PDAs. Each user can use the application which is especially developed for small-size screens. At the same time, a large Web-based system is developed for normal-size computers. Functionalities that will be provided will let the user to access and add
content to the database, by giving flexibility to the user, who can choose what kind of content to view.

The human interaction process with the system infrastructure is very flexible and gives an opportunity to the user to use multiple functions of the system. Here is a simple scenario that shows how the communication is processed. Indeed, this scenario is developed based on different activities that are initiated by primary school teachers of Växjö region of Sweden, in cooperation with “CeLeKT” Research Center. The idea of such an activity has been raised by these school teachers because they wanted to help and to ease the teaching methods and the education, especially when they perform outside activities. For them, the most typical subject that can be taught outside is biology, as well as astronomy. This sample scenario below in relation to Figure 4.4, demonstrates one activity that can be done using our application:

Users equipped with Wi-Fi devices need to access some information existing in databases of this system. User A sends a request for retrieving information about category of “astronomy” issues that exist in the database. User B requests some information about “biology”. Automatic process between Web services and databases respond to the user either with found data or nothing. Interoperability between networks exchanges all information in a useful and meaning manner. Each query sent over the system primarily is controlled by Web services that communicate with databases. To have a closer view about the interaction process from a user to a Web service, and from a Web service to the database, below is a figure presented, that shows this sequence of events of this activity. These transactions go spontaneously and repeatedly depending on the expansion of the infrastructure.

4.4 Denotation of generic architecture segments

The Core design of the project indicates that the system is voluminous. This demands an explanation of the system, by distinguishing parts of the architecture in terms of communication protocol. The goal of the project, which is to have interoperability between multiple different systems, includes variety of transport networks; therefore we take a close look particularly to the Bluetooth technology, which determines the base steps of our construction of the generic architecture. Clarification of the infrastructure is explained by introducing cascading Web services in a Bluetooth environment. This shows that other transport networks (GPRS, Wi-Fi, TCP/IP), operate similar to Bluetooth, but the difference is that they are always connected to the internet and they capture a very wide spectrum for transmission of the content.
4.4.1 Cascading Web Services – Bluetooth Technology

Bluetooth transport platform is one of the most important issues of this project in regard to developments and inventive idea comparing to constraints and difficulties this technology has. Using this short-range wireless technology, we succeeded to implement it in our diverse network system by making the application of multiple Web Services to function via Bluetooth - location based network by passing information to GPRS, Wi-Fi or any other Wired Network. We have integrated this technology, and we have achieved a progress by accessing different Web services through synchronized Bluetooth mobile devices that transmit information back and forth, from any other transport network.

4.4.2 How does Bluetooth work?

Bluetooth technology is intended to replace cables for short-range communications. The purpose of this technology is to connect portable and/or fixed devices through radio-frequency waves rather than cables. This was initiated to replace cables when such devices needed to be connected in a location based area, and in a very short diameter (ex. in a car, a room, a class…). Bluetooth connection works by sending wireless signals to other devices in the same range. Its fast performance works in a diameter of 10 meters maximum. Bluetooth has some disadvantages concerning the number of users that can be connected and exchange data at the same time, which until now is only 7 users. As a network standard it works with other devices by creating an agreement in the protocol level and in the physical level as a radio-frequency standard. (Kammer et al., 2002)

According to our project, the purpose of building such an application is to incorporate the Bluetooth technology without regarding the disadvantage as mentioned above. Because of this, we presented a couple of scenarios that can show how we think that our application can be deployed. We clearly mentioned that for any activities we use groups of users who connect with the system as a whole. Hence, whenever there is a need of more than 7 users to be connected, we give the possibility that at the same place and with the same architecture, users of other wireless networks such as GPRS can be connected and do the activity. Our purpose is to show that a Bluetooth device can transmit information through multiple networks apart from its limited area connection and its limited number of users. Bluetooth transmits in the frequency range of 2.4 to 2.4835 GHz and achieves data rates of up to 721 kbps.

Figure 4.5 represents a typical Bluetooth network and a number of devices that can be connected with a desktop computer. In our case, and based on our scenario, we use the connection of a PDA to a Laptop and vice versa. We use this connection to transmit information by cascading from a service to a service until the information reaches the destination.
This figure explains the nature of the Bluetooth technology. It presents devices as peers, which are identified by their own 48-bit unique address. This technology is based on the master/slave nature, where one device is set to master and 7 slaves can be connected at the same time. This kind of network is a point-to-multipoint feature of Bluetooth, which makes this technology unique apart from other networks (Kammer et al., 2002).

In this chapter we presented the conceptual architecture related to the problem definition stated in section 1.1, which is the basis of developing and implementing the system in general. Data flow is demonstrated according to the generic architecture by showing how messages are exchanged in such a diverse system including Bluetooth technology. In chapter 5 we give further details related to the design by presenting some scenarios that unfold the system architecture.
5. Rationale and Scenario – Based Design of our Implementation

In this chapter, we take a close look at the requirements of the software system. The purpose of describing requirements is to show what the system should offer, how the data content should be exchanged and how multiple Web services manipulate in the entire network. We also describe some requirements according to the users, by bringing results of how the software system should work in real settings. These requirements are presented in cooperation with scenarios, to show what the application is capable to offer when it is used in a real situation. Moreover, we present these scenarios that give details for some segments of the software solution. Each scenario is elaborated in connection with requirements and each scenario demonstrates a sequence of events according to requirements and sequences of the architecture.

With a scenario we get to visualize and shape the form of the software by having all requirements needed and the bases of the metrics. Therefore some highly structured scenarios are necessary to propose the concept model of the software. At the coming sections, we present some real examples that describe the interaction of users with the system. Furthermore, these scenarios describe all events that are taken during action executions. The aim of these high-level scenarios is to identify and unfold the software system and the architecture system. Below are three cases of example scenarios that describe concisely some m-learning activities which were performed by research team of “CeLeKT” Research Center in conjunction with “AMULETS” project. Using the trial of AMULETS, our intention is to show a high-level description of how our software system would be used in a different manner with possibility to produce multimedia content on mobile devices, particularly providing location-based services (Kurti et al., 2006). With these cases we also show how the generated content is collected in databases and published on the internet.

The main reason for writing scenarios based on education activities, is the fact that we conduct our master thesis in conjunction with a recently started research project at “CeLeKT Research Center” called “Data Portability and Media Migration for the Mobile Internet (MeMiMo)”. The MeMiMo project is closely related to AMULETS project. The research focus of MeMiMo was very much inline with our thesis proposal and therefore it was beneficial to explore the idea presented in this thesis in real settings and in collaboration with the research team at CeLeKT. These scenarios presented below, explore the path of using the software and signify the user interaction with it.

5.1 Case # 1 – Scenario

In this scenario we describe some events of an activity to be performed by two groups of users. These groups of users are separated geographically and are located in different environments. One of the groups is located in a field (Figure 5.1), while the other one is indoors in a laboratory (Figure 5.2). The domain in which activities are taking place is related to the field of “environmental studies” and “tree taxonomy”. The data content (images and text) are generated by these groups of users that flow from a location-based environment to wide ranges networks. Before presenting the flow of events, we provide some general requirements of our software system in relation with this scenario:

<table>
<thead>
<tr>
<th>Identifier: R1</th>
<th>Name: General Application requirements – Cascading Web Services (CWS)</th>
</tr>
</thead>
</table>
**Description:** General requirements for the thin client application and the Web-based application.

