Development Methods for Web Services

Jakub Walaszczyk

LYNGBY 2004
MSc THESIS
NR. 52/2004

IMM

Trykt af IMM, DTU

Abstract

This thesis was written in the Department of Informatics and Mathematical Modelling at the Technical University of Denmark.
The work was carried out in a period 16.02.2004 - 19.07.2004. The project was supervised by Michael Reichhardt Hansen, Associate Professor, Ph.D.
The project was focused in the area of Web Services. The final result was a successful implementation of a working system. Analyses of the technologies was performed and discussion of the obtained results presented.

Preface

This thesis is dedicated to the analysis of connectivity, interoperability and integration of entities and services on the Internet. The project may be separated into two main parts: a conceptual part, requiring research into the areas of Web Services and a practical part, involving the successful design and implementation of a particular Web Service.
In the theoretical part, in-depth analysis and discussions of service interfaces and protocols with consideration of security aspects will be carried out. The analysis will focus mainly on a software side of the problem, including examination of the Web Services standards such as SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) and UDDI (Universal Description, Discovery and Integration).
The practical part will be to formulate a design and implement a Web Service that will gather data from some independent sources in order to transmit the data to the end-user. This service will take into account crucial characteristics of an efficient solution (including scalability, performance and security). The project will take into consideration examples of Web Services, such as location services for GSM, fleet-management (e.g. taxi-companies), maps, hotel reservations, holiday planning, ticket booking etc. The chosen implementation will include optional utility of a data visualization user-interface. The solution will be implemented in Java.

The author would like to take this opportunity to thank Michael Reichhardt Hansen, Associate Professor for his supervision and guidance throughout the work on the thesis. Special thanks go to my mother, my sister and my girlfriend and to my colleagues Bogusaw Pilich and Tomasz Cholewiski.

Contents

1 Introduction **15**

1.1 The thesis questions 16

1.2 The thesis goals 16

1.3 Intended audience 16

1.4 Methodology 16

2 Background **17**

3 Middleware technologies comparison **19**

3.1 RPC 19

3.2 DCOM 19

3.3 RMI 20

3.4 CORBA 20

3.5 RMI/IIOP 21

3.6 Summary 21

4 Web Services **23**

4.1 The functional model 23

4.2 Technology 24

4.2.1 Transport 25

4.2.2 XML 25

4.2.3 WSDL 25

4.2.4 UDDI 26

4.2.5 SOAP 28

4.3 Future Indirections - WS-Specifications 29

4.4 Java specific concepts . . . 30
4.4.1 JAX-RPC . . . 30
4.4.2 SAAJ . . . 31
4.4.3 JAXP . . . 31
4.4.4 JAXB . . . 31
4.4.5 JAXR . . . 31
4.5 Applications/Examples . . . 32
4.5.1 Danske Bank . . . 32
4.5.2 Winterthur . . . 32
4.5.3 Bank of Tokyo-Mitsubishi (BTM) . . . 32
4.5.4 Amazon.com . . . 33
4.5.5 Summary . . . 33

5 Case study: Fleet management 35
5.1 Map Service . . . 35
5.1.1 Authentication . . . 36
5.1.2 Map request . . . 37
5.2 Location Web Service . . . 37
5.3 Fleet Management Web Service . . . 37
5.4 Clients . . . 38
5.4.1 Fleet Management Client . . . 38
5.4.2 Java Map Service Client . . . 38
5.4.3 .NET Map Service Client . . . 38
5.5 Models and message flow . . . 38

6 Implementation 41
6.1 Java Tools . . . 41
6.1.1 WSDP . . . 41
6.1.2 Tomcat . . . 42
6.1.3 Ant . . . 42
6.1.4 Eclipse . . . 43
6.1.5 WSDL2Java - plug-in . . . 44
6.2 .NET Tools . . . 44

6.2.1 Visual Studio .NET 44
6.2.2 IIS 45
6.2.3 WSE 45
6.3 Environment setup 45
6.3.1 Java platform setup 45
6.3.2 .NET platform setup 46
6.4 Implementation process 46
6.5 Discussion 48
6.5.1 Performance 48
6.5.2 Security 48
6.5.3 Transactions and parallelism 49
6.5.4 Platform comparison 49
6.5.5 Possible improvements 50

7 Conclusions **53**

8 Bibliography **55**

A Appendix A **61**
A.1 ArcWeb Client 61
A.2 ArcWeb .NET Client 62
A.3 Fleet Management Java Client 63

B Appendix B **65**
B.1 Form.java 65

C Appendix C **83**
C.1 fleetIF.java 83
C.2 fleetImpl.java 83
C.3 fleetMember.java 89
C.4 build.xml 90
C.5 web.xml 97
C.6 jaxrpc-ri.xml 97
C.7 build.properties 98

D Appendix D **101**

D.1 locationIF.java 101

D.2 locationImpl.java 101

D.3 location.java 102

D.4 build.xml 103

D.5 web.xml 110

D.6 jaxrpc-ri.xml 110

D.7 build.properties 111

E Appendix E **113**

E.1 client.cs 113

List of Figures

2.1 Middleware layer in simplified OSI model 17

3.1 RMI architecture 20
3.2 CORBA architecture 21

4.1 Web Service model 23
4.2 Web Service model with transport example 24
4.3 WSDL document structure 26
4.4 UDDI model[10] 27
4.5 SOAP message with optional WS-security extension 28
4.6 SOAP message flow 29
4.7 Web Services Specifications[28] 30

5.1 Fleet Management System concept 36
5.2 Map request and location request including two clients 39

6.1 Eclipse IDE 43

List of Tables

3.1 Middleware technologies comparison[21] 22

Chapter 1

Introduction

Since the Internet came into existence, it has developed into a tool used by both businesses and consumers alike. The number of computers interconnected through the Internet has been increasing rapidly. From the very beginning, the need to exchange data was crucial for the successful growth of the Internet. Nowadays, there are many methods available to both end-users and programmers that can be used to transfer any type of data. WWW and e-mail are probably the most commonly used end-user protocols over the Internet. Most of the low-level protocols however are unknown and invisible to the end-users. Programmers and network administrators are the ones who have to deal with the various implementations of protocols that lay hidden beneath the visible interface. At the present time the complexity of the applications and methods of data exchange slow down the development processes and introduce significant integration difficulties to many development projects. One of the most appealing innovations to the data exchange process was the introduction of the eXtensible Markup Language (XML) by World Wide Web Consortium (W3C), which dramatically improved the ways of finding a common form for various data representation. XML allows many hardware and software platforms to integrate by establishing common data documents. XML however, does not provide a programmer with a data exchange tool. The data transfer has to be performed independently and is not specified by the XML standard itself.

On the other hand, new data exchange methods have been constantly introduced. Unfortunately, many ideas provided by various vendors became platform specific or/and the complexity of the solution stopped its growth, thus did not ever become a common solution in most systems.

Web Services were introduced specifically to address the need to provide a common programming tool available on all platforms in order to simplify integration and interoperability of various existing and newly developed applications on the Internet.

In general, the XML Web Services are software components that provide a diversity of functionalities over the Internet. This was first introduced and standardized by the W3C Consortium in 2001. Web Services are rarely a complete solution operating on its own. In general they serve as the functional components of larger applications. Mainly they are

oriented to provide means of interaction between different software applications, running on a variety of platforms.

Web Services however will not replace World Wide Web. Their purpose is different. The most typical implementation is that of a service that providing information that might already be accessible through a standard WWW server (or any other way). The point is, to provide the same information in an easily accessible way for other software modules. This is to help cooperation of different software components that are implemented by independent programmers.

1.1 The thesis questions

How to use Web Services? What are the possible usage scenarios? What tools are needed for development? What are the alternatives, if any?

1.2 The thesis goals

Analysis of technology aspects of Web Services. Development methods of a successful service and client applications. Comparison of existing alternatives.

1.3 Intended audience

In the view of the experience gained during the work on the thesis, it became apparent that the intended audience is to be software developers and project managers that are willing to start working on distributed projects.
In order to obtain a perspective of how to approach the problem of integration of distributed systems using Web Services in particular.
Ideally, the knowledge of network programming is recommended and understanding of most common Internet technologies is required. It is advisable that familiarization with XML technologies is done as a prerequisite.

1.4 Methodology

The study was performed by looking up Internet resources, publications and books. The implementation of the program was performed on a standard PC.

Chapter 2

Background

In the history of the Internet there were many attempts to create a standard way for communication between different software components. Many of them were very successful, while some of them did not get much of the Internet community's attention. In this study the main focus is placed in the area of Web Services. However, in order to understand the reasons for the introduction of this new technology, there is a need to move back in the past and look at the history of protocols evolution. The most typical approach that came in use in the 80s was Client/Server architecture, which in principle was data-centric. The main scope of the inventors was more fixed on successful data retrieval and display, than on the particular roles in the complex distributed systems. In order to provide more advanced features in service-like applications, so-called middleware technologies have since been established. The name middleware originates from a layer of software between the network and the application layers (in accordance with OSI 7 layer model). Web Services architecture however exists across multiple OSI layers and extends beyond the scope of a single layer definition. The following diagram describes the middleware location with respect to the OSI model.

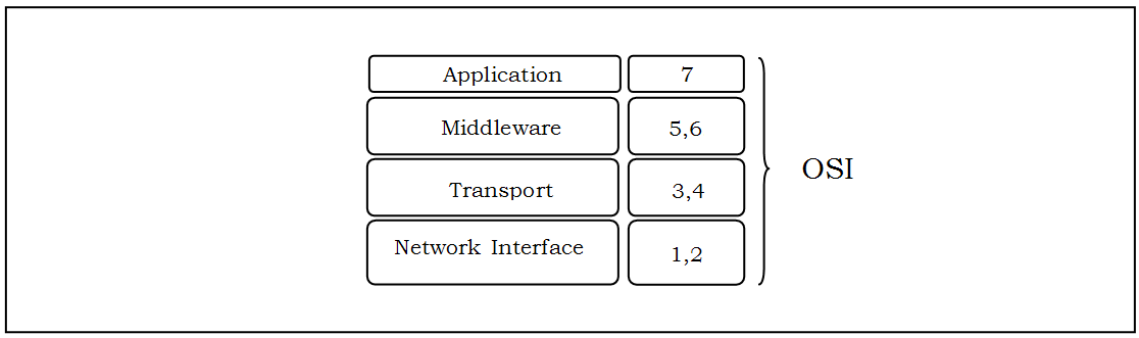


Figure 2.1: Middleware layer in simplified OSI model

In the recent years the tendency has changed. Nowadays network applications start moving towards a Service-Oriented Architecture (SOA), which is defined as follows:

"A service-oriented architecture (SOA) is a component model that inter-relates the different functional units of an application, called services, through well-defined interfaces and contracts between these services. The interface is defined in a neutral manner that should be independent of the hardware platform, the operating system, and the programming language the service is implemented in. This allows services, built on a variety of such systems, to interact with each other in a uniform and universal manner." [7]

There are many distributed technologies that comply with the SOA approach. Just to mention a few: Remote Procedure Calls (RPC), Remote Method Invocation (RMI), Common Object Request Broker Architecture (CORBA) and many others. Above technologies are inherently service-oriented and they will be described more in detail in the Chapter 3. By promoting interoperability between middleware, new technologies are supposed to make distributed applications much easier to develop. Unfortunately, none of the mentioned technologies evolved into a dominating standard. Nonetheless, Internet applications have advanced significantly throughout the years. Today, not only simple applications are within reach, but also very complex solutions for diverse distributed architectures scattered in the midst of the entire Internet. The reason for not finding a common middleware standard may have been due to the lack of agreement between the dominating software vendors (Microsoft, Sun, IBM) and the Open Source Community. The answer to the above problem is believed to be Web Services. In order to fully understand why if this is possible to happen, more detailed analyses of the alternative technologies will follow.

Chapter 3

Middleware technologies comparison

The contents of this section cover the most popular distributed technologies used in Internet development work. It serves as a basic overview to provide a wider perspective for understanding principles of Web Services which are described in detail in the next section. The majority of the data provided below was gathered from on-line specifications published by founders of the technologies on their websites, where if needed more additional information can be found.

3.1 RPC

Remote Procedure Call (RPC) was first introduced by The Open Group. It is a concept defining how a program can call a procedure on a remote machine. The running application executes a code from a program located on a remote computer in a network without a need to understand the network infrastructure. RPC in principle uses the client/server model. Similarly to a local procedure call, RPC is a synchronous operation; it requires the requesting program to be suspended until the result of the remote procedure is returned. This can be overcome however by using threads. Before an RPC call is done, a client and a server have their stubs created for their remote functions. This is usually done with use of an Interface Definition Language (IDL). When the call is performed, the arguments that are passed to the remote function are marshalled and sent to the server. RPC in principle is not object-oriented (OO) thus it lacks features required in modern programming languages. RPC was the fundamental idea that was later used and enhanced in different platforms.

3.2 DCOM

Distributed Component Object Model is a solution introduced by Microsoft. It uses basic concepts first found in RPC. Its first implementation known as COM was designed for local

usage only. DCOM is a complete distributed solution, widely used in Windows Operating System. The goal of Microsoft was to make DCOM a standard for all platforms including UNIX. However implementations other than Windows are insignificant. The major DCOM advantage is it is fully standardized by one vendor ensuring compatibility among all the systems (however Windows mostly). The specification provided by one vendor usually is very difficult to become popular at the competitor's platform. This is the main reason that limited the growth of the technology to one operating system.

3.3 RMI

Remote Method Invocation (RMI) is a Java equivalent to RPC. Traditional RPC systems are language-neutral, whereas RMI is Java specific. This means that RMI can function only among Java applications. That is the main disadvantage of this technology, which limits the number of possible applications. On the other hand, it provides an easy to use tool for Java oriented programmers, which takes advantage of Java specific security features and fully supports passing of any parameter types to invoked methods (including user defined objects). Furthermore, there is no overhead related to parameter conversion (this takes place when sending data to applications written in different programming language using other technologies). Limited usage possibilities caused by Java platform requirement, did not allow the technology to thrive and dominate in the Internet. The below diagram shows fisualization of a basic model behind the technology.

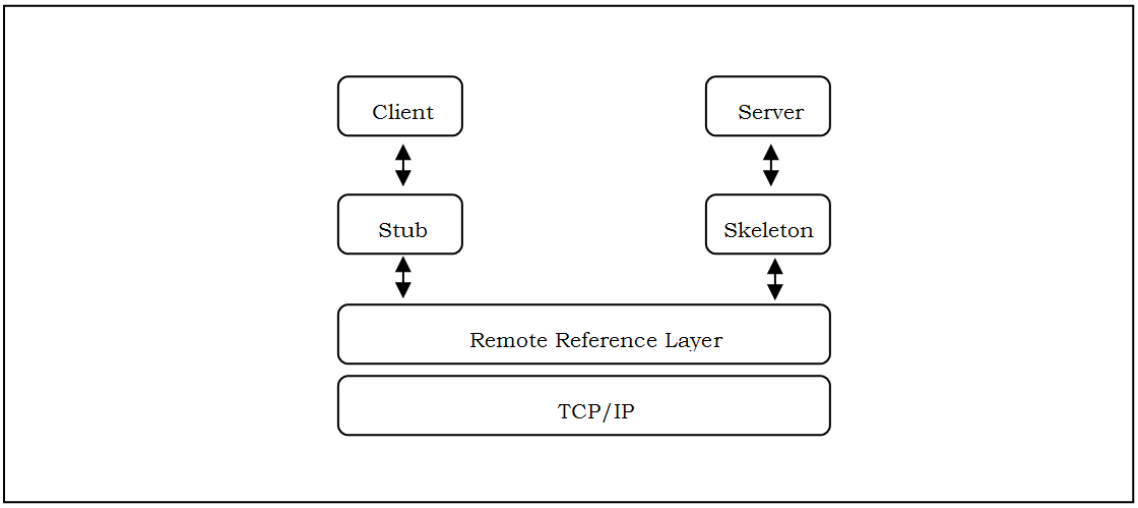


Figure 3.1: RMI architecture

3.4 CORBA

Common Object Request Broker Architecture (CORBA) was introduced in 1991 by Object Management Group (OMG). It is based on a concept of an Object Request Broker (ORB),

which manages communication and data exchange between objects. Both Client and Servant use IDL that support mapping of many popular languages and allow separation between interface and actual implementation. The ORB core uses the Internet Inter-ORB Protocol (IIOP) for data exchange The diagram below describes the overall architecture.

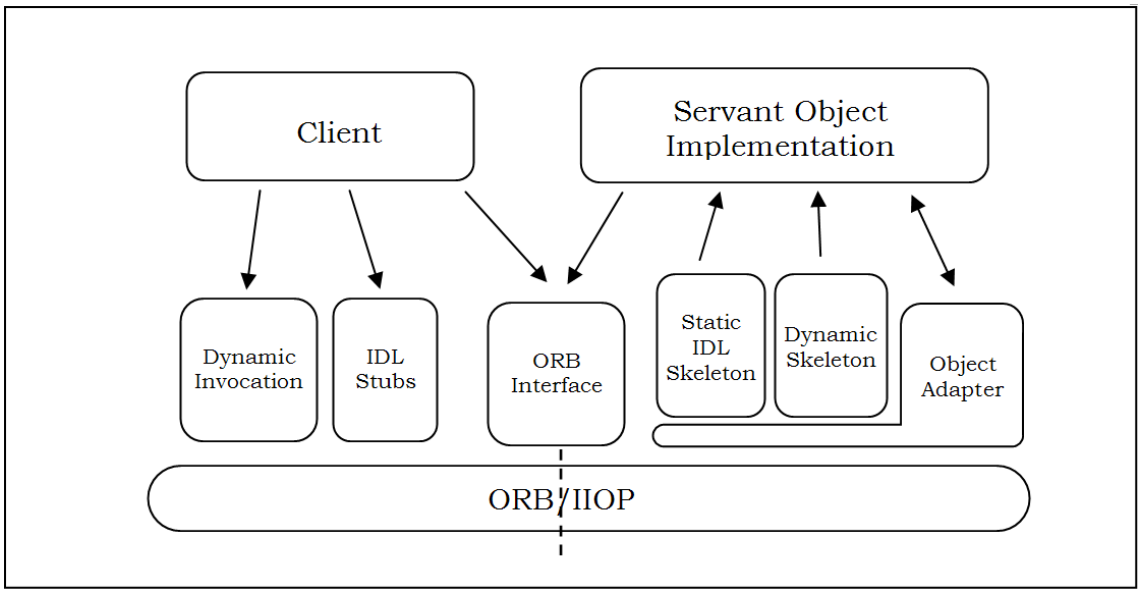


Figure 3.2: CORBA architecture

One of the issues with CORBA is number of ORB implementations that sometimes provide some vendor specific options that might not be fully compatible throughout.

3.5 RMI/IIOP

Java RMI over IIOP was developed by Sun and IBM. The major difference to standard RMI is a usage of CORBA ORB technology for transport, allowing integration with many languages and in the same time cancels the need of usage of IDL definitions. In other words Java programmers may develop their software using only Java API without need to understand specific mapping used in IDL. In other words RMI/IIOP consolidates features from two worlds: ease of use of RMI and multi-language support of CORBA.

3.6 Summary

From the above descriptions, it becomes clear that there are many powerful tools available for programmers willing to develop advanced distributed systems. However only a closer comparison of basic characteristics can highlight the advantages and disadvantages of particular designs. That is a role of the Table 3.6.

Model	Portability		Performance	Development Ease	Security
	Platform	Language			
RPC	Dependent	Dependent	Highest	Moderate	Medium
DCOM	Unlimited	Unlimited	High	Moderate	High
RMI	JVM	Java	High	Easy	High
CORBA	Unlimited	Unlimited	Moderate	Difficult	Medium
RMI/IIOP	Unlimited	Unlimited	Moderate	Easy	High
Web Services	Unlimited	Unlimited	Very Slow	Very Easy	High

Table 3.1: Middleware technologies comparison[21]

Regarding DCOM, implementations other than for Windows are rather insignificant. When using DCOM, RMI/IIOP, CORBA and Web Services, it is required an available mapping for the chosen language. When using Web Services a chosen language requires support for SOAP and XML.

To summarize, we notice that the number of possible technologies to choose from is relatively high, keeping in mind that there are others, which have not been discussed here (e.g. MOM, MPI, DCE, PVM, MQSeries, JXTA). Even so, focusing on the ones presented earlier, it seems that there is probably no need for the introduction of a new one. RMI/IIOP especially, provides already very flexible, relatively fast and reliable tool that could be used for any type of development ensuring security, interoperability and satisfying the need for integration with legacy systems. In order to find out what other possible concepts can be offered in the area of distributed programming, and if a dominant technology will emerge from these, thorough analysis of Web Services technologies has to be done and this is discussed in Chapter 4.

Chapter 4

Web Services

4.1 The functional model

The concept of Web Services can be divided into three main parts that form the whole system. When a service is created, it is first *published* in a *Service Registry* by a *Service Provider*. After that, a party interested in the usage of a particular service needs to find out where it is placed. This is a *discovery* of a service. The final stage is called a *consumption*, which is performed by a *Service Consumer*. The complete model is shown in the following figure.

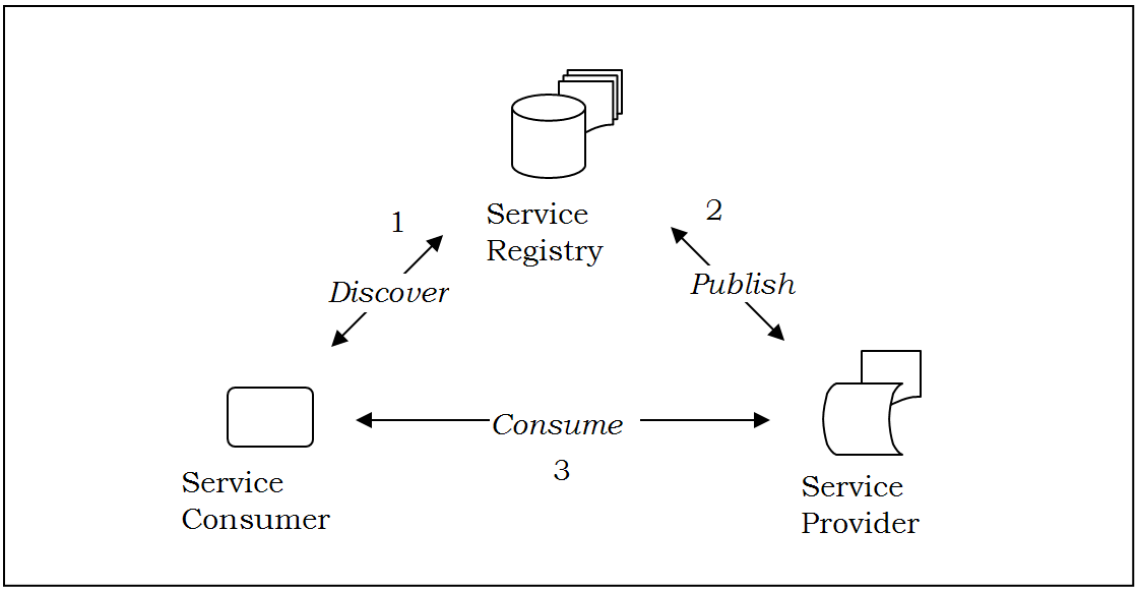


Figure 4.1: Web Service model

4.2 Technology

The functionality of Web Services is based on well-known Internet standards, including: Transmission Control Protocol (TCP/IP), Hypertext Transfer Protocol (HTTP) and Extensible Markup Language (XML). In addition, several new standards have been introduced, particularly for use with Web Services. The main are: Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), and Universal Discovery, Description, and Integration (UDDI).

