Literature Review

Graham Hunter
University of Cape Town

ghunter@cs.uct.ac.za

Abstract

This literature synthesis covers material pertaining to a mobile shopping system. Shopping behaviour is explored and certain patterns of behaviour are found that will be useful in designing the system. Certain software patterns that could be useful to the design are discussed. As the system has a Web interface as well as a mobile interface data needs to be sent to possibly heterogeneous clients. The possible middleware for this is covered, namely Java RMI and Web Services, the inter-operability of each is covered, and their performance in terms of network and CPU resources. The possible communication protocols that could be used are discussed and compared. Finally similar systems that have been developed are explored

1. Introduction

The literature being discussed is that pertaining to a mobile shopping system. This system will provide users with services that will aid in shopping such allowing users to collaborate remotely on a shared shopping list. This can be done through a Web browser or a cell-phone.

It will be split into three components, a Web application, mobile application, and an application managing the data for both applications. The application managing the data will assume the role of the server with the other applications acting as clients. This paper will focus on this server, while covering similar systems as well as the possible technology required in hosting the server and communicating with the different components.

2. Related Work

2.1 Shopping Behaviour

The design of the system relies on how people behave in a shopping environment, what functionality would be used and where. Based on a user study several common shopping behaviours are discussed [15].

There exist two kinds of shoppers, those who make mental lists and those who make physical lists. Those who keep physical lists tend to carry it on their person. Some occasionally leave the list in the shopping cart. Many of those who keep physical lists mark off items as they are bought, showing the need for interaction between the shopper and their list. Shoppers who keep mental lists rely on browsing behaviour to remind them of what items should be bought.

Shopping typically involves three phases:

- Pre-shopping phase which is where the shopper plans to go shopping and creates a list.
- Shopping phase where the person is choosing the items.
- Checkout phase where payment is made.

Shoppers often pick up items or otherwise use their hands while shopping.

Even though mobile phones are used more often than PDAs, the size of a PDA was preferred over that of a mobile phone for use while shopping.

Shoppers tend to visit the same shop often and follow the same route.

2.2 Related Systems

Systems that are related in some way to the proposed one are discussed below.

2.2.1 Similar Architecture

Listed are example systems that deliver data to either a Web interface or a mobile interface using a client/server approach.

2.2.1.1 Google Maps

Google Maps is a Web application that allows one to view road maps through a Web interface [11], as well as a mobile interface [12]. This application allows the user to view real-time traffic, get directions and search for locations. Using SMS mobile clients are limited to searching maps by text, but using a mobile application the client can view maps graphically, communicating with the server through the use of WAP. Much like the mobile shopping system, Google Maps interacts with both a Web and mobile interface to provide information to users. The difference is that the system does not deliver personalized information.

2.2.1.1 Google Calender

Google Calender is a similar application to Google Maps, but provides functionality for creating and maintaining personal and public calendars [13]. Events can be added by using natural language such as "Brunch with mom at Java Cafe 11am on Saturday" which is then parsed. Users can choose to receive SMS or email event notifications on their mobile device . Calendars can also be shared between users, allowing multiple people to edit the

calendar depending on the access rights. The application can be accessed through a Web browser or a mobile device. Google Calendar is very similar to the Mobile Shopping system architecturally and functionally. It allows the management of a list through similar interfaces, provides personalized information and lets users collaborate on a single list.

2.2.2 Similar Shopping Systems

Systems that deal with providing shoppers services to help them in the store are shown below.

2.2.2.1 Easi-Order

The Easi-Order system works on using PalmPilots to create a shopping list, then sending that shopping list to a store and having the items pre-picked [14]. The system can pre-populate the shopping list used based on the user's shopping history, as well as offer "impulse buy" suggestions. The store's products are electronically stored in a product index in categories and using a barcode scanner on the PalmPilots products can be entered into the list by scanning their barcode.

This system is similar in allowing shoppers to create and manage shopping lists. The difference is that each user has a separate list, whereas the Mobile Shopping system allows multiple users to have one list.

2.2.2.2 U-Scan Shopper

The U-Scan Shopper system involves a monitor being attached to a shopping cart handle which can display promotions when the shopper is near certain items [16]. Additionally a bar code scanner is on the shopping cart allowing the shopper to scan items for themselves and pay at an automated station. Finally the system automatically generates a shopping list for each shopper based on their shopping history with that store, viewable on the monitor.