### Requirement details

<table>
<thead>
<tr>
<th>R1:1</th>
<th>The software system should allow groups of users to use cascading Web services from multiple transport networks and specifically from the Bluetooth – location based network.</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1:2</td>
<td>The thin client application should allow groups of users to exchange or to publish content in any database system, which is available and which is connected by cascading Web services (CWS). The desktop application should allow the same activities to be performed as in the thin client application.</td>
</tr>
<tr>
<td>R1:3</td>
<td>CWS should allow users to execute queries from a specific place (outdoor-indoor activities), without limiting the connection from one place to the other, even if activities are engaged geographically far away.</td>
</tr>
<tr>
<td>R1:4</td>
<td>Every group should receive the content delivered to them.</td>
</tr>
<tr>
<td>R1:5</td>
<td>The system should be able to manage and use all Web services and databases which exist in the implemented infrastructure. Cascading procedure should be done according to the presence of the content in the database, or according to the generated content.</td>
</tr>
</tbody>
</table>

Table 1: Requirements for User Authentication

The flow of information proceeds from a Bluetooth transport network to GPRS and vice versa.

**5.1.1 Flow of Events**

1. *The field group located in a forest area takes a picture of a tree and sends the picture via Bluetooth to the indoor group.*
2. *The indoor group analyzes the picture and according to taxonomy, they generate the content related to the picture (tree).*
3. *This picture is sent back to the field group, updated with new information.*
According to the generic architecture presented in chapter 4, here we present blocks of this architecture that are in use to perform this activity. Block 1 and block 2 as presented in the general architecture are extracted from it and are designed specifically for this activity.
This particular scenario is also demonstrated in the figure below that introduces the usage of wireless devices interacting with one location of Web services (WS). It shows a dynamic act of the application integrating WS to work as a transmitter. The application plays the role of a conventional entity which initiates the calling procedure of a WS. In fact, the Bluetooth device of the field group disposes the closest WS by requesting it to forward the content. In the other way, the indoor group requests the same WS to reply to the field group with the generated content. The picture below, presents the exact process of data flow of this activity.
5.1.2 Sequence Diagram for sending and receiving pictures.

This scenario describes cascading Web services performed within two different groups of students involved in a biology case-study. They can use this system, which lets them to generate the content according to their activities, and exchange it between groups.

5.2 Case # 2 – Scenario

Here is a description of a similar activity as in the first scenario. Although this scenario describes some other functions of the software system. The scenario introduces the new feature of “searching” that allows users to request for any available content that exists in any network. First we present the requirements of the software system in accordance with this scenario:
Table 2: Requirement for the database content

### 5.2.1 Flow of Events

1. The indoor group makes a request from a location based network.
2. The application sends a request to the closest Server/WS. This request cascades from a Bluetooth network to a GPRS/Wi-Fi network and a wired network, until it finds the result.
3. Web Services collect results extracted from multiple locations and send back the response to the requester.

This activity is designed in Figure 5.6, which provides the example of how data flow happens with a request-response mechanism of the search feature.

![Figure 5.6: Requesting; responding from multiple locations.](image)

To illustrate this particular example based on the generic architecture, we provide the Figure 5.7, to show the blocks used to perform this activity.
5.2.2 Sequence Diagram for Cascading Web Services [lookUp function]

This sequence diagram which is designed according to the scenario above and in relation with Figure 5.8 in section 5.3, shows how these query commands are sent and received in the system; it shows exactly each method that executes the functions for performing the typical activity described in the scenario above and in Figure 5.8.

In the second scenario we describe the automatic catering of Web services. In fact, this is another variant compared to the first scenario, which presents the usage of cascading Web services in multiple locations. This case introduces the provision of Web services pushed through wireless terminals. It shows capabilities of wireless networks that can transmit content in multiple locations through multiple networks including wired networks.

5.3 Case # 3 – Scenario

In this section we describe a specific scenario in order to filter all doubts in relation with location-based networks. The context of this scenario is that the activity takes place in a location-based environment - field (as in Case# 1). But the difference relies primarily on the approach of using Web services that allow us to use online services of “flickr”, even
if the user is not connected to the internet and has a limited range connection. To clarify such a transaction we give a step-by-step explanation of this activity. Initially, we present some general requirements related to the use of “flickr and its API services.

<table>
<thead>
<tr>
<th>Identifier: R2</th>
<th>Name: Cascading Web Services to Bridge Wired and Wireless Networks.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong> Diverse Technologies interconnected. <strong>Requirement details</strong></td>
<td></td>
</tr>
<tr>
<td><strong>R2:1</strong></td>
<td>The software system should be implemented in multiple transport networks. Web Services should interconnect these networks and should allow groups of users to exchange content from one network to the other, without noticing the difference between networks.</td>
</tr>
<tr>
<td><strong>R2:2</strong></td>
<td>Bluetooth connection should be connected with the first server in the range of at most 10 meters far.</td>
</tr>
<tr>
<td><strong>R2:3</strong></td>
<td>Groups of users connected with Bluetooth devices should be supported by a server that is available in the same area. This server is used as an intermediate communication device, which purpose is to serve for cascading Web services to search for different content in a different server, or to publish the generated content on internet.</td>
</tr>
<tr>
<td><strong>R2:4</strong></td>
<td>The user should be able to generate content and publish it immediately on a flickr service.</td>
</tr>
<tr>
<td><strong>R2:5</strong></td>
<td>The content should be published from a Bluetooth network, cascading to a GPRS, Wi-Fi or a wired network until it reaches an internet connection.</td>
</tr>
<tr>
<td><strong>R2:6</strong></td>
<td>The user should be able to retrieve the published content from a Bluetooth device by cascading over multiple services.</td>
</tr>
</tbody>
</table>

Table 3: Requirements for CWS to bridge Wireless and Wired Networks

We start by publishing the content from a location-based network. Our built-in Web services cascade from the point where the content starts to move, until it is published on the internet, in this case on Flickr. To see this activity from the perspective of the generic architecture, we present the blocks used within this activity.

![Figure 5.9: Publishing content to Flickr – blocks 1, 2 & 4.](image)
5.3.1 Flow of Events

1. A picture is taken with a mobile device at a specific place. The picture is taken with embedded EXIF tags\textsuperscript{xxvii}.
2. The geographical position of the picture is already known and placed to the database.
3. A user wants to publish this picture on flickr.
4. The Web Service is requested to send the picture with all automatic generated EXIF data to flickr.
5. Flickr gets the message and responds to the user that the message is uploaded.
6. The picture is placed on the fly with the exact given location in the map.

![Figure 5. 10: Calling the Flickr service for publishing a picture.](image)

5.3.2 Sequence Diagram for Publishing a Picture on Internet [flickr]

![Figure 5. 11: Sequence - publishing a picture in flickr.](image)
5.3.3 Sequence Diagram for Retrieving the Published Picture from flickr

The sequence diagram of Figure 5.12 shows how a picture is published from a Bluetooth-location based network using a mobile device to flickr. In the other hand, the sequence diagram below shows how the published picture can be retrieved from flickr to a mobile device that uses the Bluetooth-location based network.

![Sequence Diagram](image)

Figure 5. 12: Sequence - Retrieving a picture from flickr to Bluetooth device.