- *Publication* - For ease of access, each Web Service is published in a UDDI registry. There are currently several major UDDI registries providing information on Web Services in a categorized manner. However, any means of making the service available for consumption is considered a publication.
- *Discovery* - The complete functionality of the Web Service is defined in a Web Service Description (WSD). The description of the protocol of the message exchange for interaction with the Web Service is expressed using WSDL.
- *Consumption* - Methods of consumption are defined in WSDL file, which is the basis for code generation in the given programming language, depending on chosen platform (e.g. C#, Java).

To visualize where each of the technologies is used the following diagram is provided.

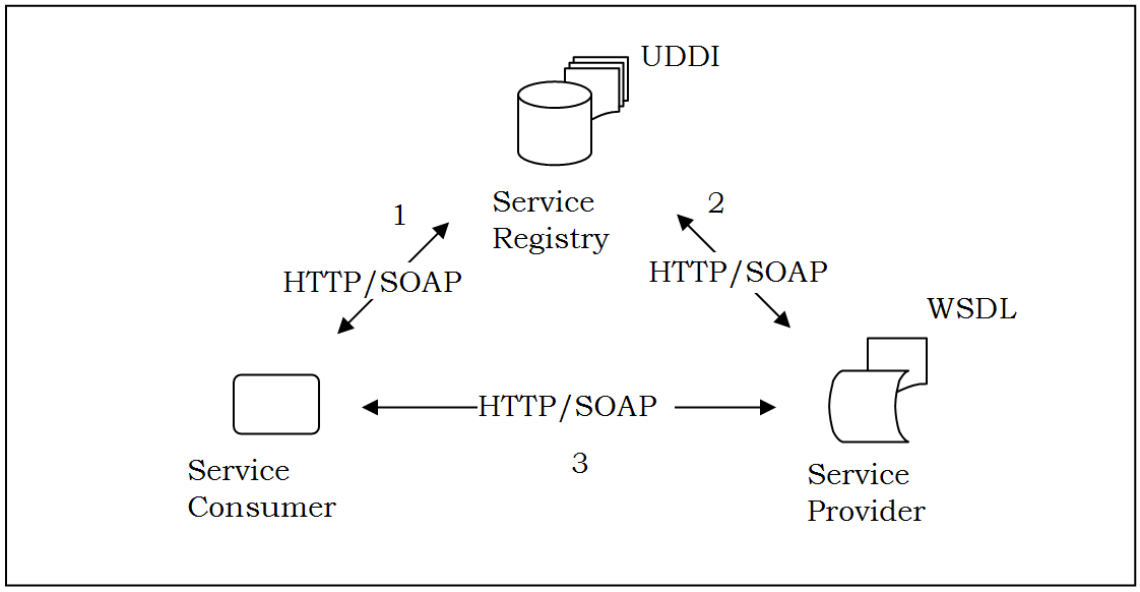


Figure 4.2: Web Service model with transport example

4.2.1 Transport

The preferable Web Service transport protocol is HTTP over TCP/IP. The reason for this is the wide deployment and firewall-friendliness of this protocol. Its popularity across a wide range of operating systems and dominating role in the Internet is also an advantage. Another advantage is the guaranteed built-in security. The concept of Web Services does not limit the programmer to one protocol, for example SMTP can be used as well.

4.2.2 XML

Extensible Markup Language (XML) was introduced in 1996 by the World Wide Web Consortium (W3C). The purpose of for developing this standard was to provide programmers with a common document format to represent any type of data. The idea was one of the most successfully applied in practice. At the present time, XML is supported by probably all modern operating systems and by all modern programming languages. It was very appealing to take advantage of this technology in order to facilitate its unquestionable features in the modern middleware solution.

4.2.3 WSDL

WSDL is described as *" an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint."* [8]

WSDL documents are the basis of interaction with Web Services. Their structure might be overwhelming at first but the syntax complexity is not immense. Each document is based on XML schema which describes the XML instance. The most appealing characteristic of WSD is its possibility for automatic generation of classes and methods in any given programming language.

The root element of a WSDL is represented by definitions. It may contain up to five different types of child elements.

- types - this element contains definitions of types that may be send and received in the messages
- message - a cross-reference associating the message with its definition.
- portType - definition of set of interfaces that can be exposed by a given Web Service.
- binding - association of the portType definition to a protocol
- service - defines all the ports exposed by the Web Service

The Figure 4.3 describes the structure of the elements inside the WSDL document.

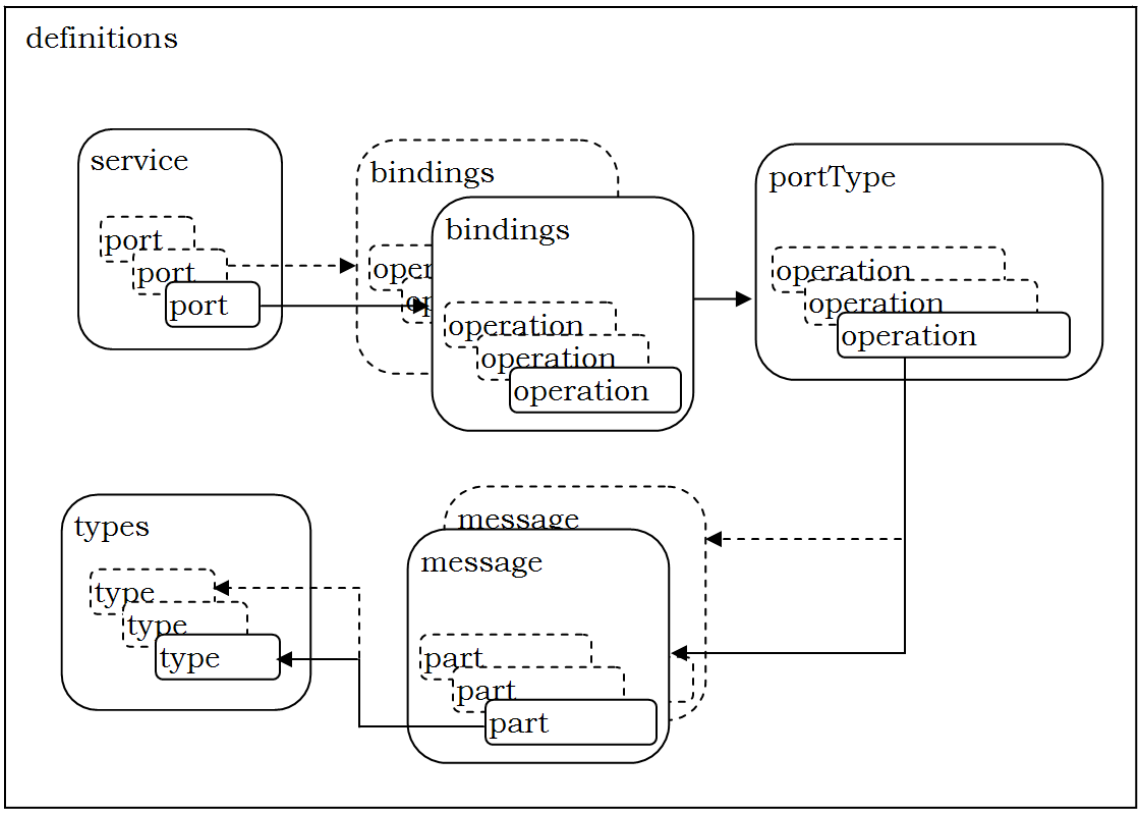


Figure 4.3: WSDL document structure

4.2.4 UDDI

"The focus of Universal Description Discovery & Integration (UDDI) is the definition of a set of services supporting the description and discovery of (1) businesses, organizations, and other Web Services providers, (2) the Web Services they make available, and (3) the technical interfaces which may be used to access those services." [10]

In the other words UDDI provides a means to publish the services on the Internet to make the information of their existence to be available to anyone. It is possible to make use of Web Services without publishing them in the UDDI registry.

The registry consists of several entities and relations between them: business entities - representing party providing a service, contains particular data about the entity (e.g. name, address, description) and its relation to the other companies (service providers)

- business service - grouping of logical services associated with the business entity
- binding template - technical information about particular Web Service, URL to the service
- technical model - specification implemented by Web Service, URL to the specifications
- publisher assertion - describes relationships between service providers

Instances of the above documents are identified by a Universally Unique Identifier (UUID). UUIDs are created in a strictly defined way depending on service providers' hardware, time of service creation, IP address, when the data is first published in the registry.

The following three types of messages can be send to and received from the UDDI server:

- inquiry messages - used for finding objects and getting information about them. (e.g. FindBusiness, FindService)
- publish messages - used for creation and management of server data. (e.g. DeleteBinding, SaveService)
- response messages - response messages to the above two sets of messages. (e.g. BindingDetail, BusinessDetail)

The simplified model of functionalities is provided in the Figure 4.4.

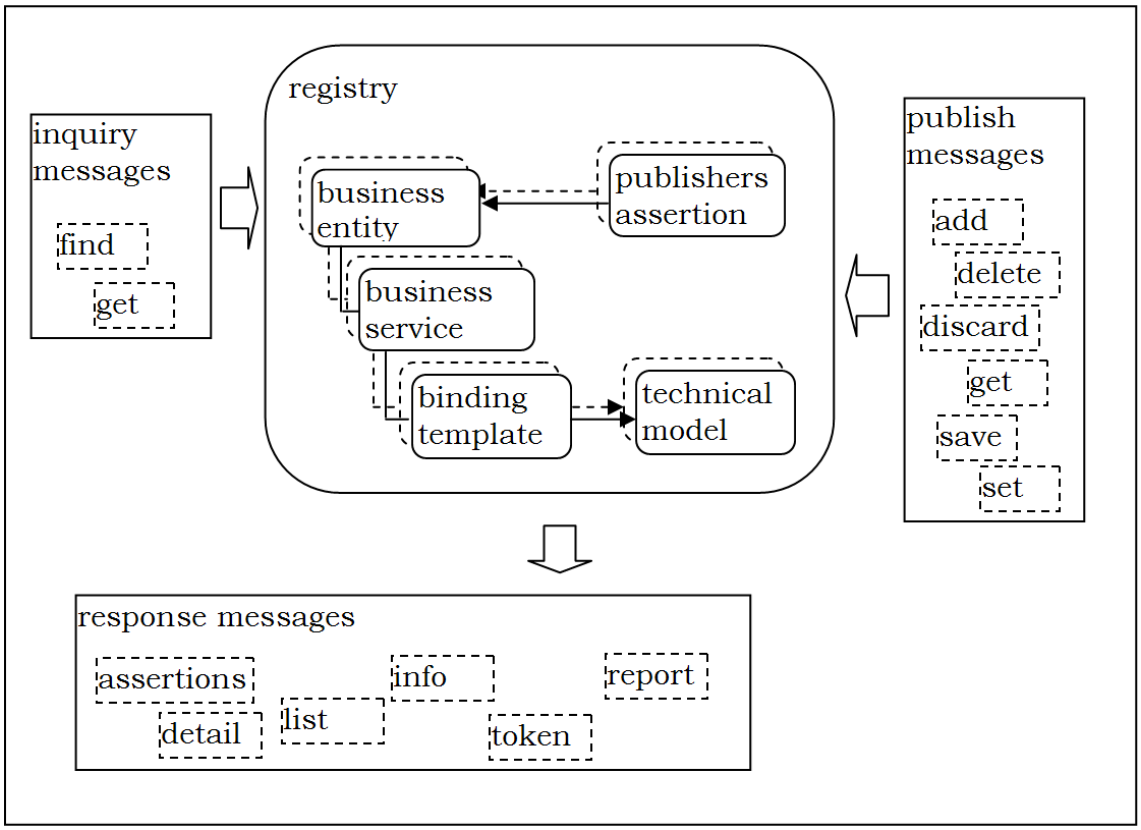


Figure 4.4: UDDI model[10]

4.2.5 SOAP

Simple Object Access Protocol (SOAP).

"SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols." [9]

The SOAP protocol is based on messages. Each SOAP message consists of a mandatory SOAP envelope and SOAP body with an optional usage of SOAP header.

The SOAP envelope
This a definition of *"an overall framework for expressing what is in a message; who should deal with it, and whether it is optional or mandatory."* [9]

The SOAP header.
Header is an optional element in a message. A header contains information about the message. In the optional WS-Security extension to SOAP header contains security information which includes e.g. signature or token.

The SOAP body.
This a definition of *"a convention that can be used to represent remote procedure calls and responses."* [9]

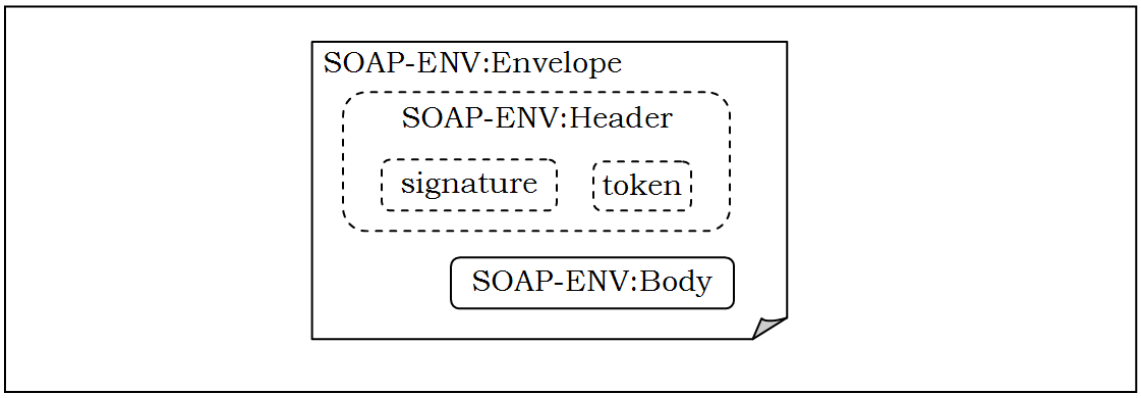


Figure 4.5: SOAP message with optional WS-security extension

The use of SOAP was primarily related to object serialization. Serialization is a process of transforming a runtime object into a stream of data. Deserialization is the opposite process which recreates a runtime object from a stream (in this case an XML document).

How SOAP fits in the Web Services model is depicted in the Figure 4.6.

Web Services are under constant development thus some of the new features and extensions appeared during the work on this thesis. WS-Security has been adopted by OASIS as a

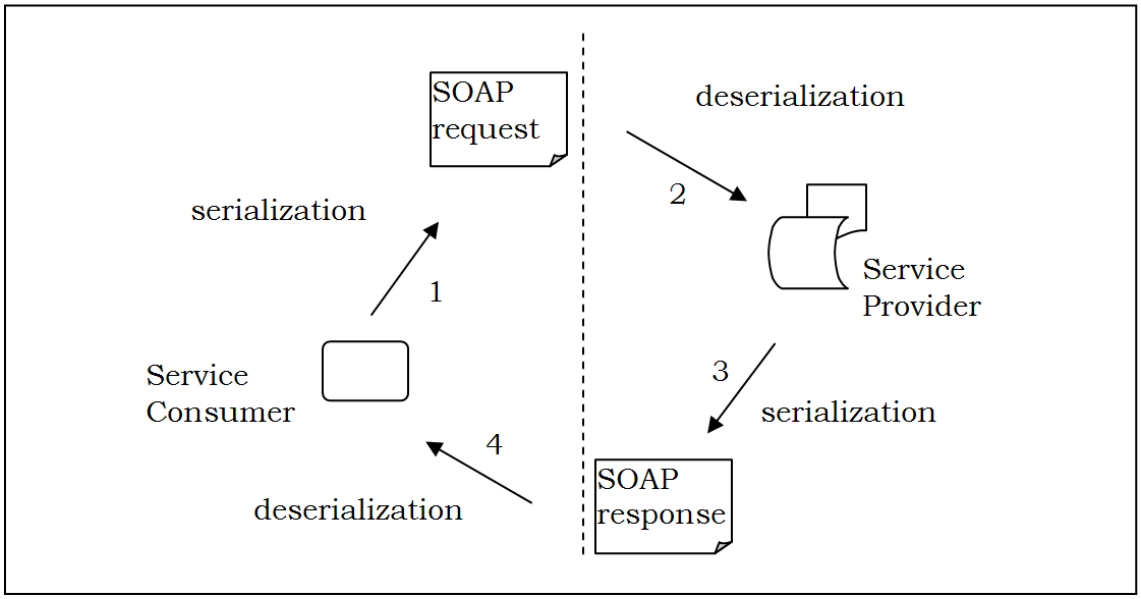


Figure 4.6: SOAP message flow

standard to SOAP on April 6th 2004. WS-Security WS-Security is an extension to SOAP protocol. It enhances existing specification by additional security features including integrity, confidentiality, and authentication. Provided definitions allow usage of a diversity of security models and encryption tools. WS-Security does not ensure security by itself and it obviously does not guarantee that the chosen security solution will not be compromised. *"WS-Security also provides a general-purpose mechanism for associating security tokens with messages. No specific type of security token is required by WS-Security. It is designed to be extensible"* [11] "Additionally, WS-Security describes how to encode binary security tokens." [11]

4.3 Future Indirections - WS-Specifications

It is clearly visible, that there is a lot of industry pressure on developing advancements and further improvements to Web Services. Many major software vendors e.g. Microsoft, IBM, BEA and some other companies are currently intensively working on WS-Specifications. Most of these specifications are in their draft phases and are a subject to change. The following specifications listed below are under intense development at the present time

- Messaging Specifications
  SOAP, WS-Addressing, MTOM (Attachments), WS-Eventing
- Security Specifications
  WS-Security, WS-SecureConversation, WS-Trust, WS-Federation, WS-Federation Active Requestor Profile, WS-Federation Passive Requestor Profile, Web Services Security Kerberos Binding, WS-security - it has already been incorporated in the SOAP specification.

- Reliable Messaging Specifications
  WS-ReliableMessaging
- Transaction Specifications
  WS-Coordination, WS-AtomicTransaction, WS-BusinessActivity
- Metadata Specifications
  WSDL, UDDI, WS-Policy, WS-PolicyAssertions, WS-PolicyAttachment, WS-SecurityPolicy, WS-Discovery,
- XML Specifications
  XML, Namespaces in XML, XML Information Set , XInclude

The above specifications are to fill up a model depicted below:

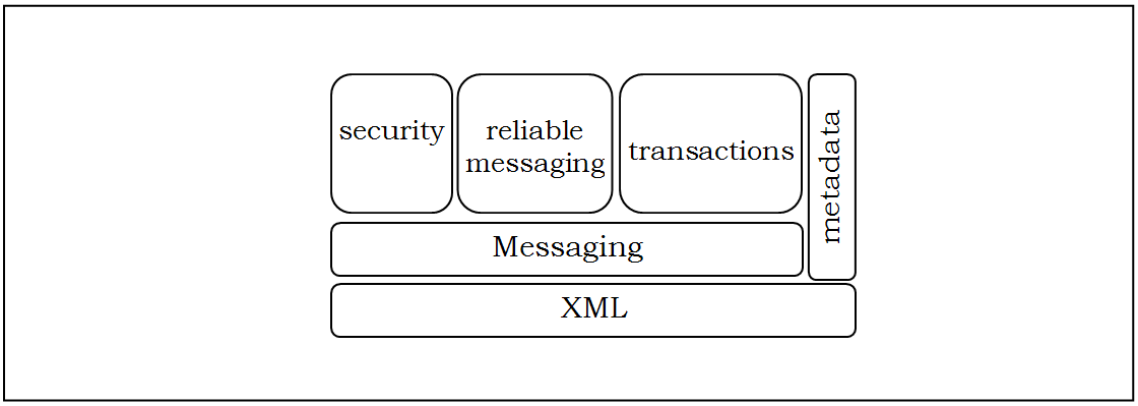


Figure 4.7: Web Services Specifications[28]

4.4 Java specific concepts

The fundamental concept of Web Services is platform independent. Applications communicating across networks may be running on various operating systems and be written in various languages. Unfortunately, implementation tools are almost completely platform specific. Anyone who wants to develop and take advantage of Web Services functionalities has to familiarize with the tools available for the chosen platform and the chosen programming language. Considering Java in this approach, there are several important ideas that a newcomer has to be familiar with. Some of the below definitions introduce more confusion and complexity than there is really need for. However without clear understanding of these notions, further analysis might be difficult.

4.4.1 JAX-RPC

JAX-RPC stands for Java API for XML based RPC. This is the core API used for Web Services in Java.

4.4.2 SAAJ

SAAJ stands for SOAP with Attachments API for Java. It provides the API for creating and building SOAP messages. To put it simply, SAAJ is the java.xml.soap package that must be imported during the development of Web Services or any other application in Java that may use SOAP.

4.4.3 JAXP

JAXP stands for Java API for XML Processing. It includes Document Object Model (DOM), Simple API for XML (SAX) and Extensible Stylesheet Language Transformations (XSLT) APIs. DOM and SAX are XML parsers and XSLT is a transformation engine for XML.

4.4.4 JAXB

Java API for XML Binding "provides a convenient way to bind an XML schema to a representation in Java code. This makes it easy for" ...a programmer... "to incorporate XML data and processing functions in applications based on Java technology without having to know much about XML itself."

4.4.5 JAXR

The Java API for XML Registries gives "a uniform way to use business registries that are based on open standards (such as ebXML) or industry consortium-led specifications (such as UDDI)"

4.5 Applications/Examples

The following case studies give an insight of how Web Services are influencing the current outlook of software development. Moreover, this section is to provide some proof points for successful real-life applications of Web services.