The similarity of this system is that it allows the shopper to view their shopping list while shopping. The difference is that the U-Scan system automatically generates this list, whereas the Mobile Shopping system allows the user to create their own.

2.2.2.3 Mobile Shopping Assistant

The mobile shopping assistant is a system developed to allow shoppers to access Web services that are published by the store by using their mobile device [2]. Some of the services available allow the customer to create a virtual shopping list, receive promotions and view a floor plan of the shop. The mobile devices used were normal mobile phones running a J2ME application. The mobile devices communicate with the Web services using SOAP messages. Caching and compressing the messages for quicker data exchange. Some of the problems of mobile computing that this system tries to overcome include:

- unstable network connection status.
- Limited bandwidth of mobile communication channels.
- No efficient XML parser.

Some of the problems encountered in development include local cache management on the mobile device, pre-fetching data from the Web services, data consistency and data conflicts.

In summary there are many systems that have a similar architectural design in having a Web and cell-phone interface connect to a server, and a few systems that deal with providing services to shoppers in stores.

3. Software Patterns

Software patterns provide proven and tested architecture designs that take into account issues not readily apparent. Using applicable software patterns is then important to reducing errors in the project. Two software patterns are discussed below.

Model –View - Controller

The Model-View-Controller framework divides an application into three parts, the model where the core application is, the controller which directs the flow of the actions, and the view which handles presentation [3]. In this system the Web and mobile device interfaces would be the View, the Controller would be what links the Web and mobile application inputs to actions that the component handling the data will perform. The model responds to input from the controller, returning data from the database or changing the state of the view depending on the business model. The benefit of this model is that individuals can develop each component separately.

Publish / Subscribe

The Publish/Subscribe pattern refers to a model where information providers publish events to the system and information consumers who subscribe to events that they want notification of [4]. The benefits of this model are that communication is asynchronous as information providers and consumers are decoupled, meaning if the clients are not reachable notifications will be stored and sent later. With this model the mobile clients can receive event notifications without having to be continuously connected to the system

3. Middleware

Middleware is a software component that sits between the network operating system and an application. It facilitates communication and management of distributed components in a network [10]. The following are examples of middleware.

3.1 Java Remote Method Invocation

Remote method invocation allows remote Java objects to be invoked from other Java virtual machines or remote hosts [6]. RMI has been in the Java SDK since version 1.1 and is used to develop distributed applications in Java. The benefit of using Java RMI is that it requires no additional libraries or modification of normal Java application. One drawback is that the client needs an open port to communicate, which is not available if the server has a firewall. Compared to Web services, using RMI is significantly faster if there are open ports to communicate in terms of network and hardware performance.

3.2 XML-RPC

XML-RPC is a way to make RPC-style function calls to a machine over a network [10]. XML-RPC communicates using the HTTP protocol and XML messages. The benefits of using XML-RPC is to have

- An easy to implement standard.
- Discoverable services.
- Messages that can go through firewalls.

XML-RPC is different from other RMI protocols as it does not need stubs generated beforehand [7]. Instead primitives are used to construct requests and get responses. XML-RPC does not marshal arguments as all data is sent as text.

3.3 Web Services

Web services refer to an application component that can be accessed over the internet and used remotely. Web services use XML and HTTP as these are interoperable standards [1]. Any computer that can communicate via HTTP can use a Web service. The difference between Web services and its predecessors is that it uses the SOAP protocol to communicate, has an interface defined using WSDL and is discoverable through the Universal Description, Discovery and Integration (UDDI) service. The benefit of using Web services is that of interoperability between different platforms such as .NET and Java. A set of APIs for Java are available that can be used to implement a Web service called the Java Web Services Developer Pack. One of these is the Java API for XML based Remote Procedure Call (JAX-RPC), which makes implementing a Web service similar to creating an RMI remote object. Web services differ from RMI in that they are executed in a Web container as servlets. The Web services classes can be converted to servlets or a servlet handler can be used to forward requests. Compared to RMI some of the features that Web services do not have include:

- Using object references.
- Distributed garbage collection.
- Dynamic class loading.
- Remote object activation.