5.3.4 Image on map – automatic geo-tagging

To place an image on a map, we use Flickr services and Flickr map. Figure 5.13, shows an image taken with a Bluetooth mobile device. This image is published on Flickr and at the same time it is automatically geo-tagged and placed on the map.

![Image on Map](image)

Figure 5. 13: Publishing image on flickr from a Bluetooth device.
Here’s the map of the place where the image is published from a Bluetooth mobile device via CWS.

![Map of Växjö with Bluetooth device location](image)

**Figure 5.14: Image on flickr map; geo-tagging.**

### 5.4 Advantages of Cascading Web Services based on Scenarios

In section 3.1.1, we have mentioned the standard of Web Services and one of their main advantages: “working independently”. When we look back at these scenarios, we have a clear picture of what are the advantages of incorporating Web services, while there are a couple of other ways to make these activities without WS. Primarily, we succeeded to comprehend a Bluetooth network which for now is one of the most limited networks, especially when it comes to involve it in an environment where wide ranges of networks are present. Nevertheless, usability of WS allows the Bluetooth network to be exposed to the wide Web. To prove this occurrence, we have presented the case #3 scenario that describes how a Web service cascades from a location-based network to the internet, in our case to flickr services. This activity is quite evident for explaining the WS advantage, and to ensure that with cascading WS, users exercise different activities in different networks, for whom otherwise these activities would become impossible, we make a comparison with an existing system that is used for the same activity but without WS.

Following the Case#1-Scenario, based on the AMULETS project of “CeLeKT” Research Center, we introduce another project which is also being developed as an idea of a Master Thesis. The purpose of this project is to develop a couple of innovative activities that are related to the content produced. A part of this Master Thesis is an activity of AMULETS called “AMULETS biology”. In Case#1 Scenario, we have explained that this activity was performed by members of “CeLeKT” Research Center. Therefore, we make a comparison between the software solution we have implemented with the software developed by CeLeKT for the activity of “AMULETS biology”.

In the table below, we present some of the main functionalities of both applications. The purpose is to show which functionalities are implemented in both applications, and at the same time, what is the basic difference. It is clearly shown that both applications
which are capable to perform the same activity have quite a lot of similar functionalities. But the main difference stands in the process of transmitting information. In our case, we integrate cascading Web services by allowing a Bluetooth user to get involved with request-response commands even in a limited environment, while the AMULETS trial, does not provide cascading Web services that are capable to transmit information from a limited area networks to any other network. Another important issue that clearly shows the difference of these two applications is the presence of multiple transport networks incorporated in our application that communicate and are interoperable regardless of diversity.

<table>
<thead>
<tr>
<th>In the Context of Content</th>
<th>CWS</th>
<th>Amulets biology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Creation</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Sending Content</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Publishing Content</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Receiving Content</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>In the Context of Technology</th>
<th>CWS</th>
<th>Amulets biology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Services</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Bluetooth</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GPRS</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Wired Network</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 4: Comparison – CWS & Amulets biology
6. Technical Implementation

In this chapter, we describe how the software system is implemented, by introducing all the developed components that shape the general solution of this software. We present the technology used for development of Web Services, Thin Client Application, and the Desktop Application. In the coming sections, we give an overview of the inner structure of the system. We present the generated classes and methods, and we give an explanation for each of them. The purpose of this explanation is to show the interrelation and the flexibility that Web services offer when they are present in various interoperating networks.

6.1 Software Components - Interoperation

There are several definitions that define the meaning of what Software Components are. In general, we can say that a software component is a predefined service, which can be integrated into other systems. Now, we give a more precise definition, which is also quite well-known, that has been published by Clemens Szyperski:

“A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties.” (Szyperski, 1999)

Related to components interoperability, there is another definition given by Johannes Sametinger:

“Software components are self-contained, clearly identifiable pieces that describe and/or perform specific functions, have clear interfaces, appropriate documentation, and a defined reuse status” (Sametinger, 1997).

In accordance with the definitions stated above, our inner system infrastructure consists of some components that interoperate with one another. Each of them provides some specific features that allow the component to communicate and be flexible, because we have designed components that are reusable in different environments with different programming languages. In our case these components are “Web services” that have a standardized interoperation mechanism, which allows us to have and to create a very generic architecture with multiple transport networks, by dividing these components in blocks of these architecture. This software component composition in the architecture, even if it is divided in distinctive blocks, has the ability of communication and interoperation.

In Figure 6.1, we present our generated components. We show how they are related with each other, which component depends on the functions of the other component, and at which application they belong. The software system is implemented on two servers of IIS. These servers host our generated Web services, through which the application connect to perform the data transactions. The mobile application and the Web-based application connect with both servers, including the procedure of cascading between Web services. For details related to the components of Web services, section 6.4.2 and section 6.4.3 provide information related to their functionalities.
6.2 Development Tools Overview

To improve communication between networks and devices, Web services are used to affect interoperability by referring to the generic architecture design of SOA as presented in Figure 4.1. In this project, Web services are defined as component-based parts of the system (Assmann, 2003). They are specifically developed to perform the action of cascading. We introduce Web services that have been created with the C# programming language, by giving detailed information related to the problem we investigate, so we can fully accomplish the defined problem and the solution that we provide as stated in section 1.1. De facto, the purpose of this description is to give an overview for the development process of each Web service. Furthermore, we give a brief description of the method that binds these Web services. We test the interoperability between networks and Web services build on multiple programming languages. Realizing this purpose, it means that Web services go beyond limits and beyond borders of distinctive networks. The C# programming language enables us to build this application of multi services, by providing tools that fully exploit computing and communication. MS SQL Server is the database system that is build independently for each existing web server. Each database is placed together with a Web service in one server. This supports the data flow through cascading process, to be performed in the system.
6.2.1 .Net Web Services

C# is a very suitable object-oriented programming language, especially for implementing some parts of the software solution. It enables us to build a highly portable application which facilitates the exchange of information and services over the Web. This language provides support for software engineering principles including strong type checking, array bounds checking, detection of attempts to use uninitialized variables, and automatic garbage collection. It also has a big importance for software robustness, durability, and programmer productivity. With C# programming language we intend to develop the software as a component, which will be suitable for deploying it in a distributed environment. Portability is an issue which is specifically taken in consideration by this language. Source code and programmers portability is an important aspect, which helps us to easily get integrated with it. C# is also a very suitable language for writing applications for embedded and hosted systems. Web services are developed using the .NET platform, therefore now we will try to give a small overview for .Net Web services and for some general tools that are needed for developing Web services in this platform. The .NET Framework is Microsoft's managed code programming model and runtime for building applications on the Windows platform. For developing .Net Web services we need Visual Studio, IIS and the .NET Framework by which we can design, consume and expose Web services. An important feature of .NET platform is the "Add Web Reference" dialog box, which automatically generates proxy code for WSDL defined Web services. In fact this enables consumption of WS from any platform.

6.3 Research Design – Artifact

The relevance of the problem and the research for this solution is demonstrated structurally and the solution must be effective based on requirements. An effective solution may not be presented as the “optimal” solution (Bell et al., 2006). But the background problem and the idea presented for this thesis may be enough descriptive according to the research for this system and according to requirements with the solution we provided. The software system is developed and can be used in any network environment that is comparable with the architecture design of this project as described in chapter 4. De facto, this artifact is a tangible byproduct, created during the development phase. For this artifact, a detailed description is presented below that describes each part of the software by providing a class diagram that matches the inner structure of the software system. With this developed software system, the innovative idea of this thesis is accomplished by implementing structure of the generic architecture.