4.5.1 Danske Bank

Dankse Bank is the biggest commercial bank in Denmark. The banks first Web service was a stock quote service. The development of Web Services was based on usage of .NET platform.
This information below was provided by Microsft:

> *With Microsoft .NET-connected software and Web services, the bank can build a set of components that can be reused and reconfigured repeatedly to expose its vast store of mainframe-based computing assets to a broad range of customers and partners. Because of the standards-based design, the bank has been able to harness the unique strengths of many technology platforms and combine them into a single services architecture that makes the most efficient use of computing resources*[38]

From the above we can see that Microsft eagers to promote this technology and that applications in the most demanding environment, where security plays the crucial role are possible.

4.5.2 Winterthur

Winterthur is a Swiss insurance company. It has decided to integrate their systems using Web Services. This company selected to apply integration to both Java and .NET components.

> *Today, Winterthur has standardized its Swiss operations on its new platform and is rolling it out globally. Like modern Web services, the platform's application infrastructure relies heavily on reusable application components connected through a standard messaging and interface platform. The difference is that Winterthur had to do much of the heavy lifting that standard platforms built around Microsoft's .Net and Sun's Java provide today because those technologies inevitably need to integrate into the existing environment.* [39]

4.5.3 Bank of Tokyo-Mitsubishi (BTM)

BTM is a member of the Mitsubishi Tokyo Financial Group. It is the world's fourth largest bank which total assets excess 94 trillion Yen.[40]

Bank of Tokyo-Mitsubishi is the first Japanese bank to implement a Letter of Credit service using Web services and Internet standards. This approach provides BTM with a flexible platform that can evolve in phases, capitalize on the existing technology investments at BTM and BTM's clients, and rapidly respond to market opportunities and client requirements.[40]

4.5.4 Amazon.com

Amazon is another big company that eagers to take advantage of Web Services. Their goal is to provide *"a direct access to Amazon's technology platform"*[41]. By using the Amazon Web Services, it is possible to *"access catalog data, create and populate an Amazon shopping cart, and even initiate the checkout process"*[41].
The Amazon Associates, can use the company's catalog data to create sites featuring Amazon products, *"including accurate and timely product pricing and availability"*[41].

4.5.5 Summary

All the earlier mentioned examples show that Web Services are suitable for huge integration projects, where a lot of security requirements must be met. They prove that the major vendors are ready to provide solutions based on this technology. In Chapter 5 it is shown, that the application scope of the technology is much broader and that it is feasible to implement solutions in any given domain.

Chapter 5

Case study: Fleet management

Fleet Management is a logistical requirement for various businesses. It is a concept that can be used in various domains such as taxi companies, package delivery firms and many others. The system developed provides a backbone for the expansion to a truly functional Fleet Management. The system that has been implemented does not intend to solve all the issues that might be met in a real life application; the important point is to take advantage of the Web Services technology and highlight the most crucial aspects of a successful design. The implementation goal was completed in a way that emphasizes the flexibility and the interoperability of Web Services. The main idea behind the creation of the system is to design a model that would allow customers of a company to contact one party that provides the complete service. This is not as straightforward as it seems at first glance, because there are more parties involved in the whole data flow of the scheme. Moreover there are certain limitations due to the use of a third party service (this will be discussed in the following sections). The main components of the system are: the client applications, the main service, the map service and the location service. All the components will be described in detail in the following sections, but in order to give an insight to the complete architecture, figure 5.1 is provided.

The objects with dashed lines represent components that can have multiple instances. Where the dotted component is the only component implemented using .NET technology, the rest of the components are written in Java. The third party Map Service implementation technology is unknown and not relevant in this study.

5.1 Map Service

Before the analysis of the main components of the implemented solution is undertaken, it is advisable to first take a look at the third party Map Service. The search for the right map service was not an easy task. Access to most of the Web Services was not free and required time consuming registration processes. After looking through several different types of available services, the ArcWeb map service was chosen. It provides very complex functionality and data visualization methods; moreover it is very well documented. The

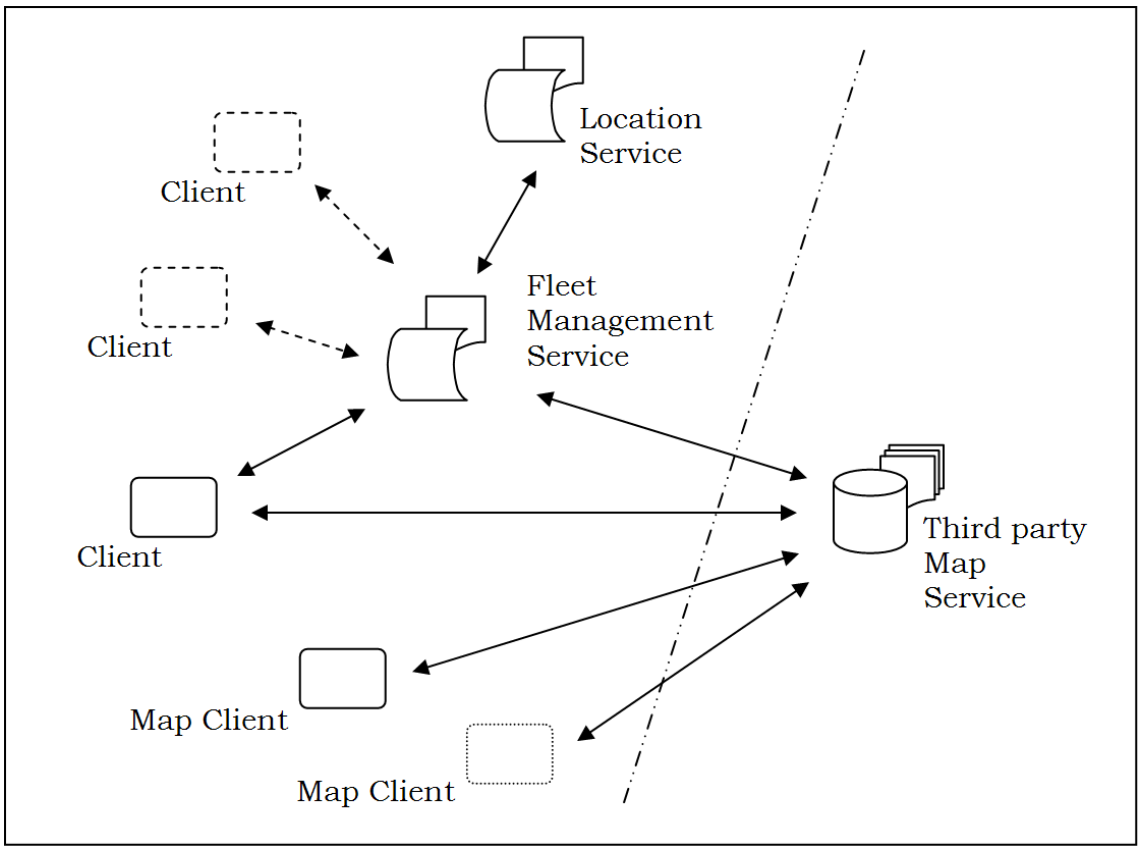


Figure 5.1: Fleet Management System concept

company website provides many useful examples and seems to be a reliable partner for cooperation. It was crucial for the service to be available throughout the development and implementation of this project. The first step to get access to the website was the registration. This is organized in an old fashion way by using an e-mail. After completing the registration process the user is provided with a username and password, which is later used for authentication to the service. There are three steps to complete in order to receive the required map.

5.1.1 Authentication

Every time a customer wants to use the map service, he is obliged to authenticate using a specific method exposed by the Web Service. This is done by sending of the username and the password. As a result, the remote method returns a token. Each token has a customizable validity time. After expiry, the token has to be renewed. The token is later sent to the service with every request. This communication is encrypted and based on HTTPS protocol.

5.1.2 Map request

Surprisingly, the map request consists of two separate requests. First the client executes a method providing the required map parameters (e.g. coordinates, map size, image format and various elements to be displayed on the map). In return a URL to the image is given. There are probably two reasons for such a solution. One reason is the integration with a legacy system which was responding to HTTP GET requests. The company may have created a Web Service which "translates" a remote method call to a GET request. This simplicity may at the first glance appear a little awkward. However the second reason may be the actual increase of the performance due to the lack of serialization of the image object, which is instead sent directly via an HTTP connection.

5.2 Location Web Service

The purpose of this service was to enable another data providing party in the whole process of data accumulation. This application is emulating a real-life location provider e.g. mobile operator. This service is relatively simple, comparing to the other applications. Introducing complexity was not necessary due to the lack of need for implementation of this application in the real-life scenario. The service exposes a single functionality which is location data retrieval for a given subscriber. The only method exposed is a localization of a subscriber. The parameter sent is the subscriber's phone number. The return data are the subscriber's position co-ordinates. The application is implemented as a Java Servlet, which is deployed to a Servlet Container. The Servlet technology detailed description can be found in the next Chapter.

5.3 Fleet Management Web Service

The main purpose behind a Fleet Management system is to create a Fleet Management Web Service. The role of this service is to provide various functionalities e.g. data management and data accumulation for client applications. The Fleet Management service allows multiple clients to operate simultaneously. This application is the most complex of all that were created in this scenario. It is both a service provider and a service consumer. The application is implemented as a Java Servlet. The component has to be deployed to a Servlet Container and then the container itself manages the clients' connectivity. This service uses a map service and a location service. In addition it provides features needed for an effective management of the fleet, including:

- adding a fleet member
- removing a fleet member
- modifying a fleet member
- setting the complete fleet
- retrieving the fleet
- locating particular fleet members

This application stores all the Fleet Members information, exposes methods for accessing and altering these data.

5.4 Clients

There are three types of clients developed in this scenario. The rationale behind the creation of these clients was to understand possible ways the services can be accessed and used. Moreover it was also needed to see the differences between the particular implementations. The differences between Java and .NET clients were considered especially noteworthy. All the three clients have a common tool for map visualization. It provides multiple features which include zooming, map panning and map reload. Java clients share the same code, when .NET client code had to be refactored. Both Java clients use Java Swing API for constructing the Graphical User Interface (GUI). The detailed description will follow in the below sections.

5.4.1 Fleet Management Client

This client invokes the remote methods on the Fleet Management Service directly. Apart from the map visualization tool it provides an interface for the Fleet Management. This interface is operating on the remote implementation located in the Fleet Management Service. The client application provides functionality available via menu, which allows locating any given Fleet Member.

5.4.2 Java Map Service Client

This client invokes the remote methods on the Map Service directly. It provides a map visualization tool. The client was developed at the start of project on the thesis, in order to gain a hands-on experience with Web Service using Java technology.

5.4.3 .NET Map Service Client

This client invokes the remote methods on the Map Service directly. It provides a map visualization tool. This client was also developed in the early stages of the project, in order to gain experience with Web Service using .NET technology.

5.5 Models and message flow

The entities involved in the process of data exchange are interacting in a specific manner. When a user of the Fleet Management Client would like to visualize a position of one, or more of the fleet members, has to perform an action. The first step is two manually request localization of a fleet member in order to get ”fresh” positioning information and

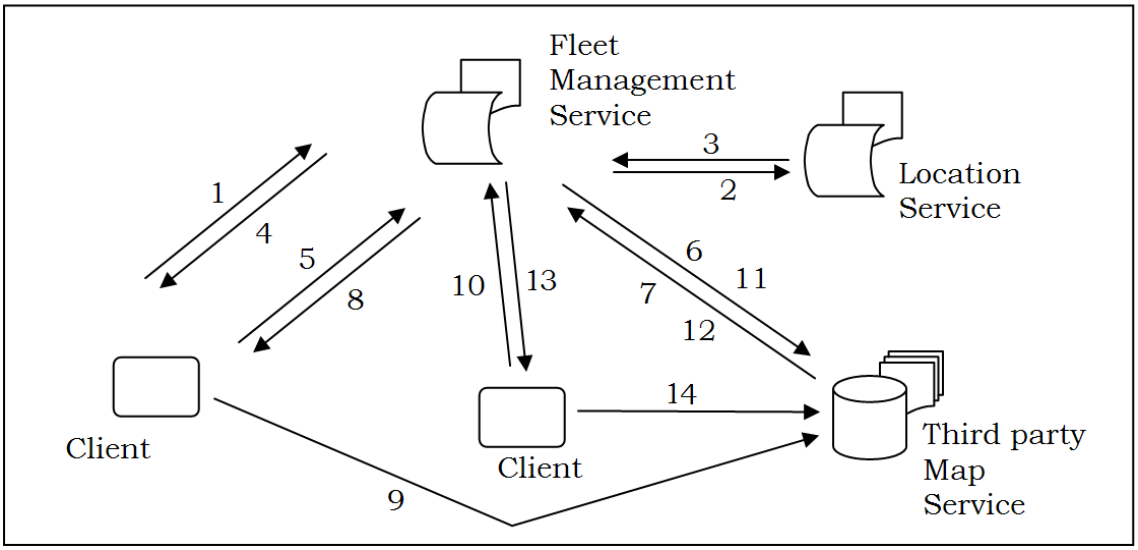


Figure 5.2: Map request and location request including two clients

then the second step is to request the map with the given fleet member to be displayed on the user's screen. The figure 5.2 shows the sequence of actions.

1. the location request is sent to the fleet management service.
2. location request is sent to the location service
3. the current position of the fleet member is returned
4. the user is informed that a new location is available
5. the user requests a map
6. the location data is combined into the map request
7. returned URL arrives to the service
8. client application receives the URL
9. and automatically requests the map
10. the second client request the map with the position of the same fleet member
11. the old position information is used
12. URL is returned to the service
13. and then to the client
14. map is downloaded

In the above sequence of action we can notice that the location data is cached on the fleet management service. It works both for one client and more clients. If one client localized a fleet member, all clients have the information renewed after reloading the map. This is particularly important when data from location provider is difficult(slow) to access or/and expensive.

Chapter 6

Implementation

This chapter covers the description of the complete process and discusses the tools used during the implementation. It also describes the implementation of applications and services and includes the models of software modules. Moreover it deals with problems that were encountered during the work. Before going into technical details of the implementation, it is worth taking a closer look at the tools used.

6.1 Java Tools

In order to get a concrete understanding of and experience with Web Services, the choice of proper development tools is essential.

6.1.1 WSDP

Web Service Development Pack (WSDP) by Sun Microsystems

"The Java Web Services Developer Pack (Java WSDP) is a free integrated toolkit you can use to build, test and deploy XML applications, Web services, and Web applications with the latest Web service technologies and standards implementations"[6]

The pack consists of several components that can be downloaded separately, which are: Ant and Tomcat. It also includes several examples and a useful tutorial. This toolkit has been under constant development and probably still is. When this document was completed version 1.4 was available. However in the start of the project earlier version 1.3 had to be used. These are recent changes that were not present at the development process:

"The Java WSDP 1.4 implements the WS-I Basic Profile 1.1 with WS-I Attachments Profile 1.0, which adds support for sending attachments with SOAP messages. Additionally, the Java WSDP 1.4 supports a full implementation of the OASIS Web Services Security (WSS) specification to provide message-level security for SOAP, which allows messages to

be sent and stored securely independent of transport or storage security measures."[6]

6.1.2 Tomcat

Tomcat is an open-source project developed by members of Apache Software Foundation.

"Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies."[30] In this project Java Servlet technology has been used. Web application is deployed to the Servlet container and handles the incoming requests. This is described more in detail in the next subsection.

Java Servlets

Java Servlets is a Sun Microsystems technology which is briefly defined as follows:

"A Servlet is a Java technology-based Web component, managed by a container, that generates dynamic content."[29]

The purpose of this is to enable Java code to run in a Servlet Container in order to handle clients requests. The main advantage of the Servlet is that it is first loaded into memory when the container starts. This means that it is not reloaded at every new request. Servlets are accessed by many clients without a need of creating new processes as it happens in some multi-threaded applications. The multi-threading is managed by the Servlet container. There is only a single instance which answers all requests concurrently, thus a very important thing during the implementation is to remember: ensuring a proper thread-safety by using monitors on sensitive data. In order to function, each Servlet has to be deployed to the Servlet Container, but before that happens, it can be first packaged and signed into a WAR file.

WAR

Web ARchive format (WAR) is a file used to pack Web applications that are to be deployed into the application container. The file contains two directories: Meta-inf and Web-inf. In the Web-inf directory, the source code of the application is located with a proper directory structure. If needed additional JAR files can be included in the WAR file and they must be located in the Web-inf/lib directory. Any application created in such a way is ready for deployment. Deployment can be performed in several ways, and in this project Ant tool has been used for WAR file deployment.

6.1.3 Ant

Ant is an open source build tool provided by the Apache Software Foundation. The most typical usage is to build Java applications. It is similar to the GNU utility 'make' that

it replaces, Ant is said to be more portable and simpler to use. Unlike many other build tools, Ant is independent of both the platform and the development environment. The biggest advantage of Ant is its XML oriented design.

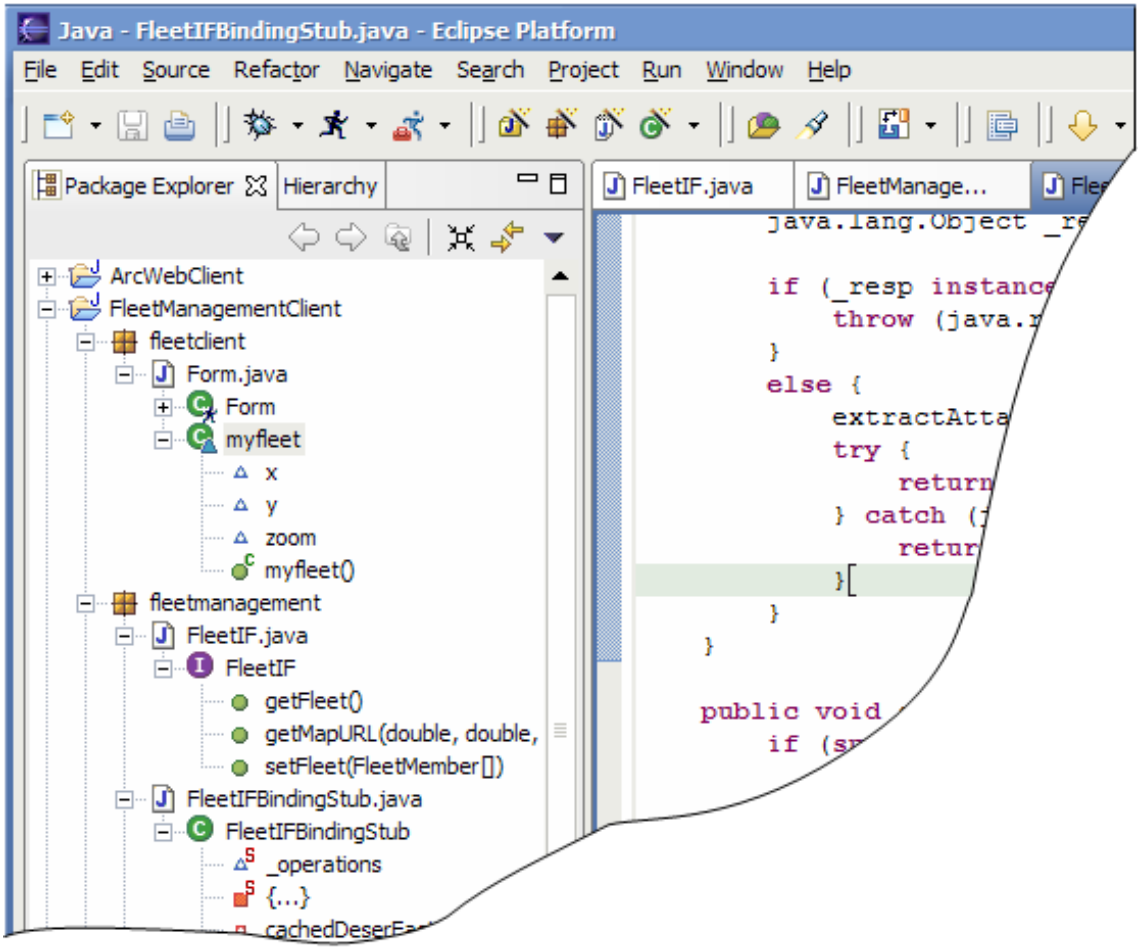


Figure 6.1: Eclipse IDE

6.1.4 Eclipse

The creation of Web Services does not require anything more than a command line prompt. However, unleashing the potential of the Integrated Development Environment (IDE) is highly recommended. There are many IDEs available for Java programmers. The most popular commercial products: JBuilder by Borland, JCreator by Xinox Software, IntelliJ IDEA by JetBrains and some open source projects: NetBeans and Eclipse.
Eclipse was chosen as the IDE tool in this project. Eclipse is not only a Java dedicated platform, it also able to support other languages, custom tools and plug-ins. Generally speaking, the choice was a matter of personal preference rather than made through sophisticated research. It is very difficult to make the right assessment when comes to choosing

among so complicated development environments without having an opportunity of working with all of them. The main factor however was to choose an open-source IDE rather then a commercial product. Eclipse provides facilities for integration with Ant tool, thanks to that all the programming was accomplished using this IDE. The Figure 6.1 gives an insight of the environment.

6.1.5 WSDL2Java - plug-in

The Eclipse environment is fully customizable. There are many plug-ins that allow simplification of the development process. In this project, there was a need to use a command line tool WSDL2Java, which generates classes out of the WSDL file. However, with usage of a dedicated Eclipse plug-in, it was possible to generate classes automatically from any given URL. The classes were then automatically imported to the Eclipse project.

6.2 .NET Tools

Development of the standalone application was done using the .NET platform in order to evaluate the differences between this and the Java platform. The application developed is a standalone MapClient which was mainly used for comparison of interoperability and performance of the development. At first glance, it can be observed that Microsoft put in a big effort to automate the whole development process, and in so doing, simplifying the tasks and reducing the time taken. All the automation, however, does not allow one to get a thorough understanding what is happening behind the scenes. In Section 6.5 the discussion of advantages and features comparison is carried out. In the following section, the basic tools are discussed.

6.2.1 Visual Studio .NET

Visual Studio .NET is a product of Microsoft. It provides plenty of programming tools in one Integrated Development Environment. It has similar features to Eclipse. But, unlike Eclipse, it allows the use of more automated tools during the development process. Moreover, it integrates its Graphical User Interface with many of Microsoft's other products to create a single interface for almost any task related to programming and development of standalone applications, databases, web sites, Web Services and others.

> *Visual Studio .NET is the only development environment built from the ground up to enable integration through XML Web services. By allowing applications to share data over the Internet, XML Web services enable developers to assemble applications from new and existing code, irregardless of the platform, programming language, or object model.*[36]

Visual Studio allows creation of Web Services using several languages including C# and Visual Basic.

6.2.2 IIS

The Internet Information Server is a standard Web Server used under the Windows operating system. It provides web programmers with all the functionalities that can be found in the .NET platform. It provides a means for developing simple web pages, script generated web sites and web applications, including support for Web services.