Web services have decreased network performance compared to RMI, due to the overhead of using SOAP. Using SOAP messages, XML needs to be serialized and de-serialized, whereas RMI uses binary serialization.

Each middleware provides different functionality and varying degrees of interoperability depending on how each sends and receives messages. An overview of the message structure of the communication protocol different middleware uses is discussed next.

4. Communication protocols

As bandwidth is limited when communicating with mobile clients, the messages each middleware used should be minimal.

4.1 SOAP

SOAP requests consist of a header and a body. The body contains the XML request object, while the header has information not required for the request [7].

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
  xmlns:m="http://www.uct.ac.za/shop">
  <soap:Header>
    <m:ID>1234</t>
  </soap:Header>
  <soap:Body>
    <m:GetList>
      <m:ListNamee>Example</m:ListName>
    </m:GetList>
  </soap:Body>
</soap:Envelope>
```

Fig 1. SOAP Request message.

The SOAP protocol was not designed for wireless communication and does not consider the limited resources that a mobile device has [8]. SOAP is generally used over HTTP and TCP, which has the following limitations :

- Congestion avoidance in TCP assumes packet losses are because of congestion, in a mobile network this could be due to disconnections instead .
- Using TCP requires a connection to be established, resulting in increased overhead as acknowledgement packets are sent.
- Network is not utilized well as a SOAP message that carries only a small amount of data will still require a connection being initialized.

By using client side caching and compressing the SOAP messages the performance of mobile clients can be significantly improved by up to 800%.

4.2 REST

REST uses standard HTTP GET requests to call functionality [6]. The parameter data is included as an HTTP request parameter. The protocol is the simplest but relies on the HTTP protocol.

```
GET /List?ListName=Example HTTP/1.1
Host: www.uct.ac.za
```

Fig 2. REST request message.

4.3 XML-RPC

XML-RPC is verbose compared to non-xml protocols like REST [9], but uses fewer bytes per message than SOAP. A request message that passes one integer is 168 bytes of which 41 bytes are used to mark up the integer parameter. Compression can be used on these messages to reduce the size. Another drawback of XML-RPC is that retrieving data using XPath can be complicated as the elements are not named semantically [6].