6.4 Interaction – General Class Diagram

Figure 6.2, displays the main implemented classes of the software. The purpose of this class diagram is to explain and define the inner structure of the software. This figure presents general classes and some methods that are developed.
6.4.1 Brief Description of Design Classes - Thin Client

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Responsibilities</th>
<th>Attributes</th>
</tr>
</thead>
</table>
| Server             | This part is responsible for connecting the client and the server. It is also responsible for connecting the database. | 1. Responsible for making a connection between the client and the server.  
2. Responsible for synchronization and login, logout functionalities. | Data about users and all the content available. |
| frmAuthentication  | This part is responsible for authentication of the client which is done by Web service. | 1. Authenticate Clients.  
2. Anonymous and personal search. | Data about users and all the content available. |
| frmMain            | This is the main component and it is used to connect the other components together.  
This component comprises some of the main functions and methods of the software. | 1. Navigate to activities.  
2. Navigate to lookup.  
3. Navigate to inbox.  
4. Navigate to Pictures.  
5. Navigate to Flickr, etc. | Data about users and all the content available. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Responsibilities</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerService</td>
<td>This part is responsible for connecting the client and the server. It is also responsible for connecting the database.</td>
<td>1. Making a connection between:</td>
<td>List of all connected clients.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a. Client with server.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. Server with server.</td>
<td></td>
</tr>
<tr>
<td>ServiceGet</td>
<td>This part is responsible for returning all kinds of data.</td>
<td>1. Return data.</td>
<td>Data about users and all</td>
</tr>
</tbody>
</table>

6.4.2 Web Service [location based]

The table below has a brief description of design classes – Web Service [location based]
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Responsibilities</th>
<th>Attributes</th>
</tr>
</thead>
</table>
| ServerService | This part is responsible for connecting the client and the server. It is also responsible for connecting the database. | 1. Making a connection between:  
   a. Client with server.  
   b. Server with server. | List of all connected clients.                                                   |
| ServiceCascade| This service is responsible for returning all kinds of data.                 | 1. Return data’s                                                                 | Data the content available in server.   |
| FlickrService | This Web service is responsible for communicating with Flickr services.      | 1. Publish Picture  
   2. Receive Pictures  
   3. Search To Flickr |                                                                         |

Table 6: Design Classes – Web Service [location-based]

6.4.3 Web Service [Cascade]

This table has a brief description of Design Classes – Cascading Web Service.
Information | This service is responsible for putting in action the function of lookup. | 1. Lookup in DB. |
---|---|---|
Service Factory | This service is the main part and it is responsible for relationship between Web services | Responsible for connecting classes. |

Table 7: Design Classes – Cascading Web Services

6.4.4 Class Analysis [Web-based Application]

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Responsibilities</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>This part is responsible for connecting the client and the server. It is also responsible for connecting the database.</td>
<td>1. Responsible for making a connection between the client and the server. 2. Responsible for synchronization and login, logout functionalities.</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>This part is responsible for authentication of the client which is done by Web service.</td>
<td>1. Authenticate Clients.</td>
<td>Data about users and all the content available.</td>
</tr>
<tr>
<td>Picture</td>
<td>This part is responsible for sending and receiving Pictures.</td>
<td>Sends different types of images.</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6. 5: Class Analysis [Cascading Web Services].
| Inbox | Here we can receive different activities and different pictures from different places where the content is generated. | 1. Receive activities.  
2. Receive images. |  |
|---|---|---|---|
| Flickr | This part is responsible to communicate with Flickr services. | 1. Publish Picture on Map.  
2. Receive Pictures.  
3. Search to Flickr. | Content generated can be published in Flickr services. |

Table 8: Design Classes – Web-based Application
7. Conclusions and Future Challenges

The main topic of this thesis focuses around one fundamental principle extracted from Web services paradigm. Thus, in this chapter we discuss the application, concerning the implementation of cascading Web services and the results that have been attained. Moreover, we give a brief description of different aspects of the research, based on cascading Web services, and we present the implication for the future work of CWS.

7.1 Results

This summary, describes the results of the project and the set of application we have developed. The purpose of the work conducted in this thesis was to document the deliberate research for the operation of cascading Web services based on theoretical interpretation. This study was conducted primarily to understand cascading Web services performance, their flexibility and their interoperability in diverse networks. With Web services we have presented a relatively new growing technology of communication, and we have supplemented this technology by expanding its repertoire with the method of how multiple services cascade. Going back to the research question of this thesis, the goal was to bring forward a new technology of communication:

“How will it be possible to conceptualize, design and implement a system architecture based on Cascading Web Services that can provide services independent from the user’s location and devices?”

We have provided new series of applications that are capable to allow multiple users to interact in system architecture of multiple diverse networks. The system signifies a flexible solution for the user, strongly based on cascading Web services that are the key feature for involving these multiple diverse networks to be interoperable. As for the application, we were able to program a couple of Web services that operate in different networks. The primary network from where CWS have been moving is the Bluetooth – location based network. These applications are developed in the context of “MeMiMo” project of the “CeLeKT” Research Center. Consequently, the project comprises two different applications. One application is a mobile application, developed especially for hand-held mobile devices (PDA). Whereas the second application is a Web-based application developed for desktop computers. The reason for building two different applications is to show how two or more groups of users interact with the system of multiple Web services and networks. The purpose was to show a couple of results:

1. Cascading Web Services from all platform networks as presented in Chapter 3.
2. System – user interaction. Transmitting different content.
3. Publishing and retrieving content from a Bluetooth mobile device to on-line services (flickr).
4. Generating different content, registering it in the database, making it available and searchable.

NOTE: For more details concerning the Thin Client and the Web-based application, see Appendix A. For details related to coding see Appendix B, C and D, and for details related to the database see Appendix E.
7.1.1 Reflections

Different approaches that have been discussed in section 2.3 have influenced our thesis, becoming comparable to issues that embody complex features. These cutting edge technologies assisted us for the conceptual design of building and implementing a system of Cascading Web services. This study has reflected our thesis in becoming more valuable by being compared to world wide known technologies. Our system can potentially be perceived as an innovative approach of today’s high-tech evolutionary systems in the context of learning technologies. The usability aspects inherited in these technologies, in one angle are comprehended in our thesis, especially when the software solution would be used for learning domains that might impact the role of the user in m-learning activities. Reflection of this thesis can in this context be explained according to the approach of heterogeneous networks that embrace Cascading Web services. Further research is needed in order to explore on-going projects similar to our thesis, making a comparison for benefits and drawbacks of our approach with others.

A number of on-going projects - applications that have been developed recently share a similar theme to our work that are supporting mobile learning activities for indoor and outdoor settings - including BWL (Chen et al., 2003), mPower system (Lee et al., 2007) and ENLACE system (Celorio and Verdejo, 2007). These projects focus on agent, multi-agent systems for context-aware services provided by mobile environments. The BWL system offers a mobile learning system which supports the students to learn through scaffolding by using mobile devices to perform bird-watching activities in outdoor settings using a Wi-Fi based network. The ENLANCE project is developed to exploring new design and implementations of a technological infrastructure, constructed especially for an educational platform, offering pervasive services for indoor and outdoor activities. Their infrastructure provides interoperability and ubiquitous delivery of information among devices that permits the data to be shared and received in different ways. The mPower is a tool that uses asynchronous message exchange among multiple agents that have the role for two main components: initiator-interaction and responder - reacting to initiator.