> *Internet Information Services (IIS) 6.0 is a powerful Web server that provides a highly reliable, manageable, and scalable Web application infrastructure for all versions of Windows Server 2003. IIS helps organizations increase Web site and application availability while lowering system administration costs.*[37]

6.2.3 WSE

Web Services Enhancements (WSE) 2.0

> *Web Services Enhancements for Microsoft .NET (WSE) is a supported add-on to Microsoft Visual Studio .NET and the Microsoft .NET Framework providing developers the latest advanced Web services capabilities to keep pace with the evolving Web services protocol specifications*[32]

WSE 2.0 should simplify the development and deployment of Web services. It should enable the programmers to have easier mean for applying security policies, establishing secure conversations, retrieval and validation of security tokens and more.[32]

6.3 Environment setup

This section will shortly cover the installation procedure of all the applications and tools used in the development. It is worth noticing, that .NET platform installation is almost completely automatic and there is no other alternative but using the tools mentioned. Whereas, tools and applications used with Java technology have to be carefully chosen and configured separately.

6.3.1 Java platform setup

1. The first in the setup is to ensure that the Java SDK platform is properly installed in the system.
2. The next step is checking the correct setup of environment variables (JAVA_HOME and CLASSPATH).
3. After that the Java Web Services Development Pack (JWSDP) has to be installed. It is bundled together with the Tomcat installation and the Ant tool.
4. Setting up all the environment variables is crucial (ANT_HOME, CATALINA_HOME, JWSDP_HOME and modification to CLASSPATH).
5. After that, installation of Eclipse should follow.
6. If needed additional installation of wsdl2java plug-in for Eclipse should be performed.

6.3.2 .NET platform setup

1. The first step is to ensure that the operating system is updated with all the security patches.
2. The next step is installation of Visual Studio .NET, during the setup all the needed components will be added and updated, including .NET framework and IIS server.
3. After that the WSE 2.0 should be installed.
4. Visiting the Windows update website is recommended again.

6.4 Implementation process

This section describes in detail the technical aspects of the implementation process of the complete system.
In order to get the system running, the Location Service first needs to be created. Compiling the Location Service requires the use of the Ant tool. The tool compiles all the Java code, generates the WSDL file, prepares the directories for a WAR file, and creates a deployable package. All the steps are fully customizable and have to be performed manually (for the first time at least). This is done by issuing a command:

```
ant build
```

When this is ready, the next step is to deploy this application to the Tomcat server. This is also performed by the Ant tool, and can be done by issuing a command:

```
ant deploy
```

When application is deployed it is accessible via a URL, which is defined in the *build.properties* configuration file used by the Ant tool. All the steps needed to create a WAR file are defined in the *build.xml* file.
The usage of the Ant tool is integrated into Eclipse, thereby simplifying the whole process slightly.

When the above procedure is successful, a Location Service WSDL file is then available for the Fleet Management Service. By using a wsdl2java command line tool, or an Eclipse wsdl2java plug-in, classes can subsequently be generated and this will be used to communicate with the Location Service.

```
wsdl2java location.wsdl
```

In order to communicate with the Map Service, we need to generate classes out of the Map Service WSDL files. The Map Service exposes two WSDLs that were used in the Fleet Management System. The first is the WSDL describing the real functionality of the Map Service and the second one describes the functionality of the authentication service which grants access to the main service. The generation of classes can be done by issuing these commands:

```
wsdl2java mapimage.wsdl
```

and

```
wsdl2java authentication.wsdl
```

When the source code is ready, the Fleet Management Service is ready to be compiled. The same steps that were used when compiling Location Service have to be performed by using the Ant tool yet again.

```
ant build
```

When this is ready, the next step is to deploy this application to the Tomcat server.

```
ant deploy
```

If everything has performed correctly so far, there will be two deployed services . This can be checked by accessing them via HTTP. These are the URLs of the two services:

```
http://localhost/fleet-rpc/fleet
```

and

```
http://localhost/location-rpc/location
```

The above locations are completely dependent on where the services are actually deployed. The client applications development is relatively simple, it requires standard compilation. However, before proceeding generation of classes out of the Fleet Management WSDL file is required.

```
wsdl2java fleet.wsdl
```

When everything is generated successfully and the source code of the client is ready, the client can be compiled by using a standard Java compiler. The output is a standalone Java application.

The above process is very important and crucial for enabling the complete system. Unfortunately the importance comes in pair with complexity. A lot of care while performing above steps must be taken. It was experienced that some of the compilation errors were very confusing. Moreover, unreported compilation errors of one service may result in runtime errors in the client applications. Any change made to the source code must be performed with particular care and patience in order for the final compilation to be successful and error free.
It is highly recommended to monitor Tomcat log files during the deployment and usage of WAR files, in order to quickly detect erroneous situations.

Implementation process using .NET was significantly simpler for two reasons. The application developed was a standalone client only. The second reason is, that in general programming using Visual Studio is automated and does not require much of configuration and prior implementation work. Generation of proxy classes from the WSDL file was performed with a dedicated wizard. The last step for getting the application running was issuing the build command.

6.5 Discussion

There are several interesting aspects worth a throughout discussion, that were encountered during the work on this project. The following section focuses on these.

6.5.1 Performance

Web Services is the slowest middleware technology available. This is due to the overhead that is introduced by using XML. The parsing of huge XML documents is very inefficient especially when the amount of data needed to be transmitted is relatively small compared to the XML data. It is also a problem when it comes to transmitting binary data encoded with base64. It has already become apparent that performance is an important issue for the designers of Web Services[21]. This must be improved on, in order to convince potential users to make use of this technology. It is known that in projects where performance is the crucial factor, Web Services cannot be used. When considering this particular implementation, the speed of applications and services is satisfactory. Especially operations designed internally in the system work well. However, it has not been tested, whether significantly increasing the number clients might influence this. The most visible waiting time occurs during the map retrieval. This is not dependent on the system design, but on the third party map provider. The reason for this is only related to the connectivity between the Fleet Management Client and the Map Service, due to the fact that the map data is already transfered via HTTP. Thus, there is no overhead related to XML parsing. Choosing another provider or improving the connection might solve this particular problem. However, it has nothing to do with Web Services on the implemented side of the system.

6.5.2 Security

Web Services currently support several popular security features for ensuring the build of a secure solution. However, SOAP - the primary Web Services protocol did not define any security model (at the start of this project). Encryption, authorization and authentication methods had to be ensured by usage of secure transport protocols. While working on this thesis, an extension to the SOAP protocol was introduced, which allows the definition of a security model for the Web Services to be made. There was, regrettably, not enough time to take advantage of this new specification. A probable next step to improving the security, would be the implementation of these particular features. Nonetheless, the security of the system can be addressed by the use of secure transport protocol e.g. HTTPS.
The SOAP extension now allows several new security features including:

- Signing and/or verification of the SOAP messages.
- Encryption and/or decryption of the SOAP messages.
- Sending UserName tokens and X509 certificate tokens inside the messages.

The introduced functionalities were not fully supported. This is the official statement:

"... the Java standards for some of these security technologies are undergoing definition under the Java Community Process"[33]

These security solution *"... are subject to change with new revisions of the technology. As standards are defined in the Web Services Security space, we will be moving toward using the appropriate standard APIs"*[33]

"JSR 105 is a standard API (in progress, almost at Proposed Final Draft) for generating and validating XML Signatures as specified by the W3C recommendation"[33]

From the above description, it can be concluded that the Web Services will soon satisfy most of the security requirements. This will allow Web Services to operate in domains exhibiting the highest threats e.g. online shopping or internet banking.

6.5.3 Transactions and parallelism

Transactions in Web Services are still a matter of releasing an adequate WS-specification. Currently there is no definition of how to deal with this issue. In the adopted project this will be a concern especially when dealing with widely distributed multiple clients working on the system. No problems where encountered during the testing of the running components. However, it does not indicate that potential problems might not occur.
On the other hand, multi-threading and parallelism of implemented components was taken into account. Modeling of the system objects was performed with particular attention paid on implementation of proper thread safety of sensitive data.

6.5.4 Platform comparison

Web Services are available for all software platforms, this was their creators goal. Despite that, there are several major vendors, that focus on providing the most attractive development solutions. Among them: Microsoft Corporation with .NET and two systems based on Java provided one by IBM Corporation and the second one by Sun Microsystems. In this project two of the available platforms have been examined closer (Sun and Microsoft), with particular stress on Java. From the encountered problems, gained experience and obtained results, it can be stated that both platforms provide tools for successful implementation of Web Services, nonetheless with particular differences. Microsoft tends to simplify the development process with introduction of automated tools, whereas Sun gives more customizable and complicated instruments which in the beginning may cause problems for newcomers.
It is well-known, that combination of Apache and Tomcat gives a more efficient solution than the one offered by IIS with respect to performance. Though, small companies willing to have fast and easy to use tools may move their attention to Microsoft products, for their user-friendliness. System's vulnerability to potential attacks may play also an important factor. The threats related to open-source software are significantly lower. Nonetheless,

the future of Web Services lies probably in the center, this is due to the interoperability of the technology that will allow integration of both platforms.

6.5.5 Possible improvements

As was mentioned earlier, the system developed in this project does not intend to provide a complete solution for Fleet Management. What it really aims to do, is to provide a complete platform with the essential issues solved, in particular those related to Web Services. However, it is worth mentioning some possibilities of enhancements to the system with features not related to Web Services technology.
The system is open to many improvements depending on the needs and the requirements of the final customer. The goal of the design is to provide a structure that will not require radical changes in the case that additional components are introduced. Here are some obvious improvements that may be necessary to add, before the system is run in a real-life environment. Currently, there is no dedicated database support. It was not introduced because, most companies that would be interested in applying this solution would probably already manage their own databases. In other words, integration with the existing database system would be a requirement. Such an integration process must be performed, taking into account the individual needs of the given company and the specification of their system.
Another aspect of the Fleet Management System is the interface to the mobile operator. The mobile operator is the provider of the location information based on positioning of the mobile phone or any similar device. It is very likely that the mobile operator will not provide the location information for free. If the required amount of money per *"location"* would be significant, their will be no possibility to have *fresh* data arriving when needed. Considering that, the payment for such a service would be similar to rates per *SMS* today, there would be a need for a company to enable a automatic control over the number of location requests. However, if prices drop continues, it will open opportunities for new improvements and additions, e.g. location scheduling.
Location scheduling might work like this: *"the fleet member"* would have a defined *"rule"* which allows him to be localized in some specified period of time, e.g. every minute during a fleet member's working time. The history of these events would be stored in a database. Moreover the definition of the mentioned *"rules"* would be located in the Fleet Management Service database in order to be accessible by any application client of the system.
The obvious improvement would be enabling access control to the system, which would limit access to location information and fleet member data. The access level could differ. For example, several types of users could be introduced: administrator, manager, fleet member. Each of these users would have different privileges in the system, for example:

- System administration access would be the broadest, allowing modification of all data, right to create new users, changing their access, etc.

- The manager of the fleet would have a possibility to modify data related to the fleet of his own. Each fleet in reality would represent a single company that has access

to the system. From the interface point of view there would not be many changes, however the Fleet Management Service architecture would expand significantly.

- Fleet member access rights would be the most limited and defined with respect to a particular requirements of the system customer

A proper authentication to the system is also a must. The logging facility of the events occurring in the system would also be recommended. In this scenario, particular features and user friendliness of the client application interface was not considered crucial. Probably more customizable and richer in features interface would be more appealing to the final customer. Nonetheless the interface provides all the means for interaction with the designed Fleet Management service. The architecture of the system allows integration with any additional components if needed. Web Services provide a flexible tool to add new graphical interfaces to the system. Taking care of particular customers needs, it would be a great idea to develop additional clients that could interact with the system. For example web-based clients that don't require any particular care on the client side, the only needed thing is a web browser.
The portability of Web Services gives opportunities to extend its scope to the mobile application development. It would allow to provide the same functionalities on the mobile phone or any portable device with support of Web Services.
All of the above features can be implemented in order to satisfy the most demanding customers' needs. Web Services technology used in the system ensures ease of integration with any additional components.
There are certain legal limitations of localizing individuals. In most countries, people have a right to privacy and to choose whether they want to be localized or not. When this involves company employees and localization due to work related activities, the procedure is different. Workers have to be informed about the fact of being tracked, and this will be done through mobile device which they will have to carry with them. These particular characteristics can also influence the final form of the system.

Chapter 7

Conclusions

At the present time, there are many different solutions for the integration of distributed systems. However, from the earlier analysis and implementation, it can be concluded that the Web Services offer a tool capable of solving complex integration tasks.
Web Services are constantly being improved upon and are currently supported by many implementation tools that simplify and reduce time needed for development.
A high level of abstraction allows integration of modern designs with virtually any existing legacy system.
Although, there are certain obstacles; in particular when trying to get familiar with the technology. The technology itself is not yet fully defined and this in turn has resulted in the existing standards being constantly updated and improved on. Tools that are available do not always follow the actual specifications available. This is due to the fact that the technology is constantly changing. This can be misleading when dealing with Web Services for the first time.
The number of commercial articles on Web services is overwhelming but most do not really go into great depth. To follow the changes happening in the technology is a very difficult task, especially when new specification are being approved all the time. A membership in one of the decision making bodies would be recommended to be always up to date with recent modifications.
On the other hand, there are several reasons that make Web Services very attractive in modern inter-operable solutions.
The defined standards, created in open community with contributions from many IT leaders (e.g. Microsoft, IBM, Sun) are certainly an obvious advantage.
The standards available makes it possible to develop applications quickly and easily. Two very useful standards are XML and HTTP which can be used to provide flexible means of data exchange. The simplification of this approach, unfortunately, limits some characteristics that may be crucial in specific applications. Web Services (due to the overhead related to usage of XML) are not the best choice when high performance is called for. Moreover, the reliability of the third-party service may not always be guaranteed. Some features are still missing that may be essential in certain applications.
Surprisingly, the complexity starts to be more apparent; and without automated tools, the development of complex solutions may become tedious. This is especially so when it comes

to debugging of distributed applications. Only if development work moves forward, will it be possible to see if the outstanding issues are solved. Nonetheless, with major vendors supporting this technology, Web Services seem destined to succeed and ultimately alter the shape of the Internet.

Bibliography

[1] Scott Short "Building XML Web Services for the Microsoft .NET platform"
Copyright 2002 Microsoft Press

[2] Robert Tabor "Microsoft .NET XML Web Services"
Copyright 2002 Sams

[3] Robin Sharp "The Poor Man's Guide to Computer Networks and their Applications"
IMM DTU

[4] David A. Chappell "Tyler Jewell - Java Web Services"
Copyright 2002 O'Reilly & Associates

[5] Adam Freeman, Allen Jones, Adam Freeman
"Microsoft .NET XML Web Services Step by Step"
Copyright 2002 Microsoft Press

[6] "Java Web Services Developer Pack (Java WSDP)", Sun Microsystems
URL: http://java.sun.com/webservices/jwsdp/index.jsp

[7] "New to SOA and Web services", IBM Corporation
URL: http://www-106.ibm.com/developerworks/webservices/newto

[8] "W3C WSDL 1.1 Specification", World Wide Web Consortium
URL: http://www.w3.org/TR/wsdl

[9] "W3C SOAP 1.2 Specification", World Wide Web Consortium
URL: http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[10] "OASIS UDDI 3.0 Specification", OASIS
URL: http://uddi.org/pubs/uddi_v3.htm

[11] "WS-Security AppNotes", IBM Corporation and Microsoft Corporation
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/
dnglobspec/html/ws-security-appnote.asp

[12] "The website of the Eclipse Foundation", Eclipse
URL: http://www.eclipse.org

[13] Maydene Fisher - "Introduction to Web Services", Sun Microsystems
URL: http://java.sun.com/webservices/docs/1.0/tutorial/doc/IntroWS.html

[14] "Esri Homepage", Esri
URL: http://www.esri.com

[15] "ArcWeb Services specification", Esri
URL: http://arcweb.esri.com/arcwebonline/index.htm

[16] "Extensible Markup Language (XML)", W3C
URL: http://www.w3c.org/XML

[17] "Web Services Activity", W3C
URL: http://www.w3c.org/2002/ws/

[18] XML Protocol Working Group, W3C
URL: http://www.w3.org/2000/xp/Group

[19] Web Services Security (WS-Security), Microsoft Corporation
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/
dnglobspec/html/ws-security.asp

[20] Mark Colan, SOA and Web Services
URL: http://www-106.ibm.com/developerworks/webservices

[21] Paul Sandoz, Santiago Pericas-Geertsen, Kohuske Kawaguchi, Marc Hadley,
and Eduardo Pelegri-Llopart - "Fast Web Services", Sun Microsystems
URL: http://java.sun.com/developer/technicalArticles/WebServices/fastWS/index.html

[22] Claire Rogers - "Developing Web Services with Eclipse and Open Source",
February, 2004
URL: http://www.eclipsecon.org/
EclipseCon_2004_TechnicalTrackPresentations/30_Rogers.pdf

[23] Eric M. Burke - "Top 15 Ant Best Practices", 17th of October, 2003
URL: http://www.onjava.com/pub/a/onjava/2003/12/17/ant_bestpractices.html

[24] Massimiliano Bigatti - "Web Services Integration Patterns, Part 1",
16th of June, 2004
URL: http://webservices.xml.com/pub/a/ws/2004/06/16/patterns.html

[25] Massimiliano Bigatti - "Web Services Integration Patterns, Part 2",
30th of June, 2004
URL: http://webservices.xml.com/pub/a/ws/2004/06/30/patterns.html

[26] "Web Services and Service-Oriented Architectures"
URL: http://www.service-architecture.com

[27] "CORBA BASICS at OMG"
URL: http://www.omg.org/gettingstarted/corbafaq.htm

[28] "Web Services Specifications, Microsoft Corporation"
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/
dnglobspec/html/wsmsgspecindex.asp

[29] "Servlets Specifications, Sun Microsystems"
URL: http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html

[30] "Tomcat, Apache Software Foundation"
URL: http://jakarta.apache.org/tomcat/

[31] "Add Web Reference for Eclipse"
URL: http://wsdl2javawizard.sourceforge.net

[32] "Web Services Enhancements (WSE), Microsoft Corporation"
URL: http://msdn.microsoft.com/webservices/building/wse/default.aspx

[33] "Web Services Security"
URL: http://docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-soap-message-security-1.0.pdf

[34] "XML Signature WG"
URL: http://www.w3.org/Signature/

[35] "XML Encryption Syntax and Processing"
URL: http://www.w3.org/TR/xmlenc-core/

[36] "Visual Studio .NET, Microsoft Corporation"
URL: http://msdn.microsoft.com/vstudio/productinfo/overview/default.aspx

[37] "Internet Information Service, Microsoft Corporation"
URL: http://www.microsoft.com/WindowsServer2003/iis/default.mspx

[38] "Case studies, Danske Bank, Microsoft Corporation"
URL: http://www.microsoft.com/resources/casestudies/
casestudy.asp?casestudyid=13756

[39] "Case Study, Winterthur, CIO Magazine"
URL: http://www.cio.com/archive/030103/et_article.html

[40] "Case study, *A recognized technology leader*, IBM Corporation"
URL: http://www-306.ibm.com/software/success/cssdb.nsf/CS/KHAL-62EUHP?OpenDocument&Site=software

[41] "Case study, Amazon.com"
URL: http://www.amazon.com/gp/browse.html/002-3171961-4242449?node=3435361

[42] "The Not So Short Introduction to LaTeX 2_ε"
URL: http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf

List of acronyms

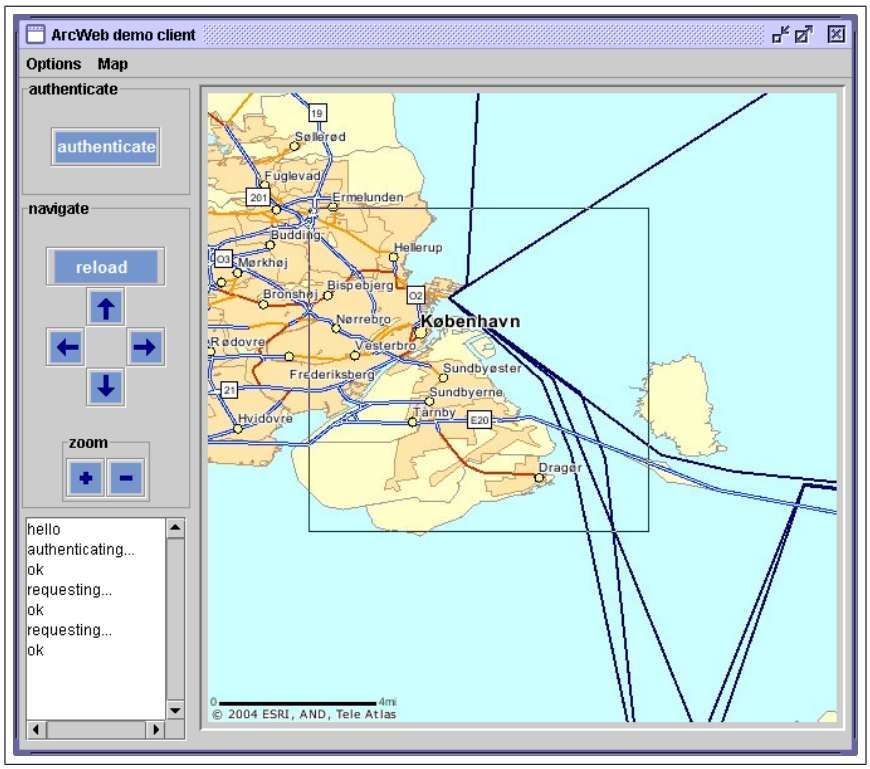
SMTP - Simple Mail Transfer Protocol
HTTP - Hyper Text Transfer Protocol
TCP - Transmission Control Protocol
IP - Internet Protocol
XML - eXtensible Markup Language
WSDL - Web Service Definition Language
UDDI - Universal Description, Discovery and Integration
SOAP - Simple Object Access Protocol
RPC - Remote Procedure Call
DCOM - Distributed Component Object Model
RMI - Remote Method Invocation
CORBA - Common Object Request Broker Architecture
OMG - Object Management Group
ORB - Object Request Broker
IIOP - Internet Inter-ORB Protocol
WS - Web Service
WWW - World Wide Web
W3C - World Wide Web Consortium
SOA - Service Oriented Architecture
API - Application Programming Interface
JAX - Java API for XML
JAX-RPC - Java API for XML-Based RPC
SAAJ - SOAP with Attachments API for Java
JAXP - Java API for XML Processing
DOM - Document Object Model
SAX - Simple API for XML
XSLT - Extensible Stylesheet Language Transformations
JAXB - Java API for XML Binding
JAXR - Java API for XML Registries
WSDP - Web Services Developer Pack
JWSDP - Java Web Services Developer Pack
OASIS - Organization for the Advancement of Structured Information Standards
GUI - Graphical User Interface
WAR - Web ARchive format
JAR - Java Archive

