Service-Oriented Architecture for Mobile Applications

Yuri Natchetoi
SAP Research
Montreal, Canada
yuri.natchetoi@sap.com

Viktor Kaufman
SAP Research
Karlsruhe, Germany
viktor.kaufman@sap.com

Albina Shapiro
SAP Labs
Montreal, Canada
albina.shapiro@sap.com

ABSTRACT
Mobile phones are becoming a new popular platform for business applications. The number of mobile users increases daily and so does the need for efficient mobile data access and management. However, a traditional approach to business application and database design is not suitable for mobile devices because of the limited memory and connection bandwidth. This paper presents a novel lightweight mobile SOA-based architecture for business applications running on J2ME enabled devices such as cell phones. The paper includes position statement based on our experience and describes a first prototype implementation of the architecture. Some important features of our design are: using the knowledge of business processes to minimize data transferred to and stored on the device; pro-active data loading; allowing applications to fully function in a disconnected mode. The above architecture results in a lightweight framework, which can be used in order to develop a wide spectrum of business-oriented mobile applications.

Categories and Subject Descriptors
H.3.5 [**Online Information Services**] Web-based services

General Terms
Design, Experimentation, Human Factors, Performance.

Keywords
Mobile; framework; web service; service composition; lightweight architecture; pro-active loading.

INTRODUCTION
There are almost 3.3 billion connected mobile devices in the world, and this number is increasing daily. Cellular telephones dominate the global telecommunications market. Due to their massive popularity and flexibility, cell phones become equipped with hardware and software technologies such as J2ME, Bluetooth, Global Positioning System, digital cameras, and more - all at a greatly reduced cost. The wide use of mobile devices, low hardware costs and improved infrastructure make cell phones a logical and convenient access point for business applications.

The field of mobile applications and services continues to be one of the most rapidly evolving areas of communications. In the corporate world, the use of employee-purchased cell phones for both personal and business use also grows rapidly. On average, mobile employees spend one-third of their time out of the office, and almost half their time in the office away from their desks [18]. A growing number of mobile employees want to make business decisions using their cell phones. As a result, cell phones will play a critical role as the devices used for accessing enterprise systems, such as ERP (Enterprise Resource Planning), CRM (Customer Relationship Management), BI (Business Intelligence), etc.

Most business applications require performing significant amounts of data processing, either locally or through high-speed networks. However, currently most of existing cell phones cannot fulfill these requirements. The main obstacles that limit the development of business applications on mobile devices remain to be unreliable network performance and limited data storage. Multiple attempts to overcome the limited capacity of mobile phones have failed, because technologies used for desktop applications don't work well on mobile phones. Business data objects have to be requested from the back-end application and stored for processing on a mobile device, which has significantly lower resources than any enterprise system. At the same time, the mobile framework must provide a comprehensive user experience as well as a convenient application implementation model for developers. Hence, business application development for mobile devices requires an innovative approach to Web Service invocation, data exchange, transformation, and interfacing with the user. The basic requirements for such a mobile solution should include the following: 1) timely, robust and easy access to Service-Oriented Architecture (SOA) system, 2) transparency between connected, occasionally-connected, and disconnected modes, 3) loose-coupling system designed to combine services on demand, 4) lightweight application composition and development and, 5) low total cost of ownership.

In this paper, we propose a lightweight SOA-based architecture for mobile devices using the following techniques: 1) minimizing the amount of data transferred to and stored on the mobile device by using the knowledge of business processes and data access statistics to identify only the data required by the user, 2) a highly compressed XML format to transfer and store data, 3) reducing the amount of information contained in SOAP messages to increase efficiency of SOA-services invocation, 4) performing pro-active loading of data from the server, taking into account the client's service-invocation schedule, and 5) providing asynchronous connectivity to the back-end system, thus allowing applications to fully function in a disconnected mode. We argue that this set of techniques as a whole has not been considered sufficiently in publications, although they are necessary and/or facilitate meeting the above-mentioned requirements.

SAM'08, May 10, 2008, Leipzig, Germany.

In Section 1 we provide an overview of the existing SOA-based approaches for mobile phones. In Section 2 we describe the technique of Business Object pruning and serialization into a compressed RDF format that enables efficient transmission and storage of data. In Section 3 we introduce the overall lightweight Web Services architecture. Section 4 describes our security considerations. Section 5 focuses on a pro-active data-feeding that we use to optimize the Web Services performance on the mobile device. Section 6 summarizes the paper.