These applications are used to perform activities in the field of learning activities reflecting our thesis. The difference that is transparent at this matter is the unique combinatorial architecture of our system that incorporated heterogeneous networks to interact through multiple Web services by sequentially transferring data. While in the above mentioned systems, they present an architecture of limiting their communication to GPRS only or Wi-Fi only. Although our system has reached a standard for being useful and flexible to users needs, we could have put more effort on these issues:

- **Database design.**
- **Optimization** – achieving the most efficient design for the software system and improving system interaction between users.
- **Efficiency** – avoiding late replies from Web services. Customizing Web services to create a special intelligent feature that offers a faster manner of bringing the result to the user, and improving the speed of data transfer.

7.2 Future Challenges

In a project of multi services and various transport platforms as this one, there are numerous areas where different techniques can be implemented as well as improved. An important issue that could be considered in the future work is to customize the application by involving Cascading Web Services in a Mobile Web Server to Bluetooth, Wi-Fi, GPRS or Wired Networks.
In section 2.1.5 we discussed the new developed technology of Nokia. A future challenge involving Nokia (Apache Raccoon) would be to involve Web Services is integrating a connection between the Mobile Web Server in a GPRS network and other web servers of different networks. As an example, consider the following: the exchange of content - from Nokia services from a user connected to a Nokia mobile web server to a user connected in a Bluetooth location-based network. This particular example is described in the picture below.

![Figure 7.1: Bluetooth – GPRS networks; Nokia Mobile Server.](image)

Another important matter at the stage of future challenge is described in Figure 7.2. The figure presents a typical Bluetooth-location based network that connects devices in the master-slave manner by crossing the limits of the Bluetooth connection, which for now is until 10m. The figure explains how a Bluetooth network can be organized in such a way that multiple devices in different locations might be connected to the server even from a distance longer than 10m. We propose a solution for this: to use the Nokia mobile web server for storing Web Services that could potentially cascade to other Bluetooth devices that are located out of the Bluetooth radio frequency spectrum.

![Figure 7.2: Future Challenge; Location – based network.](image)
7.3 Conclusions

With the focus on Mobile Technology of this thesis, we have achieved the goal of developing the technical applications that are adaptable for Mobile Devices. The system in general consists of two applications as mentioned in section 1.2: a mobile based application and a web-based application. Mobile Web Services and the ‘cascading’ procedure have been successfully implemented based on the research we did and the implementation for testing the research results. The criteria of the goals have also been achieved by accessing cascading Web services; allowing users to interact with the system based on the data flow; exchanging different content through different networks and devices; and geo-tagging.

Sustaining the overall evaluation of the results brought from this thesis, interoperability of networks is obviously very necessary for users to have flexibility when they use such a system. Bridging wireless and wired networks, and letting the user to work in a distributed system of multiple diverse networks, is a successful achievement of this thesis. According to the problem definition and the goals of this thesis, we have implemented a technical solution based on the generic architecture which shows multiple transport platforms that communicate with one-another via cascading Web services. CWS are built upon dynamic algorithms that allow multiple users that operate through multiple mobile devices to interact and communicate through different networks.

From different perspectives, the reader’s perception can lead to discovering some issues which have been studied and achieved in this thesis:

- First, it covers a deep study of Web services nature by introducing the technique of cascading Web services.
- Second, we have designed a distributed architecture which is build upon different networks where the general architecture including the inner architecture has been taken into consideration.
- Third, we have also stressed some small issues regarding the generated content and data types of the content. To conclude our work, in Chapter 5 we presented some real case scenarios by combining the architecture description, with cascading Web services. We showed how real cases can work using our system; what kinds of methods need to be developed to allow such operations; and how users interact with the rest of the network, beginning from a Bluetooth device.
Glossary

COM - Component Object Model
CORBA – Common Object Request Broker Architecture
CWS – Cascading Web Services
DCOM – Distributed Component Object Model
EJB – Enterprise Java Beans
GPRS – General Packet Radio Service
GPS – Global Positioning System
HTTP – Hypertext Transfer Protocol
ICT – Information and Communication Technology
ORB – Object Request Broker
RSS – Rich Site Summary
SOA – Service Oriented Architecture
SOAP – Simple Object Access Protocol
SQL – Structured Query Language
TCP/IP – Transmission Control Protocol / Internet Protocol
UDDI – Universal Description, Discovery and Integration
UMTS – Universal Mobile Telecommunications System
WAP – Wireless Application Protocol
Wi-Fi – Wireless Fidelity
WSDL – Web Services Description Language
WS – Web Services
XML – eXtensible Markup Language
References


Auletta V., De Cristofaro E., Blundo C., Raimato G. (2006). Performance Evaluation of Web Services over Bluetooth, University of Salerno, Baronissi, Italy


Internet sites:

http://www.flickr.com/services/api/, viewed 2007-06-01
http://code.google.com/, viewed 2007-06-01
http://w3.msi.vxu.se/~wlo/files/WSWT06/Slides0.pdf, viewed 2007-06-01
http://www.w3.org/2002/ws/, viewed 2007-06-01
http://www.w3.org/2002/ws/, viewed 2007-06-01
http://www.microsoft.com/net/basics.mspx, viewed 2007-06-01
Appendices

APPENDIX A – Demo [Thin Client and Web-based Application]

Appendix A, gives a brief demonstration of a mobile and a web-based application that we have developed in the framework of this thesis. A number of screenshots are presented to give the reader an easier way of understanding how these applications work and cooperate.

Demo – Thin Client

In the figure above we present the main screens of the Mobile Application. To be able to use the system, the user (in our case a group of users) should login in order to perform an activity.

Figure A. 1: Login and Main screenshots.

Figure A. 2: Messages Received.
Figure A.2 shows how the received content (activity or image) is displayed to the user.

In figure A.3, a group of users can generate a new activity or can upload a picture that can be sent to another group of users or to all groups of users that are available while the activity performs. These screenshots show a demonstration of one activity that could be performed by two different groups; one using the Mobile Application connected with a Bluetooth location-based network; and the other group using the Web-based application, connected with Wi-Fi, GPRS or a wired network. For details related to this activity, refer to section 5.1 Case#1 – Scenario.

Figure A.4 shows the activity or the picture received by the user. These parts of the Mobile application are used to view details of the content received.
Figure A.5, shows a specific part of the Mobile application, where the user is able to upload, publish (geo-tag) and retrieve the content using Flickr Services.

**Demo – Web-based Application**

Functionalities of the Web-based application are as same as functionalities of the Mobile Application; hence we don’t give details for each screenshot.
Figure A. 7: Search feature.

Figure A. 8: Creating an activity
Figure A. 9: Sending a picture

Figure A. 10: Viewing details of an activity
Figure A. 11: Publishing a picture in flickr

Figure A. 12: Retrieving a picture from flickr
APPENDIX B - Web services in a location-based environment

The codes below are extracted from Web Services, Thin Client application and from the Web-based application. Note that these codes are only sample codes of the software system in general, they show the main functionalities, and therefore, the whole developed code is not included here.