IDE - Integrated Development Environment

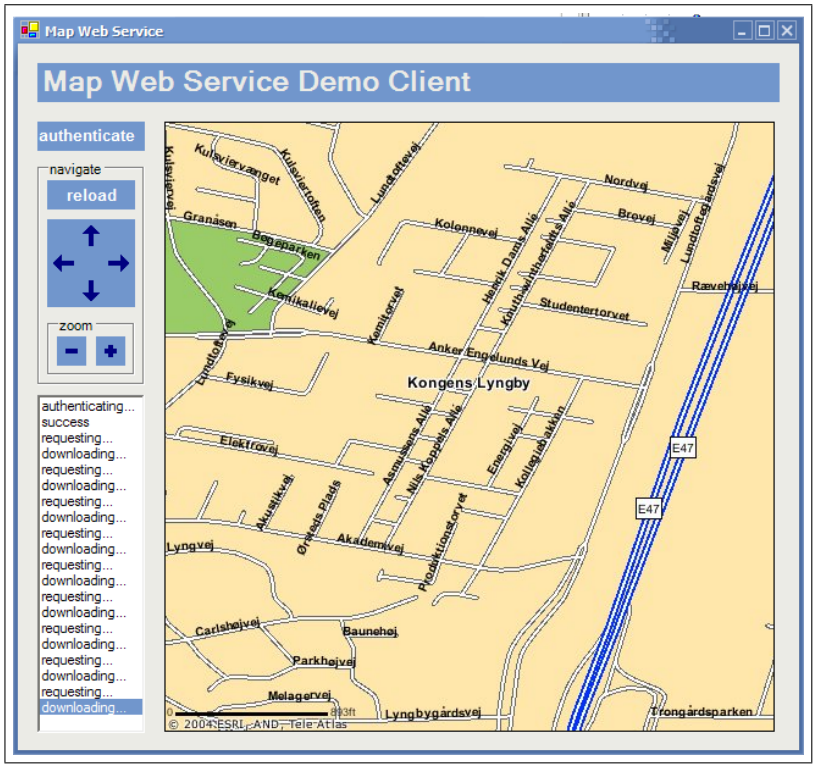
Appendix A

Application screenshots

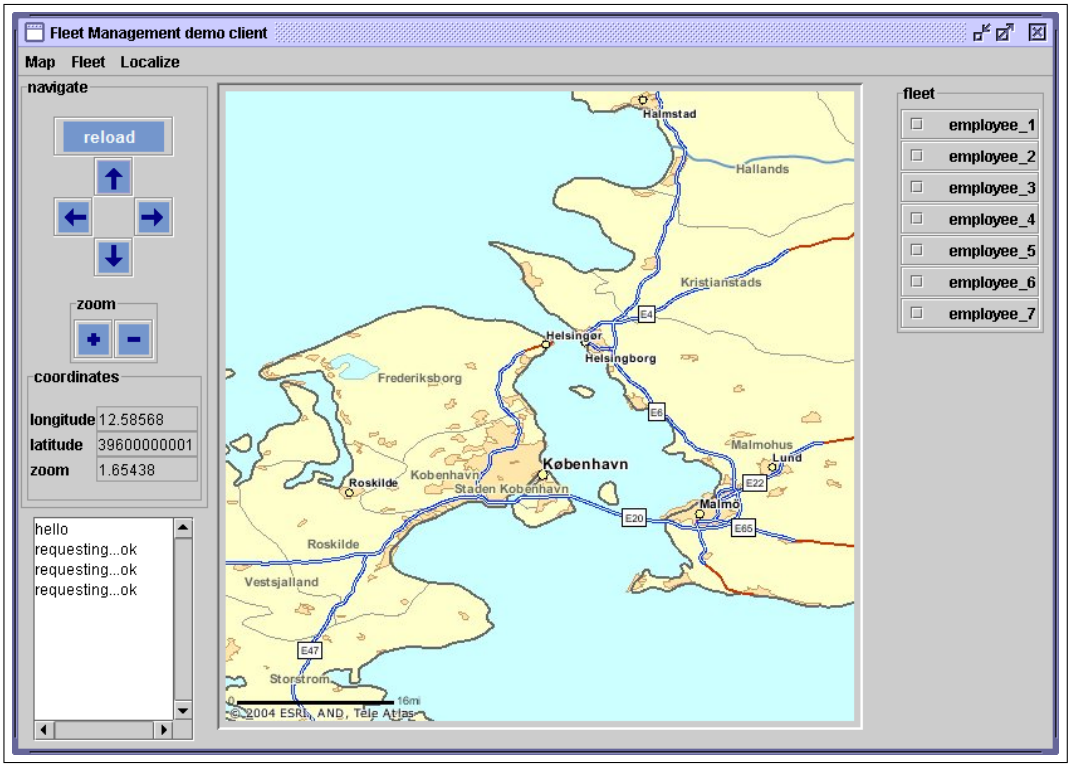
A.1 ArcWeb Client

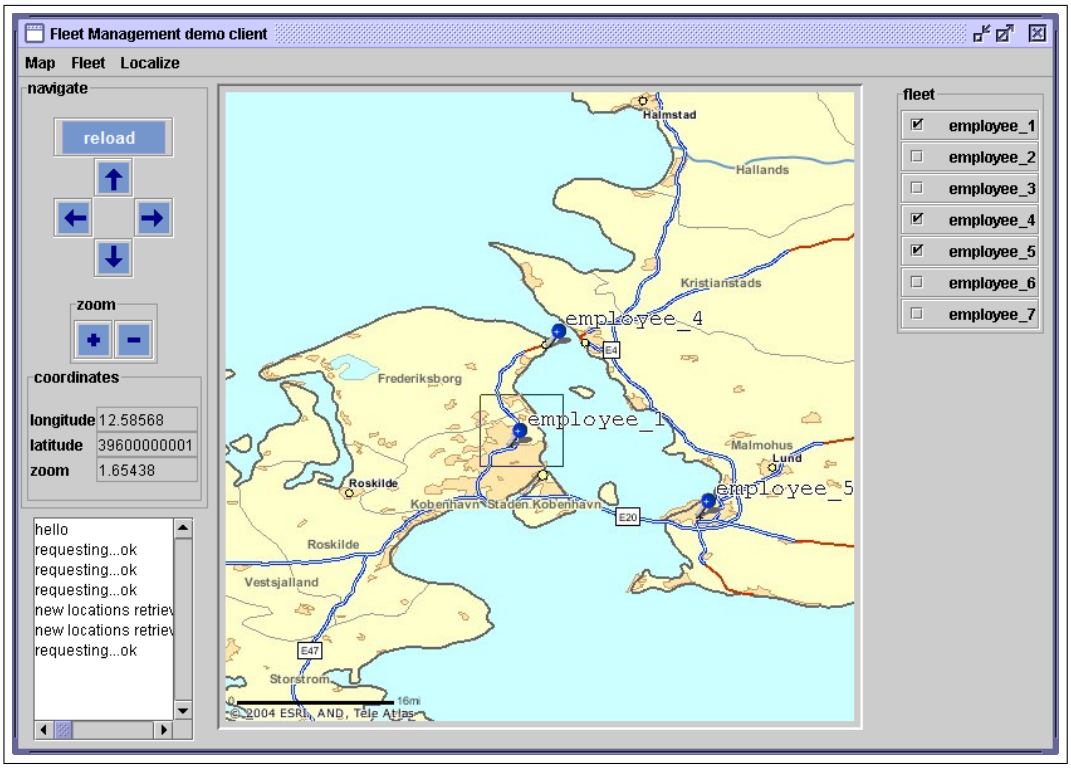


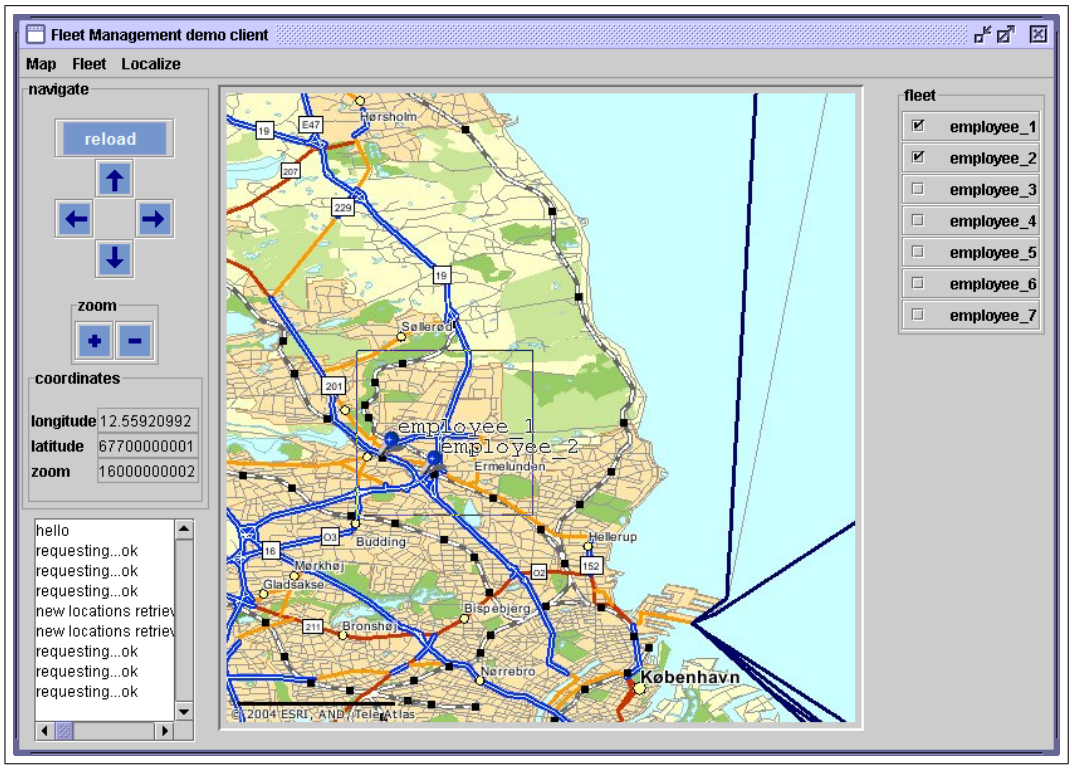
A.2 ArcWeb .NET Client

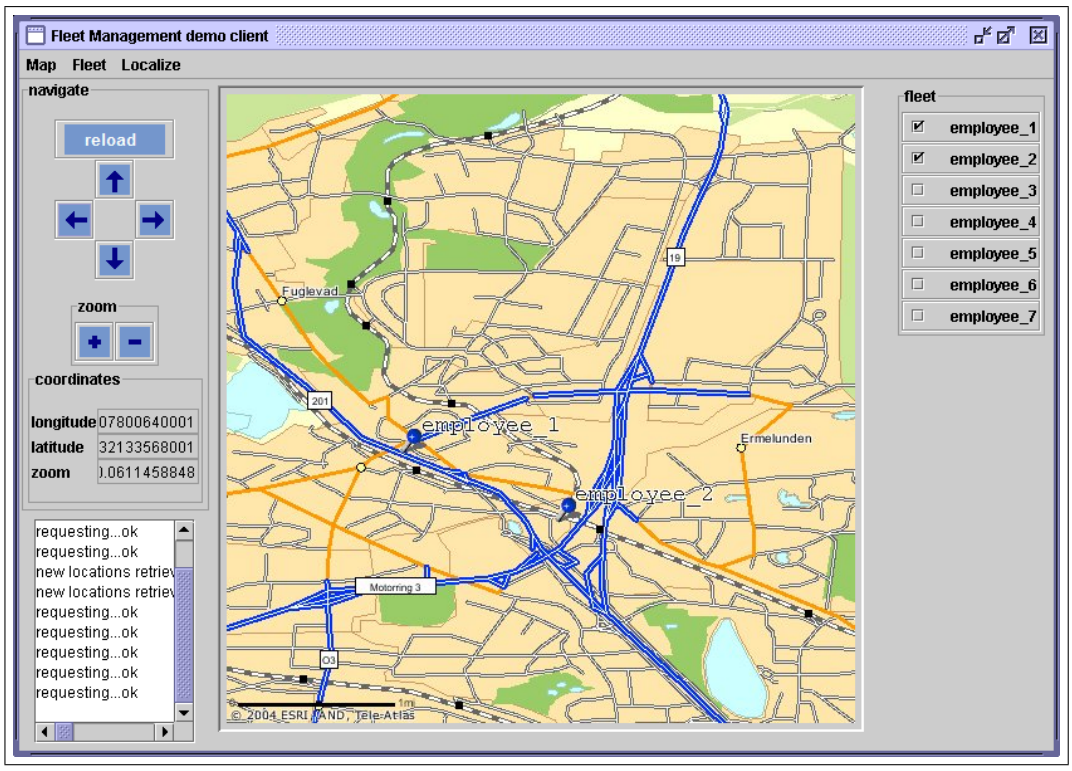


A.3 Fleet Management Java Client






Fleet Management demo client
Map Fleet Localize
navigate
reload
zoom
coordinates
longitude 12.55920992
latitude 67700000001
zoom 16000000002
hello
requesting...ok
requesting...ok
requesting...ok
new locations retriev
new locations retriev
requesting...ok
requesting...ok
requesting...ok
fleet
employee_1
employee_2
employee_3
employee_4
employee_5
employee_6
employee_7
employee_1
employee_2
Hørsholm
Sellerød
Ermelunden
Budding
Mørkhøj
Gladsaxe
Bronshøj
Bispebjerg
Nørrebro
Hellerup
København
© 2004 ESRI, AND, Tele Atlas


Fleet Management demo client
Map Fleet Localize
navigate
reload
zoom
coordinates
longitude 07800640001
latitude 32133568001
zoom 0611458848
requesting...ok
requesting...ok
new locations retriev
new locations retriev
requesting...ok
requesting...ok
requesting...ok
requesting...ok
requesting...ok
fleet
employee_1
employee_2
employee_3
employee_4
employee_5
employee_6
employee_7
employee_1
employee_2
Fuglevad
Ermelunden
Motorring 3
© 2004 ESRI, AND, Tele Atlas