1. STATE OF THE ART

Mobile applications have been making use of Web Services for quite some time already. Companies like IBM, Nokia and Microsoft have made software available for enabling mobile devices to act as Web Service clients. There exist several development toolkits for Web Services on mobile devices. On the J2ME platform (mostly for Symbian based devices), JSR-172 is a widely used set of Web Services APIs [16]. The .NET Compact Framework for smart phone architectures also supports the use of synchronous and asynchronous invocation of Web Services [17]. Nokia Research is working with Mobile BPEL [23].

Web Services also have disadvantages when used in mobile applications [18]. Some examples are the verboseness of the frequently used XML and SOAP, lack of mature standards and mature support for transactions. There are also several performance issues for Web Services on mobile devices [19]. In our approach, we have devised solutions for dealing with such issues. The advanced topic of Dynamic Service Discovery and Composition in Mobile Environments has been so far addressed by relatively few authors. The related work can be found in [20].

SOA back-end access requires standardized data exchange. XML and SOAP are established universal flexible formats that enable publishing and exchanging heterogeneous data between business systems. Unfortunately, SOAP is very verbose and implies high performance costs. It is therefore difficult to use SOAP in situations where resources, such as communication channel bandwidth and storage capacity, are constrained. In these cases, a smart compression of XML files appears to be a solution, it reduces required bandwidth, storage, and processing.

The general approach to compression is a reduction of the information entropy of the message [10]. This can be achieved by using additional information known on both sides: receiver and sender. There are two types of data compression: lossy and lossless. A lossless compression is a method that ensures that the de-compressed message is identical to the original message. Most existing approaches use lossless compression, which takes advantage of a certain coding schema (e.g., Huffman encoding [4], arithmetic encoding [7]) to encode text and symbols in the original message in order to reduce its size. Lossless compression is context-free if the compressed message contains all the information required to recover the original message (e.g., [6], [8], and [12]). Lossless compression is context-dependent if the compressed message does not contain information that is already known by the receiver (e.g., [9]). Until now, only lossless compression was used in business applications.

Lossy compression cannot recover 100% of the original data from the compressed message. It reduces the size of the original message by discarding some information. If the information is never required on the recipient side or if the probability of using the information is very small, some loss of information is justified. Lossy methods provide a higher degree of compression and result in very small compressed messages. However, they must ensure that the loss does not affect the application functionality.

While lossless compression has been studied in many research projects, there has not been much work on lossy compression methods for XML. Cannataro [2] proposed a lossy compression algorithm for XML data, where each attribute value in the XML document is compressed by losing some precision, e.g., truncating strings, or replacing precise numbers by ranges. However, in most business cases, we believe it is important to keep the precision of the data, and it is not easy to determine which level of precision can be sacrificed. Alternatively, in the approach we propose, the precision of data is not altered. Instead, we sacrifice data that is never used on the client side.

2. BUSINESS DATA DELIVERY

When providing mobile access to existing enterprise back-end applications, one must be careful to distinguish between the traditional understanding of software applications and a specific functionality in a mobile enterprise software context. The core challenge is not to deploy the entire business supporting system onto a mobile device. Rather, the challenge is to transmit only the relevant business information and software functionality required for a specific process.

A business object, in general, is an entity of significance to a business. It acts as a service provider as well as a service consumer, and is exclusively accessible through a standardized set of services. Partial copies of business objects are used in our approach to create lightweight client-side replicas, which can then be accessed by the users of the mobile middleware. For the client side to perform efficiently, a lightweight business object management system must determine which data is relevant for the business functionality.

The business objects on the server side often have redundant attributes. Application designers try to make server-side business objects as generic as possible to minimize customization effort for each specific use case. Delivering such a complex object to the client often doesn't make sense, because some of the data fields will never be used on the client side. In every given case, the application designer can manually reduce the object complexity, see e.g. Figure 1.