ServiceActions

```csharp
/*
 * This code presents a small example of Cascading Web Service
 * which trigger information from a local Web service, and from
 * another one in another server. This service returns multiple
 * results from different Web services.
 */
[WebMethod]
public string ActionCascadeFirst(string data)
{
    string printresult_1 = "";
    string printresult_2 = "";
    string printresult_3 = "No answer";
    Cascade.ServiceCascade cascade = new Cascade.ServiceCascade();

    printresult_1 = _lookupLocal(data);
    printresult_2 = cascade._LookUp_Info(data);
    try
    {
        if (_lookupLocal(data) == printresult_1 &&
            cascade._LookUp_Info(data) == printresult_2)
        {
            return _lookupLocal(data).ToString() + "" +
            cascade._LookUp_Info(data).ToString();
        }
        else
        {
            return printresult_3;
        }
    }
    catch (System.Exception ex)
    {
        return ex.Message + " :: " + ex.Source;
    }
}

/*
 * Service for looking up in local database.
 */
public String _lookupLocal(String name)
{
    connection.Open();
    string query = "SELECT * FROM M_Datas WHERE D_name like " +
        name + "]%";
    SqlCommand oCmd = new SqlCommand(query, connection);
    SqlDataReader rdr = oCmd.ExecuteReader();

    string a = "";
    string b = "";
    while (rdr.Read())
    {
```
a += rdr["d_name"].ToString();
a += ", ";
b += rdr["d_description"].ToString();
b += ", ";
}
rdr.Close();
connection.Close();
a = a + "" + b;

return a;
}*

/*
 * Wrapper for Authentication of Flickr services.
 */

[WebMethod]
public string AuthenticateFlickr()
{
    CascadeToFlickrService.FlickrService cascadeToFlickr = new
    CascadeToFlickrService.FlickrService();
    OutputTextSuccessMessage = cascadeToFlickr.AuthenticateFlickr(ApiKey, SharedSecret, AuthToken);
    return OutputTextSuccessMessage;
}

ServiceGet/*
 * Web service for getting a picture from a database.
 */

[WebMethod]
public byte[] ReturnPicture(string Id)
{
    string SQL;
    byte[] BA;
    SqlCommand SC = new SqlCommand();
    SQL = "SELECT Picture FROM M_Img WHERE id = " + Id + ";"
    SC.Connection = connection;
    SC.Connection.Open();
    SC.CommandType = System.Data.CommandType.Text;
    SC.CommandText = SQL;
    BA = ((byte[])(SC.ExecuteScalar()));
    SC.Connection.Close();
    SC.Dispose();
    return BA;
}

/*
 * Service that returns data from database as DataSet.
 */

[WebMethod]
public DataSet ReturnDataActivity(string GroupName)
{
    connection.Open();
    SqlDataAdapter SqlDa = new SqlDataAdapter("Select id, FromGroup, Activity, Temperature, Humidity, Comment from

58
M_ActivitiesSep where Status = '" + 1 + '" and ToGroup = '" + GroupName + '"', connection);
DataSet Ds = new DataSet();
SqlDa.Fill(Ds);
connection.Close();
return Ds;
}

/*
* Service for Authentication of the groups.
*/
[WebMethod]
public bool ReturnAuthentication(string Groups, string Password)
{
    connection.Open();
    string selStr = "Select Groups, Pswd from M_Groups where Groups = '" + Groups + '" and Pswd = '" + Password + '"";
    SqlDataAdapter dAdapter = new SqlDataAdapter(selStr, connection);
    DataSet dSet = new DataSet();
    dAdapter.Fill(dSet, "M_Groups");
    if (Convert.ToInt32(dSet.Tables["M_Groups"].Rows.Count.ToString()) > 0)
    {
        connection.Close();
        return true;
    }
    else
    {
        connection.Close();
        return false;
    }
}

ServiceSet

/*
* Web service for inserting image in a database.
*/
[WebMethod]
public string SetPicture(byte[] ImgIn, string Comment, string FromGroup, string ToGroup, string groupId)
{
    try
    {
        string StatusAll = "1";
        connection.Open();
        SqlCommand cmd = new SqlCommand("insert into M_Image " + 
"(Picture,Comment, DateTime, FromGroup,ToGroup, StatusAll, GroupId) values @{pic, @Com, @DT, @FromGroup, @ToGroup, @StatusAll, @GroupId}", connection);
        cmd.Parameters.Add("@pic", ImgIn);
        cmd.Parameters.Add("@Com", Comment);
        cmd.Parameters.Add("@DT", DateTime.Now);
        cmd.Parameters.Add("@FromGroup", FromGroup);
        cmd.Parameters.Add("@ToGroup", ToGroup);
        cmd.Parameters.Add("@StatusAll", StatusAll);
        cmd.Parameters.Add("@groupId", groupId);
        cmd.ExecuteNonQuery();
    }
    catch (Exception e)
    {
        throw new ArgumentException(e.Message);
    }
    finally
    {
        connection.Close();
    }
    return "Success";
}
finally
{
    connection.Close();
}
return "Picture is send to " + ToGroup;

/*
 * Service for inserting Activity by a Group.
 */
[WebMethod]
public string SetActivityByGroup(string FromGroup, string ToGroup, string Activity, string temperature, string humidity, string comment, string groupId)
{
    try
    {
        connection.Open();
        SqlCommand cmd = new SqlCommand("INSERT INTO M_ActivitiesSEP (FromGroup,ToGroup, Activity,Temperature, Humidity, Comment, DateTime, Status,GroupId) VALUES ('" + FromGroup + "," + ToGroup + "," + Activity + "," + temperature + "," + humidity + "," + comment + "," + DateTime.Now + ",", 1 + "," + groupId + ")", connection);
        cmd.ExecuteNonQuery();
    }
    finally
    {
        connection.Close();
    }
    return "your activity is sent " + ToGroup.ToString();
}

ServiceSet
The following operations are supported. For a formal definition, please review the Service Description.

- SetActivityByGroup
- SetActivityStatus
- SetActivityToAll
- SetImageStatus
- SetPicture
- SetPictureToAll
- SetPictureToPublish
- SetPublishedPicture
ServiceActions

The following operations are supported. For a formal definition, please review the Service Description.

- ActionCascadeFirst
- ActionCascadeLocalFirst
- AuthenticateFlickr
- GetPictureFromFlickr
- PublishPictureToFlickr
The following operations are supported. For a formal definition, please review the Service Description.

- ReturnTempPicture
- ReturnActivitiesStatus
- ReturnAuthentication
- ReturnCommentFromRetrieve
- ReturnDataActivity
- ReturnDataActivityAction
- ReturnDataCombo
- ReturnDataImage
- ReturnDataImageComment
- ReturnDataOldActivity
- ReturnDateOldImage
- ReturnDataPictureFromRetrieve
- ReturnDateTimeFromActivities
- ReturnDateTimeFromImage
- ReturnDateTimeFromRetrieve
- ReturnImageStatus
- ReturnMaxIdFromMPublish
- ReturnMaxOfActivity
- ReturnMaxOfImages
- ReturnPicture
- ReturnTempPictureFromMPublish
APPENDIX C – Web services
The code below shows how Web services are developed that perform the function of cascading. It also presents the developed code using flickr API for publishing a picture on flickr’s online site, geo-tagging the picture, and retrieving the published picture. All these actions are performed from a Bluetooth network, as well as they can be performed with other networks.