Appendix B

Fleet Management Client

B.1 Form.java

```
package fleetclient;

import java.awt.Color;
import java.awt.Cursor;
import java.awt.Graphics;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.KeyEvent;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.rmi.RemoteException;

import javax.swing.BorderFactory;
import javax.swing.BoxLayout;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JCheckBoxMenuItem;
import javax.swing.JFrame;
```

```
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import javax.swing.JRadioButtonMenuItem;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;

import fleetmanagement.FleetIF;
import fleetmanagement.FleetIFBindingStub;
import fleetmanagement.FleetMember;
import fleetmanagement.Location;

class myfleet extends FleetIFBindingStub implements FleetIF {
  double x=1, y=1, zoom=1;
  public myfleet() throws java.net.MalformedURLException,
     RemoteException {
    super(new URL("http://localhost:8080/fleet-jaxrpc/fleet"),
       null);
    x = 9.5;
    y = 52;
    zoom = 39;
  }
}


public class Form implements ActionListener, MouseListener,
   MouseMotionListener, ItemListener {
  static JTextArea textArea = new JTextArea();

  private static void createAndShowGUI() {
    try {
      client = new myfleet();
    }
    catch (Exception e) {
      textArea.append(e.getMessage());
      e.printStackTrace();
    }
    JFrame.setDefaultLookAndFeelDecorated(true);
        Form foremka = new Form();
        JFrame mapFrame = new JFrame("Fleet Management demo
            client");
        mapFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        mapFrame.setContentPane(foremka.mainPanel);
        mapFrame.setJMenuBar(foremka.mainMenu);
        mapFrame.setLocation(300,200);
        mapFrame.pack();
        mapFrame.setVisible(true);
    }

    private static ImageIcon createImageIcon(String path) {
      java.net.URL imageURL = Form.class.getResource(path);
      if (imageURL == null) {
          System.err.println("Resource not found: " + path);
          return null;
      }
      else {
          return new ImageIcon(imageURL);
      }
    }

    public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }

    static myfleet client = null;
    FleetMember[] members=null;
    JCheckBoxMenuItem checkboxes[] = {};
    JScrollPane areaScrollPane;

    JButton down = new JButton(createImageIcon("/images/
       arrowdown.gif"));
  int height=500;
    JLabel l = new JLabel();
    JButton left = new JButton(createImageIcon("/images/
       arrowleft.gif"));
    JMenuBar mainMenu;
    JPanel mainPanel,leftPanel;
    JButton minus= new JButton(createImageIcon("/images/minus.
       gif"));

    JPanel navigatePanel, mapPanel, listPanel;//,
       authenticatePanel;
    JButton plus = new JButton(createImageIcon("/images/plus.gif
```

```
        "));

    JButton reload = new JButton(createImageIcon("/images/reload
        .gif"));
    JButton right = new JButton(createImageIcon("/images/
        arrowright.gif"));
    JScrollPane scrollPane = null;

  int startX, startY, lastX, lastY;
    JButton up = new JButton(createImageIcon("/images/arrowup.
        gif"));

    JTextField lonT = new JTextField();
    JTextField latT = new JTextField();
    JTextField zoomT = new JTextField();

  int width=500;

    public Form(){

        navigatePanel = createNavigatePanel();
        listPanel = createListPanel();
        mapPanel = createMapPanel("/images/map.gif");

        leftPanel = new JPanel();
        leftPanel.setLayout(new BoxLayout(leftPanel, BoxLayout.
            PAGE_AXIS));
        leftPanel.setBorder(BorderFactory.createEmptyBorder
            (0,0,0,0));

        leftPanel.add(navigatePanel);
        leftPanel.add(listPanel);
        JPanel fleetPanel = createFleetPanel();

        mainPanel = new JPanel();
        mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.
            LINE_AXIS));
        mainPanel.setBorder(BorderFactory.createEmptyBorder
            (0,0,0,0));
        mainPanel.add(leftPanel);
        mainPanel.add(mapPanel);
        mainPanel.add(fleetPanel);

        reload.setActionCommand("reload");
        up.setActionCommand("up");
```

```
        down.setActionCommand("down");
        left.setActionCommand("left");
        right.setActionCommand("right");
        plus.setActionCommand("plus");
        minus.setActionCommand("minus");
        textArea.setEditable(false);
        mainMenu = createMenu();
        updateLabels(client.x,client.y,client.zoom);
    }

    public void actionPerformed(ActionEvent event) {
      try {
          if(event.getActionCommand().equals("reload")) {
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("plus")) {
            client.zoom = client.zoom / 2;
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("minus")) {
            client.zoom = client.zoom * 2;
        reload_map(client.x,client.y,client.zoom);
      }
          else if(event.getActionCommand().equals("up")) {
            client.y += 0.2 * client.zoom;
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("down")) {
            client.y -= 0.2 * client.zoom;
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("left")) {
            client.x -= 0.2*client.zoom;
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("right")) {
            client.x += 0.2*client.zoom;
        reload_map(client.x,client.y,client.zoom);
          }
          else if(event.getActionCommand().equals("lyngby")) {
            client.x = 12.51;
        client.y = 55.78;
        client.zoom = 0.05;
        reload_map(client.x,client.y,client.zoom);
          }
```

```
      else if(event.getActionCommand().equals("europe")) {
    client.x = 9.5;
    client.y = 52;
    client.zoom = 39;
    reload_map(client.x,client.y,client.zoom);
      }
      else if(event.getActionCommand().equals("sjaelland"))
         {
    client.x = 11.8;
    client.y = 55.5;
    client.zoom = 2.32986;
    reload_map(client.x,client.y,client.zoom);
      }
      else if(event.getActionCommand().equals("kobenhavn"))
         {
    client.x = 12.5;
    client.y = 55.7;
    client.zoom = 0.35;
    reload_map(client.x,client.y,client.zoom);
      }
      else if(event.getActionCommand().equals("oresund")) {
        client.x = 12.5;
    client.y = 55.7;
    client.zoom = 0.5;
    reload_map(client.x,client.y,client.zoom);
      }
      else if(event.getActionCommand().equals("selected")) {
        for(int i=0;i<members.length;i++){
          if(members[i].isVisible()){

            Location a= client.localize(members[i]);
            members[i].setX(a.getX());
        members[i].setY(a.getY());
          }
          textArea.append("new locations retrieved");
        }
        reload_map(client.x,client.y,client.zoom);
      }
      else{
        textArea.append("clicked!\n");
      }
}
catch (Exception e) {
textArea.append(e.getMessage());
e.printStackTrace();
```

```
    }
        textArea.repaint();
    }

    public void itemStateChanged(ItemEvent e) {
        Object source = e.getItemSelectable();
        for(int i=0;i<checkboxes.length;i++)
        if(source.equals(checkboxes[i])) {
          //textArea.append(members[i].getName()+"\n");
          members[i].setVisible(!members[i].isVisible());
        }
    }

    private void adjustBorders(JButton but){
    but.setHorizontalAlignment(JButton.CENTER);
    but.setVerticalAlignment(JButton.CENTER);
    but.setVerticalTextPosition(JButton.CENTER);
    but.setHorizontalTextPosition(JButton.CENTER);
    but.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createEtchedBorder(),
      BorderFactory.createEmptyBorder(2,2,2,2)));

    but.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createEmptyBorder(0,0,0,0),
      but.getBorder()));
    }

    public JPanel createFleetPanel(){
      JPanel a = new JPanel();
      a.setLayout(new BoxLayout(a, BoxLayout.Y_AXIS));
      a.setBorder(BorderFactory.createEmptyBorder(0,0,0,0));
      a.setBorder(BorderFactory.createCompoundBorder(
          BorderFactory.createTitledBorder("fleet"),
        BorderFactory.createEmptyBorder(0,0,0,0)));
      GridBagConstraints c = new GridBagConstraints();

    try {
      members = client.getFleet();
      checkboxes = new JCheckBoxMenuItem[members.length];
      for (int i=0;i<members.length;i++){
        checkboxes[i] = new JCheckBoxMenuItem(members[i].getName
            ());
        adjustBorders(checkboxes[i]);
        checkboxes[i].addItemListener(this);
        c.fill = GridBagConstraints.HORIZONTAL;
```

```
        c.gridx = 0;
        c.gridy = i;
        c.gridwidth = 1;
        a.add(checkboxes[i]);
      }
    }
    catch (RemoteException e) {
      textArea.append("unable to retrieve the fleet\n");
        textArea.append(e.getMessage());
        e.printStackTrace();
    }
    JPanel emptyPanel = new JPanel();
    emptyPanel.setSize(200,200);

    JPanel fleet = new JPanel();

    fleet.add(emptyPanel);
    fleet.add(a);
    return fleet;
    }

  private void adjustBorders(JCheckBoxMenuItem but) {
    but.setHorizontalAlignment(JButton.CENTER);
    but.setVerticalAlignment(JButton.CENTER);
    but.setVerticalTextPosition(JButton.CENTER);
    but.setHorizontalTextPosition(JButton.CENTER);
    but.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createEtchedBorder(),
      BorderFactory.createEmptyBorder(2,2,2,2)));

    but.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createEmptyBorder(0,0,0,0),
      but.getBorder()));
  }

  public JPanel createListPanel(){
    JPanel a = new JPanel();
    textArea = new JTextArea(10, 10);
    textArea.setEditable(false);
    scrollPane = new JScrollPane(
        textArea,
        JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
        JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS
    );
    GridBagConstraints c = new GridBagConstraints();
```

```
    c.gridwidth = GridBagConstraints.REMAINDER;
    c.fill = GridBagConstraints.HORIZONTAL;
    c.fill = GridBagConstraints.BOTH;
    c.weightx = 1.0;
    c.weighty = 1.0;
    a.add(scrollPane, c);
    textArea.append("hello\n");
    return a;
  }

  public JPanel createMapPanel(String icon){
    JPanel a = new JPanel();
    l.setHorizontalAlignment(JLabel.CENTER);
    l.setVerticalAlignment(JLabel.CENTER);
    l.setVerticalTextPosition(JLabel.CENTER);
    l.setHorizontalTextPosition(JLabel.CENTER);
    l.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createLoweredBevelBorder(),
      BorderFactory.createEmptyBorder(5,5,5,5)));

    l.setBorder(BorderFactory.createCompoundBorder(
      BorderFactory.createEmptyBorder(0,0,10,0),
      l.getBorder()));
    l.setIcon(createImageIcon(icon));
    l.setText("");
    a.add(l);
    return a;
  }

  public JMenuBar createMenu(){
    JMenuBar menuBar;
    JMenu menu, submenu;
    JMenuItem menuItem;
    JRadioButtonMenuItem rbMenuItem;
    JCheckBoxMenuItem cbMenuItem;

    menuBar = new JMenuBar();

//-----------------------------------------

    menu = new JMenu("Map");
    menu.setMnemonic(KeyEvent.VK_N);
    menu.getAccessibleContext().setAccessibleDescription("");

    submenu = new JMenu("go to:");
```

```
        submenu.setMnemonic(KeyEvent.VK_S);

        menuItem = new JMenuItem("Kobenhavn");
        menuItem.setActionCommand("kobenhavn");
        menuItem.addActionListener(this);
        submenu.add(menuItem);

        menuItem = new JMenuItem("Lyngby");
        menuItem.setActionCommand("lyngby");
        menuItem.addActionListener(this); submenu.add(menuItem);

        menuItem = new JMenuItem("Europe");
        menuItem.setActionCommand("europe");
        menuItem.addActionListener(this); submenu.add(menuItem);

        menuItem = new JMenuItem("Sjaelland");
        menuItem.setActionCommand("sjaelland");
        menuItem.addActionListener(this);
        submenu.add(menuItem);

        menuItem = new JMenuItem("Oresund");
        menuItem.setActionCommand("oresund");
        menuItem.addActionListener(this);
        submenu.add(menuItem);
        menu.add(submenu);
        menuBar.add(menu);

//-------------------------------------------------
        menu = new JMenu("Fleet");
        menu.setMnemonic(KeyEvent.VK_N);
        menu.getAccessibleContext().setAccessibleDescription("");

        menuItem = new JMenuItem("addMember");
        menuItem.setActionCommand("addMember");
        menuItem.addActionListener(this);
        menu.add(menuItem);

        menuItem = new JMenuItem("deleteMember");
        menuItem.setActionCommand("deleteMember");
        menuItem.addActionListener(this);
        menu.add(menuItem);

        menuBar.add(menu);
//-------------------------------------------------
        menu = new JMenu("Localize");
```

```
        menu.setMnemonic(KeyEvent.VK_N);
        menu.getAccessibleContext().setAccessibleDescription("");

        menuItem = new JMenuItem("selected");
        menuItem.setActionCommand("selected");
        menuItem.addActionListener(this);
        menu.add(menuItem);

        menuBar.add(menu);
//------------------------------------------

        menuBar.add(menu);

        return menuBar;
        }

    public JPanel createNavigatePanel(){
        JPanel a = new JPanel();

        a.setLayout(new BoxLayout(a, BoxLayout.Y_AXIS));

        adjustBorders(reload);
        adjustBorders(up);
        adjustBorders(down);
        adjustBorders(left);
        adjustBorders(right);
        adjustBorders(plus);
        adjustBorders(minus);

        JPanel arrows = new JPanel();
        arrows.setLayout(new GridBagLayout());

        GridBagConstraints c = new GridBagConstraints();
        c.fill = GridBagConstraints.HORIZONTAL;

        c.gridx = 0;
        c.gridy = 0;
        c.gridwidth = 3;
        arrows.add(reload,c);

        c.gridwidth = 1;
        c.gridx = 1;
        c.gridy = 1;
        arrows.add(up,c);
```

```
c.gridx = 1;
c.gridy = 4;
arrows.add(down,c);

c.gridx = 0;
c.gridy = 3;
arrows.add(left,c);

c.gridx = 2;
c.gridy = 3;
arrows.add(right,c);
a.add(arrows);

JPanel zoom = new JPanel();
zoom.setLayout(new BoxLayout(zoom, BoxLayout.LINE_AXIS));
zoom.setBorder(BorderFactory.createCompoundBorder(
        BorderFactory.createTitledBorder("zoom"),
        BorderFactory.createEmptyBorder(0,0,0,0)));
zoom.add(plus);
zoom.add(minus);

a.add(zoom);

JPanel coordinates = new JPanel();
coordinates.setLayout(new GridBagLayout());
coordinates.setBorder(
    BorderFactory.createCompoundBorder(
        BorderFactory.createTitledBorder("coordinates"),
            BorderFactory.createEmptyBorder(0,0,0,0)));

    JLabel lonL = new JLabel();
    JLabel latL = new JLabel();
    JLabel zoomL = new JLabel();

    lonL.setText("longitude");
    latL.setText("latitude");
    zoomL.setText("zoom");

  lonT.setColumns(7);
  latT.setColumns(7);
  zoomT.setColumns(7);
  lonT.setEditable(false);
  latT.setEditable(false);
  zoomT.setEditable(false);
```

```
        c.gridx = 0;
        c.gridy = 0;
        coordinates.add(lonL,c);
        c.gridx = 1;
        c.gridy = 0;
        coordinates.add(lonT,c);
        c.gridx = 0;
        c.gridy = 1;
        coordinates.add(latL,c);
        c.gridx = 1;
        c.gridy = 1;
        coordinates.add(latT,c);
        c.gridx = 0;
        c.gridy = 2;
        coordinates.add(zoomL,c);
        c.gridx = 1;
        c.gridy = 2;
        coordinates.add(zoomT,c);

        a.add(coordinates);
        a.setBorder(BorderFactory.createEmptyBorder(0,0,0,0));
        a.setBorder(BorderFactory.createCompoundBorder(
          BorderFactory.createTitledBorder("navigate"),
        BorderFactory.createEmptyBorder(0,0,0,0)));

    reload.addActionListener(this);
    up.addActionListener(this);
    down.addActionListener(this);
    left.addActionListener(this);
    right.addActionListener(this);
    plus.addActionListener(this);
    minus.addActionListener(this);
    l.addMouseListener(this);
    l.addMouseMotionListener(this);

    return a;
  }


  public Image download_map(String url) throws IOException {
    HttpURLConnection c = null;
    InputStream is = null;
    byte[] data = null;
    int rc;
    Image image = null;
```

```
    try {
      c = (HttpURLConnection)new URL(url).openConnection();
      rc = c.getResponseCode();
      if (rc != HttpURLConnection.HTTP_OK) {
        throw new IOException("HTTP response code: " + rc);
      }
      is = c.getInputStream();

      String type = c.getContentType();

      int len = c.getContentLength();
      if (len > 0) {
        int actual = 0;
        int bytesread = 0 ;
        data = new byte[len];
        while ((bytesread != len) && (actual != -1)) {
          actual = is.read(data, bytesread, len - bytesread);
          bytesread += actual;
        }
      } else {
        int ch, count=0;
        byte []mdata = new byte [len];
        if (mdata == null) {
          throw new IOException("problems");
        }
        while ((ch = is.read()) != -1) {
          mdata[count] = (byte)ch;
          count++;
        }
        data = new byte[count];
        System.arraycopy (mdata, 0, data, 0, count);
      }
    } catch (ClassCastException e) {
      throw new IllegalArgumentException("Not an HTTP URL");
    } finally {
      if (is != null) is.close();
    }
    image = mainPanel.getToolkit().createImage(data);
    return image;
  }

  private void drawRectangle(Graphics g, int startX, int startY,
     int stopX, int stopY ) {
    int x, y, w, h;
    x = Math.min(startX, stopX);
```

```
    y = Math.min(startY, stopY);
    w = Math.abs(startX - stopX);
    h = Math.abs(startY - stopY);
    g.drawRect(x, y, w, h);
  }

  public void mouseClicked(MouseEvent arg0) {
  }

  public void mouseDragged(MouseEvent event) {
     int x = event.getX();
       int y = event.getY();

       Graphics g = l.getGraphics();
       g.setXORMode(Color.lightGray);
       drawRectangle(g, startX, startY, lastX, lastY);
       drawRectangle(g, startX, startY, x, y);

       lastX = x;
       lastY = y;
  }

  public void mouseEntered(MouseEvent arg0) {
  }

  public void mouseExited(MouseEvent arg0) {
  }

  public void mouseMoved(MouseEvent arg0) {
  }

  public void mousePressed(MouseEvent event) {
    startX = event.getX();
      startY = event.getY();
      lastX = startX;
      lastY = startY;
  }

  public void mouseReleased(MouseEvent event) {
    boolean tooSmall = false;
    tooSmall = Math.abs(startX - event.getX()) < 10;
    if(!tooSmall) tooSmall = Math.abs(startY - event.getY()) <
       10;

    Graphics g = l.getGraphics();
```

```
      if(tooSmall)g.setXORMode(Color.lightGray);
      else g.setXORMode(Color.BLACK);
      drawRectangle(g, startX, startY, lastX, lastY);

      if (tooSmall)return;

    double midx,midy;

    midx = startX + (event.getX() - startX) / 2;
    midy = startY + (event.getY() - startY) / 2;

    client.x -= (width / 2 - midx) * client.zoom / (width);
    client.y += (height / 2 - midy) * client.zoom / (height);

    client.zoom = (Math.abs(event.getX() - startX) * client.zoom
       ) / width;

    try {
      reload_map(client.x,client.y, client.zoom);
    } catch (Exception e) {
      textArea.append(e.getMessage());
    }
  }

  public void updateLabels(double x, double y, double zoom){
      lonT.setText(String.valueOf(x));
      latT.setText(String.valueOf(y));
      zoomT.setText(String.valueOf(zoom));
  }

    public void reload_map(double x, double y, double zoom)
       throws fleetmanagement.IOException, RemoteException,
       IOException{
    l.setCursor(Cursor.getPredefinedCursor(Cursor.WAIT_CURSOR));

    updateLabels(x,y,zoom);

    textArea.append("requesting...");
      textArea.repaint();
        ImageIcon a = new ImageIcon();
        Image b = download_map(client.getMapURL(x,y,zoom,members
           ));
        a.setImage(b);
        l.setIcon(a);
        textArea.append("ok\n");
```

```
        l.setCursor(Cursor.getPredefinedCursor(Cursor.
            DEFAULT_CURSOR));
    }
}
```

Appendix C

Fleet Management Service

C.1 fleetIF.java

```
package fleetmanagement;

import java.io.IOException;
import java.net.MalformedURLException;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.Vector;
import location.Location;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.MarkerDescription;

public interface fleetIF extends Remote {
  public String getMapURL(double x, double y, double zoom,
      fleetMember[] membersToShow) throws IOException,
      RemoteException;
  public fleetMember[] getFleet()throws RemoteException;
  public void setFleet(fleetMember[] a)throws RemoteException;
  public Location localize(fleetMember a)throws RemoteException;
  public void deleteFleetMember(String a)throws RemoteException;
  public void addFleetMember(fleetMember a)throws
      RemoteException;
  public void replaceFleetMember(String a,fleetMember b)throws
      RemoteException;
}
```

C.2 fleetImpl.java

```
package fleetmanagement;

import java.io.IOException;
import java.net.MalformedURLException;
import java.net.URL;
import java.rmi.RemoteException;
import java.util.Vector;
import location.*;

import org.apache.axis.AxisFault;

import com.esri.arcweb.v2.IAuthentication;
import com.esri.arcweb.v2.IAuthenticationStub;
import com.esri.arcweb.v2.IMapImage;
import com.esri.arcweb.v2.IMapImageStub;
import com.themindelectric.www._package.
    com_esri_is_services_common_v2_geom.Envelope;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.MapImageInfo;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.MapImageOptions;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.MapImageSize;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.PixelCoord;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.MarkerDescription;
import com.themindelectric.www._package.
    com_esri_is_services_glue_v2_mapimage.LabelDescription;
import com.themindelectric.www._package.
    com_esri_is_services_common_v2_geom.Point;


class myAuth  extends IAuthenticationStub implements
    IAuthentication{
  public myAuth() throws AxisFault, MalformedURLException {
    super(new URL("https://arcweb.esri.com/services/v2/
        Authentication"),null);
  }
}

class myLocationService extends LocationIFBindingStub implements
    LocationIF{
  public myLocationService() throws java.net.
```

```
      MalformedURLException, RemoteException {
    super(new URL("http://localhost:8080/location-jaxrpc/
        location"),null);
  }
}

class myMapImage extends IMapImageStub implements IMapImage{
  public myMapImage() throws AxisFault, MalformedURLException {
    super(new URL("http://arcweb.esri.com/services/v2/MapImage")
        ,null);
  }
}

public class fleetImpl implements fleetIF {

  private static int width=500;
  private static int height=500;

  private String token;
  Vector fleetMembers=null;
  myLocationService myloc =null;

  public fleetImpl() throws Exception{
    myloc = new myLocationService();
    fleetMembers = new Vector();
    fleetMembers.addElement(new fleetMember("employee_1", "007",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_2", "603",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_3", "994",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_4", "333",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_5", "300",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_6", "040",
        0, 0, false));
    fleetMembers.addElement(new fleetMember("employee_7", "0011"
        , 0, 0, false));
    authenticate();
  }

  public synchronized Location localize(fleetMember who)throws
      RemoteException{
    for (int i=0;i<fleetMembers.size();i++){
```

```
      if (who.getName().equals(((fleetMember)fleetMembers.get(i)
          ).getName())) {
        Location  newlocation = myloc.getLocation(((fleetMember)
            fleetMembers.get(i)).getPhoneNumber());
        ((fleetMember)fleetMembers.get(i)).setX(newlocation.getX
            ());
        ((fleetMember)fleetMembers.get(i)).setY(newlocation.getY
            ());
        return newlocation;
      }
    }
    return new Location();
  }

  public synchronized fleetMember[] getFleet()throws
      RemoteException{
    fleetMember[] array = new fleetMember[fleetMembers.size()];
    for (int i=0;i<fleetMembers.size();i++){
      array[i] = (fleetMember)fleetMembers.get(i);
    }
    return array;
  }

  public synchronized void setFleet(fleetMember[] a)throws
      RemoteException{
    for (int i=0;i<a.length;i++){
      fleetMembers.addElement(a[i]);
    }
  }

  public synchronized void addFleetMember(fleetMember a)throws
      RemoteException{
    fleetMembers.addElement(a);
  }

  public synchronized void deleteFleetMember(String a) throws
      RemoteException{
    for (int i=0;i<fleetMembers.size();i++){
      fleetMember tmp = (fleetMember)fleetMembers.get(i);
      if (a.equals(tmp.getName())) {
        fleetMembers.removeElementAt(i);
      }
    }
  }
  public synchronized void replaceFleetMember(String a,
```

```
     fleetMember b) throws RemoteException{
    for (int i=0;i<fleetMembers.size();i++){
      fleetMember tmp = (fleetMember)fleetMembers.get(i);
      if (a.equals(tmp.getName())) {
        fleetMembers.removeElementAt(i);
        fleetMembers.addElement(b);
      }
    }
  }

  public synchronized void deleteAllFleet() throws
     RemoteException{
    fleetMembers.removeAllElements();
  }

    public synchronized void authenticate() throws
       MalformedURLException, RemoteException{
      myAuth myAuthentication = new myAuth();
      token = myAuthentication.getToken("test60","123qweasd",
         1440);
  }

    public synchronized String getToken(){
      return token;
    }

    public String getMapURL(double x,double y,double zoom,
       fleetMember[] membersToShow) throws MalformedURLException
       ,RemoteException
  {
    myMapImage mapImage=null;
    mapImage = new myMapImage();

    //Specify the map extent
    Envelope env = new Envelope();

    env.setMaxx(x+0.5*zoom);
    env.setMinx(x-0.5*zoom);
    env.setMaxy(y+0.5*zoom);
    env.setMiny(y-0.5*zoom);

    //Specify mapimage options
    MapImageOptions mapImageOptions = new MapImageOptions();
    mapImageOptions.setDataSource("TA.Streets.EU");
    mapImageOptions.setMapImageFormat("jpg");
```

```
MapImageSize mapImageSize = new MapImageSize();
mapImageSize.setHeight(height);
mapImageSize.setWidth(width);
mapImageOptions.setMapImageSize(mapImageSize);

//Specify optional map items (Scale Bar, Legend,Icon,Circle)
PixelCoord pixelCoord = new PixelCoord();
pixelCoord.setX(2);
pixelCoord.setY(13);
//specify scale bar location
mapImageOptions.setScaleBarPixelLocation(pixelCoord);
mapImageOptions.setDrawScaleBar(true);

int size=0;
for(int i=0;i<membersToShow.length;i++){
  if(membersToShow[i].isVisible())size++;
}

MarkerDescription markers[] = new MarkerDescription[size];

for(int i=0, k=0;i<membersToShow.length;i++){
  if (membersToShow[i].isVisible()){
    MarkerDescription marker = new MarkerDescription();
    Point myPoint = new Point();
    myPoint.setX(membersToShow[i].getX());
    myPoint.setY(membersToShow[i].getY());
    marker.setName("pushpin_blue.gif");
    marker.setIconDataSource("ESRI.Raster.Icons");
    marker.setColor("255,0,0");
    marker.setLabel(membersToShow[i].getName());
    marker.setLocation(myPoint);
    LabelDescription ld = new LabelDescription();
    ld.setAntialiasing("true");
    ld.setFont("Courier New");
    ld.setFontColor("0,0,0");
    ld.setFontSize(18);
    ld.setFontStyle("regular");
    ld.setInterval(15);
    ld.setOutlineColor("255,255,255");
    ld.setHalignment("right");
    ld.setValignment("top");
    ld.setTransparency(0.9);

    marker.setLabelDescription(ld);
    markers[k++] = marker;
```

```
        }
      }

      mapImageOptions.setMarkers(markers);

      MapImageInfo mapImageInfo;
      //Call mapimage web service
      mapImageInfo = mapImage.getMap(env,mapImageOptions,getToken
          ());
      return mapImageInfo.getMapUrl();
    }
}
```

C.3 fleetMember.java

```
package fleetmanagement;

public class fleetMember
{
    private String name;
    private String phone;
    private double X;
    private double Y;
    private boolean visible;

    public fleetMember(String name, String phone, double x,
        double y, boolean visible) {
    this.name = name;
    this.phone = phone;
    X = x;
    Y = y;
    this.visible = visible;
  }

    public fleetMember() {
  }
  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
  public String getPhoneNumber() {
    return phone;
  }
```

```
  public void setPhoneNumber(String phone) {
    this.phone = phone;
  }
  public boolean isVisible() {
    return visible;
  }
  public void setVisible(boolean visible) {
    this.visible = visible;
  }
  public double getX() {
    return X;
  }
  public void setX(double x) {
    X = x;
  }
  public double getY() {
    return Y;
  }
  public void setY(double y) {
    Y = y;
  }
}
```