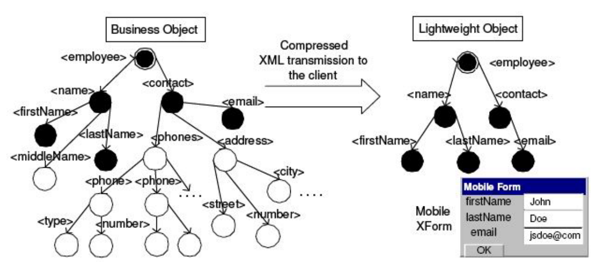

Figure 1. Semantic data pruning

Our goal was to find a generic approach that allows automatic pruning of the business object, only preserving the data that is required on the client side. In order to prune business objects, we need to have some knowledge of the object usage to determine which information will not be used on the client side. In our Framework, users can access business objects only through a (Web-) Services directory. Users can view and edit these objects through interactive tables and forms. In both cases, the list of data attributes (database table and columns) used on the client side can be clearly identified from the definition of the Web Services implemented on the client, compare Figure 1. This knowledge defines the specific information then requested from the general back-end Services. Furthermore, caching of information facilitates its immediate reuse, see Sections 3 and 5 for more information.

In order to transmit the business objects to the client side, we serialize each object to produce an RDF document. In [9], we have proposed a context-dependent XML compression technique that allows indexed data search and update without decompression of the XML file. The proposed XML compression is based on the knowledge of the business object data structure and the occurrence statistics. The principles guiding this approach are based on [5], where the authors claim that the information to be transmitted between the two systems may be reduced when considering the knowledge that both systems share. This shared knowledge may be at the pragmatic, semantic, syntactic or lexical level.

File type	*XSLT*	*WSDL*	*XForm*	*SOAP*	*SVG*	*RDF/ RSS*	*All files*
bzip2	0.551	0.196	0.490	0.564	0.449	0.454	0.426
EXEM	0.131	0.148	0.159	0.197	0.394	0.437	0.201
Gzip	0.501	0.178	0.445	0.499	0.432	0.417	0.393
Xmill	0.631	0.238	0.576	0.661	0.553	0.498	0.498

Table 1. XML Compression ratio comparison

By using the information about the data model (object description, DTD, XML schema) and XML lexical element occurrence frequency, we can significantly compress data by applying Huffman encoding to every context of the XML file such as a tag name, an attribute name or an attribute value. The lossless context-based compression allows not only to efficiently store RDF files, but also to search through the file and build search indexes without decompression. We have performed some experiments in order to compare compression ratio for different compression methods and different types of XML files. The results of our compression experiments, described in details in [11], are presented in Table 1. In the simple m-business application presented in [15] and in Figures 3 and 4, we have achieved binary XML compression ratio about 0.18.

3. MOBILE ARCHITECTURE

Based on the business cases we have analyzed, we have made the following assumptions about the mobile applications environment: 1) the online connection to the host is slow and expensive; 2) the mobile device has very limited local memory. Using these assumptions, we have implemented a lightweight mobile application framework design that provides an efficient SOA back-end connection solution with a minimal cost and hardware requirements.

In our Framework, the business objects are serialized, compressed and transmitted to the client side in the form of a compressed RDF message. The information is stored in the local Persistent Data Store (Fig.2) in a compressed RDF format, making it possible to store a significantly larger number of business objects as compared to a traditional file system or relational database. The pro-active Business Object Manager enables local service calls, updates on the Business Objects, and pro-active loading of data required later on in the business process. The client application uses this local data to support the off-line work.

The Data Connection Manager (CM and Data Parser in Fig.2), parses the incoming compressed SOAP files and builds indices on the Business Objects. They are passed to the Business Object Manager to be used to perform local operations on business objects. The cached objects are decompressed only when requested by an application.

Our Framework provides an asynchronous remote invocation mechanism to support on-demand requests of server-side Web Services for updating or inserting new data. The client sends update requests to the server as compressed SOAP messages. The server responds to remote calls from the mobile client as well as sends real-time alerts and data-updates notifications to the client side.