ServiceCascade

```csharp
/*
* Service for looking up information requested by another Web
* service from a location-based environment.
*/

[WebMethod]
public string _LookUp_Info(string nameIn)
{
    String serviceInfo = "Information";
    Service_Factory ServiceFactory = new Service_Factory();
    Information i = (Information)ServiceFactory.getOne(serviceInfo);
    Information info = new Information();
    string printresult = "";
    string iString = i.lookup(nameIn);
    printresult += iString;
    Information.con.Close();
    return printresult;
}
```

FlickrService

```csharp
private string Frob;
private string OutputTextFrob;
private string OutputTextURL;
private string OutputTextSuccessMessage;

[WebMethod]
public string AuthenticateFlickr(string ApiKey, string SharedSecret, string AuthToken)
{
    Flickr flickr = new Flickr(ApiKey, SharedSecret);
    Frob = flickr.AuthGetFrob();
    OutputTextFrob += "Frob = " + Frob + "\r\n";
    string url = flickr.AuthCalcUrl(Frob, AuthLevel.Write);
    OutputTextURL += "Url = " + url + "\r\n";
    return OutputTextFrob + " " + OutputTextURL;
}
/*
* This Web service is created for publishing a picture to
* flickr.
* */

[WebMethod]
public string PublishPictureToFlickr(string Filename, string description, string ApiKey, string SharedSecret, string AuthToken)
{
    Flickr flickr = new Flickr(ApiKey, SharedSecret, AuthToken);
```
bool uploadAsPublic = true;
string title = "Master Thesis 'CWS'";
string photoId = flickr.UploadPicture(Filename, title, description, "", uploadAsPublic, false, false);
flickr.PhotosGeoSetLocation(photoId, 56.87067113864678, 14.79072175412314);
OutputTextSuccessMessage = "Image is Published Successfully on Flickr";

return OutputTextSuccessMessage;

private string Title;
private string PhotoId;
private string URL;

/*
 *This Web service is created for retrieving a picture from
 *flickr services.
 */
[WebMethod]
public string GetPictureFromFlickr(string ApiKey, string SharedSecret, string AuthToken, string Tags)
{
    try
    {
        Flickr flickr = new Flickr(ApiKey, SharedSecret, AuthToken);
        Auth auth = flickr.AuthCheckToken(AuthToken);
        PhotoSearchOptions options = new PhotoSearchOptions(auth.User.UserId);
        options.SortOrder = PhotoSearchSortOrder.DatePostedDesc;
        options.Text = Tags;
        options.PerPage = 100;
        Photos photos = flickr.PhotosSearch(options);
        Flickr.FlushCache(flickr.LastRequest);
        Photo photo = photos.PhotoCollection[0];
        Title = photo.Title;
        PhotoId = photo.PhotoId;
        URL = photo.SmallUrl.ToString();
        return URL;
    }
    catch (Exception)
    {
        return URL;
    }
}
ServiceCascade

The following operations are supported. For a formal definition, please review the Service Description.

- CascadeData
- LookUp_Info

FlickrService

The following operations are supported. For a formal definition, please review the Service Description.

- AuthenticateFlickr
- GetPictureFromFlickr
- PublishPictureToFlickr
- SetGeotagging
APPENDIX D – Thin Client

/*
 * Loading Data into the ListView; private DataSet from the
 * Web service that returns info as DataSet.
 * This function is for loading data for Activities.
 */
private void LoadListActivity()
{
    WSGet.ServiceGet getService = new WSGet.ServiceGet();
    DataSet ds = new DataSet("M_ActivitiesSEP");
    ds = getService.ReturnDataActivity(Server.gUserName);
    ListViewItem lit;
    foreach (DataRow dr in ds.Tables[0].Rows)
    {
        lit = new ListViewItem(dr["id"].ToString());
        lit.SubItems.Add(dr["FromGroup"].ToString());
        lit.SubItems.Add(dr["Activity"].ToString());
        lit.SubItems.Add(dr["Temperature"].ToString());
        lit.SubItems.Add(dr["Humidity"].ToString());
        lit.SubItems.Add(dr["Comment"].ToString());
        lvActivities.Items.Add(lit);
    }
}

/*
 * Loading Data into the ListView; private DataSet from Web
 * service that returns info as DataSet.
 * This function is for loading data (images).
 */
private void LoadListImage()
{
    WSGet.ServiceGet getService = new WSGet.ServiceGet();
    DataSet ds = new DataSet("M_Image");
    ds = getService.ReturnDataImage(Server.gUserName);
    ListViewItem lit;
    lvPictures.Items.Clear();
    foreach (DataRow dr in ds.Tables[0].Rows)
    {
        lit = new ListViewItem(dr["id"].ToString());
        lit.SubItems.Add(dr["FromGroup"].ToString());
        lit.SubItems.Add(dr["Comment"].ToString());
        lvPictures.Items.Add(lit);
    }
}

/*
 * The code for getting a picture from a database
 * through Web service.
 */
public void SetPicBox(byte[] ImageArray)
{
    try
    {
        int ArraySize = ImageArray.GetUpperBound(0);
        fs.Write(ImageArray, 0, ArraySize + 1);
    }
}
fs.Close();
inkPicture.SizeMode = PictureBoxSizeMode.StretchImage;
inkPicture.Image = new Bitmap("tmp.jpg");
throw new OutOfMemoryException();
}
catch (OutOfMemoryException ex)
{
    Console.WriteLine("{0} Caught exception #1.", ex);
}
finally
{
    Console.WriteLine("Executing finally block.");
}