C.4 build.xml

```
<project name="FleetManagementService" default="build" basedir="
    .">

  <property file="build.properties"/>
  <property name="axis.lib.dir" value="c:/Program Files/eclipse/
      plugins/org.apache.axis_1.1"/>
  <property name="catalina.dir" value="c:/tomcat-jwsdp-1.4"/>
  <property name="wsdl2java-generated.dir" value="src/com"/>

  <path id="catalina-server.path">
    <fileset dir="${catalina.dir}/server/lib">
      <include name="*.jar"/>
      <exclude name="log4j-1.2.8.jar"/>
    </fileset>
  </path>

  <path id="compile.classpath">
    <pathelement location="${javamail.jar}"/>
    <pathelement location="${jaf.jar}"/>
    <pathelement location="${jaxp-api.jar}"/>
```

```
    <pathelement location="${dom.jar}"/>
    <pathelement location="${sax.jar}"/>
    <pathelement location="${xalan.jar}"/>
    <pathelement location="${xercesImpl.jar}"/>
    <pathelement location="${jaxrpc-api.jar}"/>
    <pathelement location="${jaxrpc-impl.jar}"/>
    <pathelement location="${jaxrpc-spi.jar}"/>
    <pathelement location="${commons-logging.jar}"/>
    <pathelement location="${saaj-api.jar}"/>
    <pathelement location="${saaj-impl.jar}"/>
    <pathelement location="${relaxngDatatype.jar}"/>
    <pathelement location="${xsdlib.jar}"/>
    <pathelement location="${jax-qname.jar}"/>
    <pathelement location="${ant.jar}"/>
    <pathelement location="${build}/shared"/>
    <pathelement location="${build.dir}" />
    <pathelement location="${axis.lib.dir}/axis.jar" />
    <pathelement location="${axis.lib.dir}/axis-ant.jar" />
    <pathelement location="${axis.lib.dir}/commons-logging.jar"
       />
    <pathelement location="${axis.lib.dir}/commons-discovery.jar
       " />
    <pathelement location="${axis.lib.dir}/jaxrpc.jar" />
    <pathelement location="${axis.lib.dir}/wsdl4j.jar" />
    <pathelement location="${axis.lib.dir}/saaj.jar" />
    <pathelement location="${axis.lib.dir}/log4j-1.2.8.jar" />
  </path>

<path id="run.classpath">
     <path refid="compile.classpath"/>
     <pathelement location="${dist}/${client.jar}"/>
</path>

<taskdef name="deploy" classname="org.apache.catalina.ant.
    DeployTask">
     <classpath refid="catalina-server.path"/>
</taskdef>
<taskdef name="undeploy" classname="org.apache.catalina.ant.
    UndeployTask">
     <classpath refid="catalina-server.path"/>
</taskdef>
<taskdef name="start" classname="org.apache.catalina.ant.
    StartTask">
     <classpath refid="catalina-server.path"/>
</taskdef>
```

```
    <taskdef name="stop" classname="org.apache.catalina.ant.
        StopTask">
      <classpath refid="catalina-server.path"/>
  </taskdef>

    <target name="deploy"
    description="Deploys Web Application">
    <deploy url="${url}" username="${username}" password="${
        password}"
    path="/${context.path}" war="file:${war.path}"/>
    </target>

    <target name="undeploy"
        description="Undeploys Web Application">
        <undeploy url="${url}" username="${username}" password="
            ${password}"
          path="/${context.path}"
        />
    </target>

    <target name="redeploy"
         description="Undeploys and Deploys Web Aplication">
        <antcall target="undeploy" />
        <antcall target="deploy" />
    </target>

    <target name="start"
        description="Starts a Web application">
       <echo message="Starting the application...."/>
       <start
           url="${url}"
           username="${username}"
           password="${password}"
           path="/${context.path}"
        />
    </target>

    <target name="stop"
        description="Stops a Web application">
       <echo message="Stopping the application...."/>
       <stop
           url="${url}"
           username="${username}"
           password="${password}"
           path="/${context.path}"
```

```
      />
</target>

<target name="build" depends="build-service"
  description="Executes the targets needed to build the
     service.">
  <javac srcdir="src" destdir="build">
    <classpath refid="compile.classpath" />
  </javac>
</target>

<target name="build-service" depends="clean,compile-service,
   generate-sei-service,
    setup-web-inf,package-service,process-war"
   description="Executes the targets needed to build the
      service.">
</target>

<target name="clean" description="Removes the build directory"
   >
    <delete dir="${build}" />
    <delete dir="${dist}" />
</target>

<target name="compile-service" depends="prepare"
    description="Compiles the server-side source code">
    <echo message="Compiling the server-side source code...."/
       >
    <javac srcdir="${src}"
           destdir="${build}"
           includes="*.java"
           excludes="*Client.java">
      <classpath refid="compile.classpath"/>
    </javac>
</target>

<target name="prepare" description="Creates the build
   directory" >
  <echo message="Creating the required directories...." />
  <mkdir dir="${build}/${example}" />
</target>

<target name="generate-sei-service" description="Runs
   wscompile to generate the model file">
    <antcall target="run-wscompile">
```

```
        <param name="param1" value="-define -d ${build} -nd ${
            build}
          -classpath ${build} ${config.interface.file}
          -model ${build}/${model.file}"/>
      </antcall>
      <delete file="${build}/FleetManagement.wsdl"/>
  </target>

  <target name="setup-web-inf"
       description="Copies files to build/WEB-INF">
      <echo message="Setting up ${build}/WEB-INF...."/>
      <delete dir="${build}/WEB-INF" />
      <copy todir="${build}/WEB-INF/classes/${example}">
     <fileset dir="${build}/${example}" />
      </copy>
      <copy todir="${build}/WEB-INF/classes/com">
     <fileset dir="${build}/com" />
      </copy>
    <copy todir="${build}/WEB-INF/classes/location">
      <fileset dir="${build}/location" />
      </copy>
      <copy todir="${build}/WEB-INF/lib">
         <fileset dir="${axis.lib.dir}" includes="*.jar"
            excludes="log4j-1.2.8.jar"/>
         <fileset dir="c:/Program Files/soap/soap-2_3_1/lib"
            includes="*.jar"/>
         <fileset dir="c:/Program Files/soap/xerces-1_2_3"
            includes="*.jar"/>
           <fileset dir="c:/Program Files/soap/jaf-1.0.2"
              includes="*.jar"/>
         <fileset dir="c:/Program Files/soap/javamail-1.3.1/lib
            " includes="*.jar"/>
      </copy>

      <copy file="${build}/${model.file}" todir="${build}/WEB-
         INF" />
      <copy file="web.xml" todir="${build}/WEB-INF" />
      <copy file="jaxrpc-ri.xml" todir="${build}/WEB-INF" />
    </target>

    <target name="package-service" depends="prepare-dist"
       description="Packages the WAR file">
      <echo message="Packaging the WAR...."/>
      <delete file="${dist}/${portable.war}" />
      <jar jarfile="${dist}/${portable.war}" >
```

```
        <fileset dir="${build}" includes="WEB-INF/**" />
      </jar>
</target>

<target name="prepare-dist"
  description="Creates the dist directory" >
  <echo message="Creating the required directories...." />
  <mkdir dir="${dist}" />
</target>

<target name="process-war" depends="set-wsdeploy"
      description="Runs wsdeploy to generate the ties and
         create a deployable WAR file">
  <delete file="dist/${deployable.war}" />
  <antcall target="run-wsdeploy">
    <param name="param1" value="-o dist/${deployable.war} dist
       /${portable.war}"/>
  </antcall>
</target>

<target name="set-wsdeploy" >
  <condition property="wsdeploy" value="${wsdeploy.dir}/
     wsdeploy.bat">
    <os family="windows"/>
  </condition>
  <condition property="wsdeploy" value="${wsdeploy.dir}/
     wsdeploy.sh">
    <not>
      <os family="windows"/>
    </not>
  </condition>
</target>

<target name="set-wscompile" >
  <condition property="wscompile" value="${wscompile.dir}/
     wscompile.bat">
    <os family="windows"/>
  </condition>
  <condition property="wscompile" value="${wscompile.dir}/
     wscompile.sh">
    <not>
      <os family="windows"/>
    </not>
  </condition>
</target>
```

```
<target name="run-wscompile" depends="prepare,set-wscompile"
      description="Runs wscompile">
  <echo message="Running wscompile:"/>
  <echo message="  ${wscompile} ${param1}"/>
  <exec executable="${wscompile}">
     <arg line="${param1}"/>
  </exec>
</target>

<target name="run-wsdeploy" depends="prepare,set-wsdeploy"
      description="Runs wsdeploy">
  <echo message="Running wsdeploy:"/>
  <echo message="  ${wsdeploy} ${param1}"/>
  <exec executable="${wsdeploy}">
     <arg line="${param1}"/>
  </exec>
</target>

<target name="listprops" depends="set-wscompile"
   description="Displays values of some of the properties of
      this build file">
 <echo message="jaxrpc.home = ${jaxrpc.home}" />
 <echo message="wscompile = ${wscompile}" />
 <echo message="build = ${build}" />
 <echo message="src = ${src}" />
 <echo message="dist = ${dist}" />
 <echo message=" " />
 <echo message="example = ${example}" />
 <echo message="client.jar = ${client.jar}" />
 <echo message="client.class = ${client.class}" />
 <echo message=" " />
 <echo message="host = ${host}" />
 <echo message="port = ${port}" />
 <echo message="secure.port = ${secure.port}" />
 <echo message="trust.store = ${trust.store}" />
 <echo message="trust.store.password = ${trust.store.password}
    " />
 <echo message=" " />
 <echo message="username = ${username}" />
 <echo message="password = ${password}" />
 <echo message="url = ${url}" />
 <echo message=" " />
 <echo message="context.path = ${context.path}" />
 <echo message="secure.context.path = ${secure.context.path}"
```

```
      />
    <echo message="url.pattern = ${url.pattern}" />
    <echo message="endpoint.address = ${endpoint.address}" />
    <echo message="secure.endpoint = ${secure.endpoint}" />
    <echo message=" " />
    <echo message="war.path = ${war.path}" />
    <echo message="portable.war = ${portable.war}" />
    <echo message="deployable.war = ${deployable.war}" />
  </target>
</project>
```

C.5 web.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE web-app
  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN
      "
  "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
  <display-name>Fleet Management Web Service</display-name>
  <description>Fleet Management application</description>
  <session-config>
    <session-timeout>60</session-timeout>
  </session-config>
</web-app>
```

C.6 jaxrpc-ri.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<webServices
  xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/dd"
  version="1.0"
  targetNamespaceBase="urn:fleetmanagement"
  typeNamespaceBase="urn:fleetmanagement"
  urlPatternBase="/ws">

  <endpoint
    name="myfleet"
    displayName="Fleet Management Web Service"
    description="Fleet Management Web Service"
    interface="fleetmanagement.fleetIF"
    model="/WEB-INF/model.gz"
```

```
    implementation="fleetmanagement.fleetImpl" />

  <endpointMapping
    endpointName="myfleet"
    urlPattern="/fleet" />
</webServices>
```

C.7 build.properties

```
ant.jar = ${jwsdp.home}/apache-ant/lib/ant.jar
deployable.war = ${context.path}.war
tutorial.install = ${tutorial.home}/jwstutorial13
wscompile.dir = ${jaxrpc.home}/bin
secure.port = 8443
portable.war = ${context.path}-portable.war
host = localhost
example = fleetmanagement
url = http://${host}:${port}/manager
xsdlib.jar = ${jwsdp.shared}/lib/xsdlib.jar
dist = dist
commons-logging.jar = ${jwsdp.shared}/lib/commons-logging.jar
saaj.home = ${jwsdp.home}/saaj
tut.root = ${tutorial.install}
wsdeploy.dir = ${jaxrpc.home}/bin
jaf.jar = ${jwsdp.shared}/lib/activation.jar
saaj-api.jar = ${saaj.home}/lib/saaj-api.jar
sax.jar = ${jaxp.home}/lib/endorsed/sax.jar
jaxrpc-impl.jar = ${jaxrpc.home}/lib/jaxrpc-impl.jar
context.path = ${fleet.context}
config.wsdl.file = config-wsdl.xml
config.interface.file = config-interface.xml
jwsdp.home = c:/jwsdp-1.4
tutorial.home = c:/jwsdp-1.4/docs
src = src
javamail.jar = ${jwsdp.shared}/lib/mail.jar
dom.jar = ${jaxp.home}/lib/endorsed/dom.jar
jaxrpc.home = ${jwsdp.home}/jaxrpc
jaxp.home = ${jwsdp.home}/jaxp
url.pattern = /fleet
model.file = model.gz
war.path = c:/progra~1/eclipse/workspace/FleetManagementService
    /${dist}/${deployable.war}
password = 123qweasd
username = vrm
jax-qname.jar = ${jwsdp.shared}/lib/jax-qname.jar
```

```
xalan.jar = ${jaxp.home}/lib/endorsed/xalan.jar
port = 8080
jaxp-api.jar = ${jaxp.home}/lib/jaxp-api.jar
fleet.endpoint = http://${host}:${port}/${fleet.context}${url.
    pattern}
fleet.context = fleet-jaxrpc
build = build
jaxrpc-api.jar = ${jaxrpc.home}/lib/jaxrpc-api.jar
jaxrpc-spi.jar = ${jaxrpc.home}/lib/jaxrpc-spi.jar
client.jar = client.jar
relaxngDatatype.jar = ${jwsdp.shared}/lib/relaxngDatatype.jar
saaj-impl.jar = ${saaj.home}/lib/saaj-impl.jar
xercesImpl.jar = ${jaxp.home}/lib/endorsed/xercesImpl.jar
jwsdp.shared = ${jwsdp.home}/jwsdp-shared
custom = false
```

Appendix D

Location Service

D.1 locationIF.java

```
package LocationService;

import java.io.IOException;
import java.net.MalformedURLException;
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface locationIF extends Remote {
  public Location getLocation(String id) throws IOException,
     RemoteException;
}
```

D.2 locationImpl.java

```
package LocationService;

import java.io.IOException;
import java.net.MalformedURLException;
import java.net.URL;
import java.rmi.RemoteException;
import java.util.Vector;

class subscriber{
  private String phoneNumber;
  private Location location;

  public subscriber(){}
  public subscriber(String s,double x,double y){
```

```
    phoneNumber = s;
    location = new Location(x,y);
  }
  public Location getLocation(){
    return location;
  }
  public String getPhoneNumber(){
    return phoneNumber;
  }
}

public class locationImpl implements locationIF {
  Vector subscribers=null;
  public locationImpl(){
    subscribers  = new Vector();
    subscribers.addElement(new subscriber("0011"
        ,12.6507,55.6209));
    subscribers.addElement(new subscriber("007",12.5029,55.7693)
        );
    subscribers.addElement(new subscriber("603",12.5179,55.7626)
        );
    subscribers.addElement(new subscriber("994",12.7031,56.0543)
        );
    subscribers.addElement(new subscriber("333",12.6026,56.0327)
        );
    subscribers.addElement(new subscriber("300",12.999,55.585));
    subscribers.addElement(new subscriber("040",12.5079,55.7626)
        );
  }

  public synchronized Location getLocation(String s)throws
     RemoteException{
    for (int i=0;i<subscribers.size();i++){
      subscriber tmp = (subscriber)subscribers.get(i);
      if (s.equals(tmp.getPhoneNumber()))
          return tmp.getLocation();
    }
    throw new RemoteException("unknown subscriber");
  }
}
```

D.3 location.java

```
package LocationService;
public class Location implements java.io.Serializable {
```

```
    private double x;
    private double y;

    public Location() {
      x=0;
    y=0;
    }

    public Location(double px,double py) {
      x = px;
      y = py;
    }

    public double getX() {
        return x;
    }

    public void setX(double px) {
        x = px;
    }

    public double getY() {
        return y;
    }

    public void setY(double py) {
        y = py;
    }
}
```

D.4 build.xml

```
<project name="LocationService" default="build" basedir=".">

  <property file="build.properties"/>
  <property name="axis.lib.dir" value="c:/Program Files/eclipse/
     plugins/org.apache.axis_1.1"/>
  <property name="catalina.dir" value="c:/tomcat-jwsdp-1.4"/>

  <path id="catalina-server.path">
    <fileset dir="${catalina.dir}/server/lib">
      <include name="*.jar"/>
      <exclude name="log4j-1.2.8.jar"/>
    </fileset>
  </path>
```

```
<path id="compile.classpath">
  <pathelement location="${javamail.jar}"/>
  <pathelement location="${jaf.jar}"/>
  <pathelement location="${jaxp-api.jar}"/>
  <pathelement location="${dom.jar}"/>
  <pathelement location="${sax.jar}"/>
  <pathelement location="${xalan.jar}"/>
  <pathelement location="${xercesImpl.jar}"/>
  <pathelement location="${jaxrpc-api.jar}"/>
  <pathelement location="${jaxrpc-impl.jar}"/>
  <pathelement location="${jaxrpc-spi.jar}"/>
  <pathelement location="${commons-logging.jar}"/>
  <pathelement location="${saaj-api.jar}"/>
  <pathelement location="${saaj-impl.jar}"/>
  <pathelement location="${relaxngDatatype.jar}"/>
  <pathelement location="${xsdlib.jar}"/>
  <pathelement location="${jax-qname.jar}"/>
  <pathelement location="${ant.jar}"/>
  <pathelement location="${build}/shared"/>
  <pathelement location="${build.dir}" />
  <pathelement location="${axis.lib.dir}/axis.jar" />
  <pathelement location="${axis.lib.dir}/axis-ant.jar" />
  <pathelement location="${axis.lib.dir}/commons-logging.jar"
     />
  <pathelement location="${axis.lib.dir}/commons-discovery.jar
     " />
  <pathelement location="${axis.lib.dir}/jaxrpc.jar" />
  <pathelement location="${axis.lib.dir}/wsdl4j.jar" />
  <pathelement location="${axis.lib.dir}/saaj.jar" />
  <pathelement location="${axis.lib.dir}/log4j-1.2.8.jar" />
 </path>

<path id="run.classpath">
     <path refid="compile.classpath"/>
     <pathelement location="${dist}/${client.jar}"/>
</path>

<taskdef name="deploy" classname="org.apache.catalina.ant.
    DeployTask">
    <classpath refid="catalina-server.path"/>
</taskdef>
<taskdef name="undeploy" classname="org.apache.catalina.ant.
    UndeployTask">
    <classpath refid="catalina-server.path"/>
```

```
</taskdef>
<taskdef name="start" classname="org.apache.catalina.ant.
    StartTask">
     <classpath refid="catalina-server.path"/>
</taskdef>
  <taskdef name="stop" classname="org.apache.catalina.ant.
      StopTask">
     <classpath refid="catalina-server.path"/>
</taskdef>

  <target name="deploy"
  description="Deploys Web Application">
  <deploy url="${url}" username="${username}" password="${
      password}"
  path="/${context.path}" war="file:${war.path}"/>
  </target>

  <target name="undeploy"
      description="Undeploys Web Application">
       <undeploy url="${url}" username="${username}" password="
           ${password}"
         path="/${context.path}"
       />
  </target>

  <target name="redeploy"
       description="Undeploys and Deploys Web Aplication">
       <antcall target="undeploy" />
       <antcall target="deploy" />
  </target>

  <target name="start"
      description="Starts a Web application">
      <echo message="Starting the application...."/>
      <start
          url="${url}"
          username="${username}"
          password="${password}"
          path="/${context.path}"
       />
  </target>

  <target name="stop"
      description="Stops a Web application">
      <echo message="Stopping the application...."/>
```

```
      <stop
          url="${url}"
          username="${username}"
          password="${password}"
          path="/${context.path}"
        />
</target>

<target name="build" depends="build-service"
   description="Executes the targets needed to build the
      service.">
  <javac srcdir="src" destdir="build">
    <classpath refid="compile.classpath" />
  </javac>
</target>

<target name="build-service" depends="clean,compile-service,
    generate-sei-service,
     setup-web-inf,package-service,process-war"
    description="Executes the targets needed to build the
       service.">
</target>

<target name="clean" description="Removes the build directory"
   >
    <delete dir="${build}" />
    <delete dir="${dist}" />
</target>

<target name="compile-service" depends="prepare"
     description="Compiles the server-side source code">
    <echo message="Compiling the server-side source code...."/
       >
    <javac srcdir="${src}"
           destdir="${build}"
           includes="*.java"
           excludes="*Client.java">
      <classpath refid="compile.classpath"/>
    </javac>
</target>

<target name="prepare" description="Creates the build
    directory" >
  <echo message="Creating the required directories...." />
  <mkdir dir="${build}/${example}" />
```

```
</target>

<target name="generate-sei-service" description="Runs
    wscompile to generate the model file">
    <antcall target="run-wscompile">
      <param name="param1" value="-define -d ${build} -nd ${
          build}
        -classpath ${build} ${config.interface.file}
        -model ${build}/${model.file}"/>
    </antcall>
    <delete file="${build}/LocationService.wsdl"/>
</target>

<target name="setup-web-inf"
     description="Copies files to build/WEB-INF">
     <echo message="Setting up ${build}/WEB-INF...."/>
     <delete dir="${build}/WEB-INF" />
     <copy todir="${build}/WEB-INF/classes/${example}">
    <fileset dir="${build}/${example}" />
     </copy>
     <copy todir="${build}/WEB-INF/lib">
        <fileset dir="${axis.lib.dir}" includes="*.jar"
           excludes="log4j-1.2.8.jar"/>
        <fileset dir="c:/Program Files/soap/soap-2_3_1/lib"
           includes="*.jar"/>
        <fileset dir="c:/Program Files/soap/xerces-1_2_3"
           includes="*.jar"/>
          <fileset dir="c:/Program Files/soap/jaf-1.0.2"
             includes="*.jar"/>
        <fileset dir="c:/Program Files/soap/javamail-1.3.1/lib
           " includes="*.jar"/>
     </copy>

     <copy file="${build}/${model.file}" todir="${build}/WEB-
        INF" />
     <copy file="web.xml" todir="${build}/WEB-INF" />
     <copy file="jaxrpc-ri.xml" todir="${build}/WEB-INF" />
  </target>

  <target name="package-service" depends="prepare-dist"
      description="Packages the WAR file">
      <echo message="Packaging the WAR...."/>
      <delete file="${dist}/${portable.war}" />
      <jar jarfile="${dist}/${portable.war}" >
        <fileset dir="${build}" includes="WEB-INF/**" />
```

```
      </jar>
</target>

<target name="prepare-dist"
   description="Creates the dist directory" >
  <echo message="Creating the required directories...." />
  <mkdir dir="${dist}" />
</target>

<target name="process-war" depends="set-wsdeploy"
      description="Runs wsdeploy to generate the ties and
         create a deployable WAR file">
  <delete file="dist/${deployable.war}" />
  <antcall target="run-wsdeploy">
    <param name="param1" value="-o dist/${deployable.war} dist
       /${portable.war}"/>
  </antcall>
</target>

<target name="set-wsdeploy" >
  <condition property="wsdeploy" value="${wsdeploy.dir}/
     wsdeploy.bat">
    <os family="windows"/>
  </condition>
  <condition property="wsdeploy" value="${wsdeploy.dir}/
     wsdeploy.sh">
    <not>
      <os family="windows"/>
    </not>
  </condition>
</target>

<target name="set-wscompile" >
  <condition property="wscompile" value="${wscompile.dir}/
     wscompile.bat">
    <os family="windows"/>
  </condition>
  <condition property="wscompile" value="${wscompile.dir}/
     wscompile.sh">
    <not>
      <os family="windows"/>
    </not>
  </condition>
</target>
```

```
<target name="run-wscompile" depends="prepare,set-wscompile"
      description="Runs wscompile">
  <echo message="Running wscompile:"/>
  <echo message="  ${wscompile} ${param1}"/>
  <exec executable="${wscompile}">
     <arg line="${param1}"/>
  </exec>
</target>

<target name="run-wsdeploy" depends="prepare,set-wsdeploy"
      description="Runs wsdeploy">
  <echo message="Running wsdeploy:"/>
  <echo message="  ${wsdeploy} ${param1}"/>
  <exec executable="${wsdeploy}">
     <arg line="${param1}"/>
  </exec>
</target>

<target name="listprops" depends="set-wscompile"
   description="Displays values of some of the properties of
      this build file">
 <echo message="jaxrpc.home = ${jaxrpc.home}" />
 <echo message="wscompile = ${wscompile}" />
 <echo message="build = ${build}" />
 <echo message="src = ${src}" />
 <echo message="dist = ${dist}" />
 <echo message=" " />
 <echo message="example = ${example}" />
 <echo message="client.jar = ${client.jar}" />
 <echo message="client.class = ${client.class}" />
 <echo message=" " />
 <echo message="host = ${host}" />
 <echo message="port = ${port}" />
 <echo message="secure.port = ${secure.port}" />
 <echo message="trust.store = ${trust.store}" />
 <echo message="trust.store.password = ${trust.store.password}
    " />
 <echo message=" " />
 <echo message="username = ${username}" />
 <echo message="password = ${password}" />
 <echo message="url = ${url}" />
 <echo message=" " />
 <echo message="context.path = ${context.path}" />
 <echo message="secure.context.path = ${secure.context.path}"
    />
```

```
    <echo message="url.pattern = ${url.pattern}" />
    <echo message="endpoint.address = ${endpoint.address}" />
    <echo message="secure.endpoint = ${secure.endpoint}" />
    <echo message=" " />
    <echo message="war.path = ${war.path}" />
    <echo message="portable.war = ${portable.war}" />
    <echo message="deployable.war = ${deployable.war}" />
  </target>
</project>
```