```
<?xml version="l.0" encoding=
"ISO-8859-i"?>
<methodCall>
<methodName>act. GetList</methodName>
<params>
<param>
<value><string>Example</string></value>
</param>
</params>
</methodCall>
```

Fig 3. XML-RPC request message.

The protocols differ in how many bytes each message needs to be, with REST requiring the least, and SOAP the most. XML-RPC lies in the middle, but requires complex parsing.

5. Web Servers

The choice of what Web server software to use is important as it must support whatever middleware is chosen, as well as being easy to maintain and develop for.

5.1 Jakarta-Tomcat

Jakarta-Tomcat is the Apache Software Foundation's project for handling servlets and JSPs. A servlet is a program that is executed in response to an HTTP request and returns an HTTP response [5]. Servlets are object classes and are run inside the Java Virtual Machine, which means servlets can run on many operating systems. As the software runs Java it allows the use of Java-RMI as well as Web services.

5.2 Microsoft Internet Information Services

Microsoft Internet Information Services (IIS) is a Web Server built into the Microsoft Windows framework that can handle Web services and Web applications developed in ASP.NET [18]. As IIS does not support Java, Java RMI cannot be used. When developing Web services in ASP.NET many of the details such as creating WSDL documents, creating the SOAP messages and Web Discovery Service File are done automatically by Visual Studio .NET [17].

Summary

Shopping behaviour has been explored in user studies before. Shoppers either keep a physical list or rely on the shop environment to remind them of what to buy. Meaning that a shopping system should offer a list management service and the ability to view a floor plan of the shop with items listed. Shopping can be divided into three phases, a phase where the shopper decides what to buy, a phase where they choose the items at the shop, and a phase where they pay for the items. The shopping system should provide different services at each stage. Shopping involves the frequent use of both hands and constant attention, therefore the system should be fast to use as well as usable by one hand only.

In terms of similar systems there are many that use both a mobile and a Web interface to show data. For the mobile interface multiple communication protocols can be used such as SMS for simple text response/request messages or a mobile application that uses GPRS to interface with the server. Systems that focus on providing mobile services while shopping have been implemented before. Some automatically generate a shopping list based on previous purchases, while others allow the shopper to create and manage their own lists. One used a PalmPilot as a mobile device, another a custom device attached to the shopping cart and finally one used a mobile phone. Neither system had a Web interface providing similar services that the mobile device had. The problems highlighted in developing these systems involved those related to mobile computing. Mobile devices have unreliable

network connections, limited bandwidth and a limited XML parser. Some systems use client-side caching of data and compression of the messages used to increase performance.

Two middleware choices were covered, Java RMI which allowed a normal application to communicate remotely with another, and XML based Web Services which requires a Web Server. Performance wise Java RMI uses less network bandwidth due to the shorter messages and less CPU resources because of not needing to serialize and de-serialize XML messages. The drawback is that it is less interoperable than Web Services as the clients need to be able to run Java code. The choice of middleware is important due to the limited resources the mobile client has. Using tomcat as a server allows the use of Java RMI, Web Services and normal Web Applications, whereas using IIS without Web Services drastically reduces interoperability.

References

1. Chavda, K. F. 2004. Anatomy of a Web service. J. Comput. Small Coll. 19, 3 (Jan. 2004), 124-134.
2. Wu, H. and Natchetoi, Y. 2007. Mobile shopping assistant: integration of mobile applications and Web services. In Proceedings of the 16th international Conference on World Wide Web (Banff, Alberta, Canada, May 08 - 12, 2007). WWW '07. ACM Press, New York, NY, 1259-1260.
3. Cugola, G. and Jacobsen, H. 2002. Using publish/subscribe middleware for mobile systems. SIGMOBILE Mob. Comput. Commun. Rev. 6, 4 (Oct. 2002), 25-33.
4. Lerner, R. M. 2001. At the Forge: Server-Side Java with Jakarta-Tomcat. Linux J. 2001, 84es (Apr. 2001), 10.
5. Juric, M. B., Kezmah, B., Hericko, M., Rozman, I., and Vezocnik, I. 2004. Java RMI, RMI tunneling and Web services comparison and performance analysis. SIGPLAN Not. 39, 5 (May. 2004), 58-65.
6. SOAP, REST and XML-RPC. http://www.kbcafe.com/rss/?guid=20060704042846
7. Allman, M. 2003. An evaluation of XML-RPC. SIGMETRICS Perform. Eval. Rev. 30, 4 (Mar. 2003), 2-11.
8. Phan, K. A., Tari, Z., and Bertok, P. 2006. A benchmark on soap's transport protocols performance for mobile applications. In Proceedings of the 2006 ACM Symposium on Applied Computing (Dijon, France, April 23 - 27, 2006). SAC '06. ACM Press, New York, NY, 1139-1144.
9. Hanslo, W. and MacGregor, K. 2004. The efficiency of XML as an intermediate data representation for wireless middleware communication. In Proceedings of the 2004 Annual Research Conference of the South African institute of Computer Scientists and information Technologists on IT Research in Developing Countries (Stellenbosch, Western Cape, South Africa, October 04 - 06, 2004). G. Marsden, P. Kotze, and A. Adesina-Ojo, Eds. ACM International Conference Proceeding Series, vol. 75. South African Institute for Computer Scientists and Information Technologists, 279-283.
10. XML-RPC Specification. Last visited July 2007. http://www.xmlrpc.com/spec
11. Google Maps. http://maps.google.com/
12. Google Mobile Maps. http://www.google.com/gmm/index.html
13. Google Calendar. http://www.google.com/googlecalendar/overview.html
14. CASE STUDY: Customers scan their palms before http://specials.ft.com/ln/ftsurveys/q4a2e.htm
15. Newcomb, E., Pashley, T., and Stasko, J. 2003. Mobile computing in the retail arena. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Ft. Lauderdale, Florida, USA, April 05 - 10, 2003). CHI '03. ACM Press, New York, NY, 337-344.
16. The U-Scan Shopper: System Components and Functionality. http://www.kleverkart.com/html/system.html.
17. VanLengen, C. A. and Haney, J. D. 2004. Creating Web services using ASP.NET. J. Comput. Small Coll. 20, 1 (Oct. 2004), 262-275.
18. Internet Information Services. http://www.microsoft.com/iis