} /*
 * Function that Publishes a picture to Flickr Services by
 * Cascading Web Services
*/
private void btnPublish_Click(object sender, EventArgs e)
{
    lblMessage.Text = "";
    if ((txtDescription.Text) == ")
    {
        MessageBox.Show("please write Description...");
    }
    else
    {
        FileStream fs = new FileStream(Server.copy1.ToString(), FileMode.Open);
        int photoLength = (int)fs.Length;
        byte[] photoBytes = new byte[photoLength];
        int n = fs.Read(photoBytes, 0, photoLength);
        WSSet.ServiceSet setService = new WSSet.ServiceSet();
        setService.SetPictureToPublish(photoBytes, txtDescription.Text, Server.gUserName, "56.87067113864678", "14.79072175412314");
        WSGet.ServiceGet getService = new WSGet.ServiceGet();
        getService.ReturnTempPictureFromMPublish(getService.ReturnMaxIdFromMPublish());
        WSActions.ServiceActions serviceActions = new WSActions.ServiceActions();
        lblMessage.Text = "Your " + serviceActions.PublishPictureToFlickr("C:\Inetpub\wwwroot\MeMiMo_WS\temp.jpeg", txtDescription.Text);
    }
} /*
 * Function that Retrieves a picture from Flickr Services by
 * Cascading Web Services.
 */
private void btnRetrive_Click(object sender, EventArgs e)
{
    try
```csharp
{ 
    WSActions.ServiceActions serviceActions = new MeMiMo_Mobile.WSActions.ServiceActions();
    Uri url = new Uri(serviceActions.GetPictureFromFlickr(txtText.Text));
    WebBrowser1.Navigate(url);
} 
catch (Exception)
{
}
```

```csharp
/*
* Function code for browsing a picture
*/
private void btnOpen_Click(object sender, EventArgs e)
{
    OpenFileDialog fileDlg = new OpenFileDialog();
    fileDlg.Filter = "Image files (*.bmp; *.gif; *.jpg; *.png)|*.bmp;*.gif;*.jpg;*.png|" + "All files (*.*)|*.*";
    if (fileDlg.ShowDialog() == DialogResult.OK)
    {
        try
        {
            System.Drawing.Bitmap immagine = new Bitmap(fileDlg.FileName);
            Server.copy = fileDlg.FileName.ToString();
            inkPicture.Image = new Bitmap(fileDlg.FileName);
            inkPicture.SizeMode = PictureBoxSizeMode.StretchImage;
        }
        catch (Exception)
        {
            MessageBox.Show("Unable to load the image file!", "Load Image Failed");
        }
    }
}
```

```csharp
/*
* Function returning the authentication for a specific group
*/
WSGet.ServiceGet serviceGet = new WSGet.ServiceGet();
if (serviceGet.ReturnAuthentication(cmbGroups.Text, txtPass.Text) == true)
{
    Server.Group = cmbGroups.Text;
    Server.status = cmbGroups.SelectedValue.ToString();
    frmMain frm = new frmMain();
    frm.Show();
}
else
    MessageBox.Show("Please provide your correct Group or Code!!");
```
APPENDIX E – Database

```sql
IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_ActivitiesSEP]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_ActivitiesSEP]
GO

IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_Datas]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_Datas]
GO

IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_Groups]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_Groups]
GO

IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_Image]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_Image]
GO

IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_Publish]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_Publish]
GO

IF EXISTS (SELECT * FROM sysobjects WHERE id = object_id(N'[dbo].[M_Retrieve]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [dbo].[M_Retrieve]
GO

CREATE TABLE [dbo].[M_ActivitiesSEP] (  
    [id] [int] IDENTITY (1, 1) NOT NULL,  
    [FromGroup] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [ToGroup] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [Activity] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [Temperature] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [Humidity] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [Comment] [nvarchar] (200) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [DateTime] [datetime] NULL,  
    [Status] [int] NULL,  
    [GroupId] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ) ON [PRIMARY]
GO

CREATE TABLE [dbo].[M_Datas] (  
    [id] [int] IDENTITY (1, 1) NOT NULL,  
    [d_name] [nvarchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL,  
    [d_description] [nvarchar] (150) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ) ON [PRIMARY]
GO
```
CREATE TABLE [dbo].[M_Groups] (  
[Id] [int] IDENTITY (1, 1) NOT NULL ,  
[Groups] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Status] [int] NULL ,  
[Pswd] [nvarchar] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ) ON [PRIMARY]  
GO  

CREATE TABLE [dbo].[M_Image] (  
[id] [int] IDENTITY (1, 1) NOT NULL ,  
[Picture] [image] NULL ,  
[Comment] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[DateTime] [datetime] NULL ,  
[FromGroup] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[ToGroup] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[ToAllGroups] [char] (10) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[StatusAll] [int] NULL ,  
[GroupId] [int] NULL ) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]  
GO  

CREATE TABLE [dbo].[M_Publish] (  
[Id] [int] IDENTITY (1, 1) NOT NULL ,  
[FromGroup] [nvarchar] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[Picture] [image] NULL ,  
[Description] [nvarchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[dateTime] [datetime] NULL ,  
[longitude] [char] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,  
[latitude] [char] (50) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]  
GO  

CREATE TABLE [dbo].[M_Retrieve] (  
[id] [int] IDENTITY (1, 1) NOT NULL ,  
[Picture] [image] NULL ,  
[DateTime] [datetime] NULL ,  
[Comment] [nvarchar] (100) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]  
GO
<table>
<thead>
<tr>
<th>Endnote</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Service Oriented Architecture (SOA) is a software architecture that defines a relationship between two or more services.</td>
</tr>
<tr>
<td>2</td>
<td>Users can access data while they are in motion.</td>
</tr>
<tr>
<td>3</td>
<td>A self-configuring mobile network connected with wireless links that can operate in a standalone fashion and can be connected to larger networks.</td>
</tr>
<tr>
<td>4</td>
<td>Wrapper is a package that covers methods of a Web Service.</td>
</tr>
<tr>
<td>5</td>
<td>Information and communications technology.</td>
</tr>
<tr>
<td>7</td>
<td><a href="http://w3.msi.vxu.se/~wlo/files/WSWT06/Slides0.pdf">http://w3.msi.vxu.se/~wlo/files/WSWT06/Slides0.pdf</a></td>
</tr>
<tr>
<td>8</td>
<td><a href="http://www.w3.org/2002/ws/">http://www.w3.org/2002/ws/</a></td>
</tr>
<tr>
<td>9</td>
<td><a href="http://www.iso.org/iso/en/ISOOnline.frontpage">http://www.iso.org/iso/en/ISOOnline.frontpage</a></td>
</tr>
<tr>
<td>10</td>
<td>Electro-magnetic waves carry information through atmospheric space.</td>
</tr>
<tr>
<td>13</td>
<td><a href="http://www.omg.org/gettingstarted/corbafaq.htm">http://www.omg.org/gettingstarted/corbafaq.htm</a></td>
</tr>
<tr>
<td>14</td>
<td><a href="http://www.microsoft.com/com/default.mspx">http://www.microsoft.com/com/default.mspx</a></td>
</tr>
<tr>
<td>15</td>
<td><a href="http://java.sun.com/products/ejb/">http://java.sun.com/products/ejb/</a></td>
</tr>
<tr>
<td>16</td>
<td><a href="http://www.w3.org/2002/ws/">http://www.w3.org/2002/ws/</a></td>
</tr>
<tr>
<td>17</td>
<td><a href="http://www.microsoft.com/net/basics.mspx">http://www.microsoft.com/net/basics.mspx</a></td>
</tr>
<tr>
<td>18</td>
<td><a href="http://www.macdevcenter.com/pub/a/mac/2005/08/02/flickr.html">http://www.macdevcenter.com/pub/a/mac/2005/08/02/flickr.html</a></td>
</tr>
<tr>
<td>19</td>
<td><a href="http://www.flickr.com/services/api/">http://www.flickr.com/services/api/</a></td>
</tr>
<tr>
<td>21</td>
<td><a href="http://code.google.com/">http://code.google.com/</a></td>
</tr>
<tr>
<td>22</td>
<td><a href="http://research.nokia.com/research/projects/mobile-Web-server/">http://research.nokia.com/research/projects/mobile-Web-server/</a></td>
</tr>
<tr>
<td>23</td>
<td>Slave is the second unit (i.e. mobile device) controlled by a master (server).</td>
</tr>
<tr>
<td>24</td>
<td>Personal Area Network.</td>
</tr>
<tr>
<td>25</td>
<td><a href="http://www.usbyte.com/common/binary_system.htm">http://www.usbyte.com/common/binary_system.htm</a></td>
</tr>
<tr>
<td>26</td>
<td><a href="http://www.bluetooth.com/Bluetooth/Learn/Works/">http://www.bluetooth.com/Bluetooth/Learn/Works/</a></td>
</tr>
<tr>
<td>28</td>
<td>EXIF stands for Exchangeable Image File Format. This is a format that stores generated information of the image file such as: Camera Type, Date Taken, Longitude, and Latitude (Kurti et al., 2006).</td>
</tr>
<tr>
<td>29</td>
<td>Geographical identification metadata – including latitude and longitude coordinates.</td>
</tr>
</tbody>
</table>