D.5 web.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE web-app
  PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN
      "
  "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
  <display-name>Location Web Service</display-name>
  <description>Location application</description>
  <session-config>
    <session-timeout>60</session-timeout>
  </session-config>
</web-app>
```

D.6 jaxrpc-ri.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<webServices
  xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/dd"
    version="1.0"
    targetNamespaceBase="urn:location"
    typeNamespaceBase="urn:location"
    urlPatternBase="/ws">

    <endpoint
        name="myLocation"
        displayName="Location Web Service"
        description="Location Web Service"
        interface="LocationService.locationIF"
        model="/WEB-INF/model.gz"
        implementation="LocationService.locationImpl"/>
```

```
    <endpointMapping
        endpointName="myLocation"
        urlPattern="/location" />

</webServices>
```

D.7 build.properties

```
ant.jar = ${jwsdp.home}/apache-ant/lib/ant.jar
deployable.war = ${context.path}.war
tutorial.install = ${tutorial.home}/jwstutorial13
wscompile.dir = ${jaxrpc.home}/bin
secure.port = 8443
portable.war = ${context.path}-portable.war
host = localhost
example = LocationService
url = http://${host}:${port}/manager
xsdlib.jar = ${jwsdp.shared}/lib/xsdlib.jar
dist = dist
commons-logging.jar = ${jwsdp.shared}/lib/commons-logging.jar
saaj.home = ${jwsdp.home}/saaj
tut.root = ${tutorial.install}
wsdeploy.dir = ${jaxrpc.home}/bin
jaf.jar = ${jwsdp.shared}/lib/activation.jar
saaj-api.jar = ${saaj.home}/lib/saaj-api.jar
sax.jar = ${jaxp.home}/lib/endorsed/sax.jar
jaxrpc-impl.jar = ${jaxrpc.home}/lib/jaxrpc-impl.jar
context.path = ${location.context}
config.wsdl.file = config-wsdl.xml
config.interface.file = config-interface.xml
jwsdp.home = c:/jwsdp-1.4
tutorial.home = c:/jwsdp-1.4/docs
src = src
javamail.jar = ${jwsdp.shared}/lib/mail.jar
dom.jar = ${jaxp.home}/lib/endorsed/dom.jar
jaxrpc.home = ${jwsdp.home}/jaxrpc
jaxp.home = ${jwsdp.home}/jaxp
url.pattern = /location
model.file = model.gz
war.path = c:/progra~1/eclipse/workspace/LocationService/${dist
    }/${deployable.war}
password = 123qweasd
username = vrm
jax-qname.jar = ${jwsdp.shared}/lib/jax-qname.jar
```

```
xalan.jar = ${jaxp.home}/lib/endorsed/xalan.jar
port = 8080
jaxp-api.jar = ${jaxp.home}/lib/jaxp-api.jar
location.endpoint = http://${host}:${port}/${location.context}${
    url.pattern}
location.context = location-jaxrpc
build = build
jaxrpc-api.jar = ${jaxrpc.home}/lib/jaxrpc-api.jar
jaxrpc-spi.jar = ${jaxrpc.home}/lib/jaxrpc-spi.jar
client.jar = client.jar
relaxngDatatype.jar = ${jwsdp.shared}/lib/relaxngDatatype.jar
saaj-impl.jar = ${saaj.home}/lib/saaj-impl.jar
xercesImpl.jar = ${jaxp.home}/lib/endorsed/xercesImpl.jar
jwsdp.shared = ${jwsdp.home}/jwsdp-shared
custom = false
```

Appendix E

.NET ArcWeb client

E.1 client.cs

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using mi=WindowsApplication1.com.esri.arcweb;
using mi2=WindowsApplication1.com.esri.arcweb1;

namespace WindowsApplication1
{
  public class Form1 : System.Windows.Forms.Form
  {
    private System.Windows.Forms.PictureBox pictureBox5;
    private System.Windows.Forms.PictureBox pictureBox9;
    private System.Windows.Forms.PictureBox pictureBox10;
    private System.Windows.Forms.PictureBox pictureBox11;
    private System.Windows.Forms.PictureBox pictureBox12;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.PictureBox pictureBox13;
    private System.Windows.Forms.PictureBox pictureBox14;
    private System.Windows.Forms.PictureBox arrow_up;
    private System.Windows.Forms.PictureBox arrow_left;
    private System.Windows.Forms.PictureBox arrow_down;
    private System.Windows.Forms.PictureBox arrow_right;
    private System.Windows.Forms.PictureBox reduce;
    private System.Windows.Forms.PictureBox magnify;
```

```
public Double x=12.55;
public Double y=55.75;
public Double tmpx = 0;
public Double tmpy = 0;
public int mx=0;
public int my=0;

public int width=500;
public int height=500;

public Double zoom = 1;
public string token;
private System.Windows.Forms.ListBox listBox1;
private System.Windows.Forms.PictureBox pictureBox1;
private System.Windows.Forms.GroupBox navigate;
private System.Windows.Forms.GroupBox zoom_group;

private System.ComponentModel.Container components = null;

public Form1()
{
  InitializeComponent();
}

protected override void Dispose( bool disposing )
{
  if( disposing )
  {
    if (components != null)
    {
      components.Dispose();
    }
  }
  base.Dispose( disposing );
}

#region Windows Form Designer generated code
private void InitializeComponent()
{
  System.Resources.ResourceManager resources = new System.
     Resources.ResourceManager(typeof(Form1));
  this.arrow_up = new System.Windows.Forms.PictureBox();
  this.arrow_left = new System.Windows.Forms.PictureBox();
  this.arrow_down = new System.Windows.Forms.PictureBox();
  this.arrow_right = new System.Windows.Forms.PictureBox();
```

```
this.pictureBox5 = new System.Windows.Forms.PictureBox();
this.reduce = new System.Windows.Forms.PictureBox();
this.magnify = new System.Windows.Forms.PictureBox();
this.pictureBox9 = new System.Windows.Forms.PictureBox();
this.pictureBox10 = new System.Windows.Forms.PictureBox();
this.pictureBox11 = new System.Windows.Forms.PictureBox();
this.pictureBox12 = new System.Windows.Forms.PictureBox();
this.label1 = new System.Windows.Forms.Label();
this.pictureBox13 = new System.Windows.Forms.PictureBox();
this.pictureBox14 = new System.Windows.Forms.PictureBox();
this.listBox1 = new System.Windows.Forms.ListBox();
this.pictureBox1 = new System.Windows.Forms.PictureBox();
this.navigate = new System.Windows.Forms.GroupBox();
this.zoom_group = new System.Windows.Forms.GroupBox();
this.navigate.SuspendLayout();
this.SuspendLayout();
//
// arrow_up
//
this.arrow_up.BackColor = System.Drawing.SystemColors.
    Desktop;
this.arrow_up.Cursor = System.Windows.Forms.Cursors.Hand;
this.arrow_up.Image = ((System.Drawing.Bitmap)(resources.
    GetObject("arrow_up.Image")));
this.arrow_up.Location = new System.Drawing.Point(48, 144)
    ;
this.arrow_up.Name = "arrow_up";
this.arrow_up.Size = new System.Drawing.Size(24, 24);
this.arrow_up.TabIndex = 0;
this.arrow_up.TabStop = false;
this.arrow_up.Click += new System.EventHandler(this.
    arrow_up_Click);
//
// arrow_left
//
this.arrow_left.BackColor = System.Drawing.SystemColors.
    Desktop;
this.arrow_left.Cursor = System.Windows.Forms.Cursors.Hand
    ;
this.arrow_left.Image = ((System.Drawing.Bitmap)(resources
    .GetObject("arrow_left.Image")));
this.arrow_left.Location = new System.Drawing.Point(72,
    168);
this.arrow_left.Name = "arrow_left";
this.arrow_left.Size = new System.Drawing.Size(24, 24);
```

```
this.arrow_left.TabIndex = 1;
this.arrow_left.TabStop = false;
this.arrow_left.Click += new System.EventHandler(this.
    arrow_left_Click);
//
// arrow_down
//
this.arrow_down.BackColor = System.Drawing.SystemColors.
    Desktop;
this.arrow_down.Cursor = System.Windows.Forms.Cursors.Hand
    ;
this.arrow_down.Image = ((System.Drawing.Bitmap)(resources
    .GetObject("arrow_down.Image")));
this.arrow_down.Location = new System.Drawing.Point(48,
    192);
this.arrow_down.Name = "arrow_down";
this.arrow_down.Size = new System.Drawing.Size(24, 24);
this.arrow_down.TabIndex = 2;
this.arrow_down.TabStop = false;
this.arrow_down.Click += new System.EventHandler(this.
    arrow_down_Click);
//
// arrow_right
//
this.arrow_right.BackColor = System.Drawing.SystemColors.
    Desktop;
this.arrow_right.Cursor = System.Windows.Forms.Cursors.
    Hand;
this.arrow_right.Image = ((System.Drawing.Bitmap)(
    resources.GetObject("arrow_right.Image")));
this.arrow_right.Location = new System.Drawing.Point(24,
    168);
this.arrow_right.Name = "arrow_right";
this.arrow_right.Size = new System.Drawing.Size(24, 24);
this.arrow_right.TabIndex = 3;
this.arrow_right.TabStop = false;
this.arrow_right.Click += new System.EventHandler(this.
    arrow_right_Click);
//
// pictureBox5
//
this.pictureBox5.BorderStyle = System.Windows.Forms.
    BorderStyle.FixedSingle;
this.pictureBox5.Cursor = System.Windows.Forms.Cursors.
    Cross;
```

```
this.pictureBox5.Image = ((System.Drawing.Bitmap)(
    resources.GetObject("pictureBox5.Image")));
this.pictureBox5.Location = new System.Drawing.Point(120,
    64);
this.pictureBox5.Name = "pictureBox5";
this.pictureBox5.Size = new System.Drawing.Size(500, 500);
this.pictureBox5.SizeMode = System.Windows.Forms.
    PictureBoxSizeMode.CenterImage;
this.pictureBox5.TabIndex = 4;
this.pictureBox5.TabStop = false;
this.pictureBox5.MouseUp += new System.Windows.Forms.
    MouseEventHandler(this.pictureBox5_MouseUp);
this.pictureBox5.MouseMove += new System.Windows.Forms.
    MouseEventHandler(this.pictureBox5_MouseMove);
this.pictureBox5.MouseDown += new System.Windows.Forms.
    MouseEventHandler(this.pictureBox5_MouseDown);
//
// reduce
//
this.reduce.BackColor = System.Drawing.SystemColors.
    Desktop;
this.reduce.Cursor = System.Windows.Forms.Cursors.Hand;
this.reduce.Image = ((System.Drawing.Bitmap)(resources.
    GetObject("reduce.Image")));
this.reduce.Location = new System.Drawing.Point(32, 240);
this.reduce.Name = "reduce";
this.reduce.Size = new System.Drawing.Size(24, 24);
this.reduce.TabIndex = 5;
this.reduce.TabStop = false;
this.reduce.Click += new System.EventHandler(this.
    reduce_Click);
//
// magnify
//
this.magnify.BackColor = System.Drawing.SystemColors.
    Desktop;
this.magnify.Cursor = System.Windows.Forms.Cursors.Hand;
this.magnify.Image = ((System.Drawing.Bitmap)(resources.
    GetObject("magnify.Image")));
this.magnify.Location = new System.Drawing.Point(64, 240);
this.magnify.Name = "magnify";
this.magnify.Size = new System.Drawing.Size(24, 24);
this.magnify.TabIndex = 6;
this.magnify.TabStop = false;
this.magnify.Click += new System.EventHandler(this.
```

```
    magnify_Click);
//
// pictureBox9
//
this.pictureBox9.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox9.Location = new System.Drawing.Point(24,
    144);
this.pictureBox9.Name = "pictureBox9";
this.pictureBox9.Size = new System.Drawing.Size(24, 24);
this.pictureBox9.TabIndex = 8;
this.pictureBox9.TabStop = false;
//
// pictureBox10
//
this.pictureBox10.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox10.Location = new System.Drawing.Point(72,
    144);
this.pictureBox10.Name = "pictureBox10";
this.pictureBox10.Size = new System.Drawing.Size(24, 24);
this.pictureBox10.TabIndex = 9;
this.pictureBox10.TabStop = false;
//
// pictureBox11
//
this.pictureBox11.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox11.Location = new System.Drawing.Point(72,
    192);
this.pictureBox11.Name = "pictureBox11";
this.pictureBox11.Size = new System.Drawing.Size(24, 24);
this.pictureBox11.TabIndex = 10;
this.pictureBox11.TabStop = false;
//
// pictureBox12
//
this.pictureBox12.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox12.Location = new System.Drawing.Point(24,
    192);
this.pictureBox12.Name = "pictureBox12";
this.pictureBox12.Size = new System.Drawing.Size(24, 24);
this.pictureBox12.TabIndex = 11;
this.pictureBox12.TabStop = false;
```

```
//
// label1
//
this.label1.BackColor = System.Drawing.SystemColors.
    Desktop;
this.label1.Font = new System.Drawing.Font("Microsoft Sans
     Serif", 18F, System.Drawing.FontStyle.Bold, System.
    Drawing.GraphicsUnit.Point, ((System.Byte)(238)));
this.label1.ForeColor = System.Drawing.SystemColors.
    Control;
this.label1.Location = new System.Drawing.Point(16, 16);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(608, 32);
this.label1.TabIndex = 12;
this.label1.Text = "Map Web Service Demo Client";
//
// pictureBox13
//
this.pictureBox13.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox13.Location = new System.Drawing.Point(48,
    168);
this.pictureBox13.Name = "pictureBox13";
this.pictureBox13.Size = new System.Drawing.Size(24, 24);
this.pictureBox13.TabIndex = 13;
this.pictureBox13.TabStop = false;
//
// pictureBox14
//
this.pictureBox14.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox14.Cursor = System.Windows.Forms.Cursors.
    Hand;
this.pictureBox14.Image = ((System.Drawing.Bitmap)(
    resources.GetObject("pictureBox14.Image")));
this.pictureBox14.Location = new System.Drawing.Point(24,
    112);
this.pictureBox14.Name = "pictureBox14";
this.pictureBox14.Size = new System.Drawing.Size(72, 24);
this.pictureBox14.TabIndex = 14;
this.pictureBox14.TabStop = false;
this.pictureBox14.Click += new System.EventHandler(this.
    reload_map);
//
// listBox1
```

```
//
this.listBox1.Location = new System.Drawing.Point(16, 288)
    ;
this.listBox1.Name = "listBox1";
this.listBox1.Size = new System.Drawing.Size(88, 277);
this.listBox1.TabIndex = 20;
//
// pictureBox1
//
this.pictureBox1.BackColor = System.Drawing.SystemColors.
    Desktop;
this.pictureBox1.Cursor = System.Windows.Forms.Cursors.
    Hand;
this.pictureBox1.Image = ((System.Drawing.Bitmap)(
    resources.GetObject("pictureBox1.Image")));
this.pictureBox1.Location = new System.Drawing.Point(16,
    64);
this.pictureBox1.Name = "pictureBox1";
this.pictureBox1.Size = new System.Drawing.Size(88, 24);
this.pictureBox1.TabIndex = 21;
this.pictureBox1.TabStop = false;
this.pictureBox1.Click += new System.EventHandler(this.
    pictureBox1_Click);
//
// navigate
//
this.navigate.Controls.AddRange(new System.Windows.Forms.
    Control[] {
                                            this.zoom_group});
this.navigate.Location = new System.Drawing.Point(16, 96);
this.navigate.Name = "navigate";
this.navigate.Size = new System.Drawing.Size(88, 184);
this.navigate.TabIndex = 22;
this.navigate.TabStop = false;
this.navigate.Text = "navigate";
//
// zoom_group
//
this.zoom_group.Location = new System.Drawing.Point(8,
    128);
this.zoom_group.Name = "zoom_group";
this.zoom_group.Size = new System.Drawing.Size(72, 48);
this.zoom_group.TabIndex = 0;
this.zoom_group.TabStop = false;
this.zoom_group.Text = "zoom";
```

```
        //
        // Form1
        //
        this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
        this.ClientSize = new System.Drawing.Size(640, 579);
        this.Controls.AddRange(new System.Windows.Forms.Control[]
            {
                                            this.pictureBox1,
                                            this.listBox1,
                                            this.pictureBox14,
                                            this.pictureBox13,
                                            this.label1,
                                            this.pictureBox12,
                                            this.pictureBox11,
                                            this.pictureBox10,
                                            this.pictureBox9,
                                            this.magnify,
                                            this.reduce,
                                            this.arrow_right,
                                            this.arrow_down,
                                            this.arrow_left,
                                            this.arrow_up,
                                            this.pictureBox5,
                                            this.navigate});
        this.Name = "Form1";
        this.Text = "Map Web Service";
        this.navigate.ResumeLayout(false);
        this.ResumeLayout(false);

    }
    #endregion

    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]
    static void Main()
    {

        Application.Run(new Form1());
    }

    void image_clear(){
        pictureBox5.Image=null;
        pictureBox5.Refresh();
```

```
}

private void log(string par){
  listBox1.Items.Add(par);
  listBox1.Refresh();
  listBox1.SetSelected(listBox1.Items.Count-1,true);
}

private void authenticate(){
  try
  {
    log("authenticating...");
    mi2.Authentication myAuthentication = new mi2.
        Authentication();
    token = myAuthentication.getToken("test60","123qweasd",
        5);
    log("success");
  }
  catch(Exception e)
  {
    log("authentication failed\n" + e.Message);
    return;
  }
}

private void reload_map()
{
  reload_map(null,null);
}

private void reload_map(object sender, System.EventArgs e)
{
  image_clear();
  mi.MapImage mapImage = new mi.MapImage();

  //Specify the map extent
  mi.Envelope env = new mi.Envelope();
  env.maxx = x+0.5*zoom;
  env.minx = x-0.5*zoom;
  env.maxy = y+0.2*zoom;
  env.miny = y-0.5*zoom;

  //Specify mapimage options
  mi.MapImageOptions mapImageOptions = new mi.
      MapImageOptions();
```

```
    mapImageOptions.dataSource = "TA.Streets.EU";
    mapImageOptions.mapImageFormat = "jpg";
    mi.MapImageSize mapImageSize = new mi.MapImageSize();
    mapImageSize.height =Convert.ToInt32(height);
    mapImageSize.width = Convert.ToInt32(width);
    mapImageOptions.mapImageSize = mapImageSize;

    //Specify optional map items (Scale Bar, Legend,Icon,
        Circle)
      mi.PixelCoord pixelCoord = new mi.PixelCoord();
      pixelCoord.x = Convert.ToInt32(2);
      pixelCoord.y = Convert.ToInt32(13);
      //specify scale bar location
      mapImageOptions.scaleBarPixelLocation = pixelCoord;
      mapImageOptions.drawScaleBar = true;

    mi.MapImageInfo mapImageInfo;
    try
    {
      log("requesting...");
      //Call mapimage web service
      mapImageInfo = mapImage.getMap(env,mapImageOptions,token
          );
    }
    catch(Exception ex){
      log("request failed\n"+ex.Message);
      return;
    }

    byte[] map;
    try
    {
      log("downloading...");
      System.Net.WebClient a = new System.Net.WebClient();
      map = a.DownloadData(mapImageInfo.mapUrl);
      System.IO.MemoryStream str = new System.IO.MemoryStream
          ();
      str.Write(map,0,map.Length);
      pictureBox5.Image= Image.FromStream(str);
    }
    catch(Exception ex){
      log("download failed\n"+ex.Message);
      return;
    }
  }
```

```
private void arrow_left_Click(object sender, System.
    EventArgs e)
{
  x+=0.2*zoom;
  reload_map();
}

private void arrow_up_Click(object sender, System.EventArgs
    e)
{
  y+=0.2*zoom;
  reload_map();
}

private void arrow_right_Click(object sender, System.
    EventArgs e)
{
  x-=0.2*zoom;
  reload_map();
}

private void arrow_down_Click(object sender, System.
    EventArgs e)
{
  y-=0.2*zoom;
  reload_map();
}

private void magnify_Click(object sender, System.EventArgs e
    )
{
  zoom=zoom/2;
  reload_map();
}

private void reduce_Click(object sender, System.EventArgs e)
{
  zoom=zoom*2;
  reload_map();
}

private void pictureBox1_Click(object sender, System.
    EventArgs e)
{
```

```
        authenticate();
    }

    private void pictureBox5_MouseDown(object sender, System.
        Windows.Forms.MouseEventArgs e)
    {
      mx = e.X;
      my = e.Y;
      pictureBox5.Capture=true;
    }

    private void pictureBox5_MouseUp(object sender, System.
        Windows.Forms.MouseEventArgs e)
    {
      Double endx,endy;
      endx = mx + ((e.X-mx)/2);
      endy = my + ((e.Y-my)/2);
      zoom = (System.Math.Abs(e.X-mx)*zoom)/width;
      x -= (width/2-endx)*zoom/(width);
      y += (height/2-endy)*zoom/(height);
      pictureBox5.Capture=false;
      reload_map();
    }

    private void pictureBox5_MouseMove(object sender, System.
        Windows.Forms.MouseEventArgs e)
    {
      if(pictureBox5.Capture&&pictureBox5.Image!=null)
      {
        System.Drawing.Graphics a =  System.Drawing.Graphics.
            FromImage(pictureBox5.Image);
        System.Drawing.Pen pen2 = new Pen(Color.Empty);
        if(mx<e.X)
        {
          if(my<e.Y) a.DrawRectangle(pen2,mx,my,System.Math.Abs(
              mx-e.X),System.Math.Abs(my-e.Y));
          else a.DrawRectangle(pen2,mx,e.Y,System.Math.Abs(mx-e.
              X),System.Math.Abs(my-e.Y));
        }
        else
        {
          if(my<e.Y) a.DrawRectangle(pen2,e.X,my,System.Math.Abs
              (mx-e.X),System.Math.Abs(my-e.Y));
          else a.DrawRectangle(pen2,e.X,e.Y,System.Math.Abs(mx-e
              .X),System.Math.Abs(my-e.Y));
```

```
            }
            pictureBox5.Refresh();
            a.Dispose();
        }
    }
  }
}
```