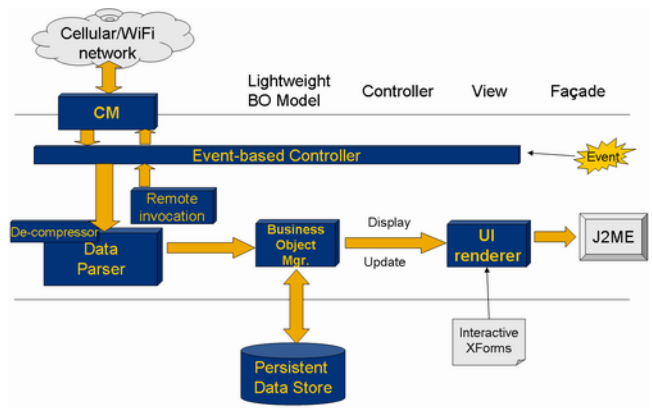

Figure 2. Mobile client Framework architecture

Providing efficient connection to the back-end Enterprise system is an important part of our lightweight architecture. First, an asynchronous, message-based communication is used as it is a better fit for a mobile environment in which the sender and the receiver are loosely coupled. Second, mobile devices make use of many different network channels with different capabilities. The Connection Manager selects the best communication channel based on availability and user's preferences. It then uses asynchronous communication to prevent unstable networks from causing long delays and canceling the entire transmission process. The Connection Manager can merge different communication channels into a single session. For example, if a mobile phone is in a close proximity to a computer, the Smart Connection Manager will use Bluetooth. However, when the user walks out of the Bluetooth range, the Connection Manager will transparently switch the session to a GPRS or EDGE channel.

Business data exchange is performed on demand, so that long offline phases are possible. In our Framework, business data exchange is implemented between original Business Object and its loosely-coupled replica, which resides on the client and communicates via asynchronous message exchange. Business objects are accessible only via published service interfaces, compliant with the Open Standards Gateway initiative (OSGi) [22]. We have designed our solution as a set of OSGi-compliant components. Communication through dedicated interfaces simplifies process modeling and makes the process flow transparent. The Framework supports both push and pull models for SOAP message exchange. The push model provides a better message delivery speed, but is sometimes difficult to implement. A pull model is simpler to implement and is better suited for low-priority messages.

We work with a Java midlet that asynchronously exchanges information with the server. In our solution, neither the business logic nor the user interface forms are hard-coded in the client application. Instead, the client application partially implements interpreters for open industry standards like SOAP, RDF, SVG, BPEL and Xforms, as well as uses open-source interpreters. The application logic and the user interface can be easily modified or augmented at low cost, since we are using standard formats. To further lighten the application, only subsets of the mentioned standards are used. For example, a very limited subset of the BPEL operations is implemented on the mobile device; this enables basic composition of the workflow scenarios using local and remote services.

The proposed SOA-based mobile architecture consists of Core Services and Composite Services, originally inspired by [24]. The Core Services provide a generic interface to business logic consisting of common building blocks. Composite Web Services account for more complex business processes and are implemented as a combination of core services. The core and composite Web Services are distributed across the client and an auxiliary proxy server and implement the business logic.

Figure 3. Sample m-business application using the Framework

We follow the ESOA approach and use the core services as common building blocks. The Web Services for Mobile Business Applications can be grouped into functional modules. Each module defines a number of Web Services related to a specific type of business or user role, etc. Our simplified implementation of the internal structure and provided interfaces of such modules will be validated at a later stage. We are working on identification of particularly useful interfaces and/or composite services that could be subject to standardization.

The back-end-access Web Service is represented by a set of mobile application components based on the J2ME technology. These components provide user access to back-end Services from ERP or CRM. The client application provides customizable user-centric access to the service functionality.

The mobile client is a set of OSGi components that expose their functionality as services published in the OSGi registry. Services in the OSGi registry are not Web Services, but a special proxy component can bridge remote Web Service invocations and make them look like local service calls to the mobile application components. The back-end-access component publishes the list of selected client Services available on the server in the local OSGi registry.

Future uses of our Framework can include: mobile notification service that shows a pop-up message on the user's mobile phone, complex collaboration scenarios and invocations of proprietary back-end APIs supplied by the service providers as Web Services. The server-side workflow scripts can also remotely call OSGi services implemented on the mobile clients using a Web Service interface provided by the middleware.

This approach will enable creation of widely distributed business processes that involve not only machine-based, but also user-performed operations. Because of the OSGi-based architecture of the mobile client, the new components can be easily created and integrated with Web Services and features available on the server side and on the mobile devices, such as phone camera, Bluetooth, GPS, etc.

4. SECURITY

One of the most important issues in every business context is security. The mobile business applications that provide access to the back-end enterprise systems should be secure and ideally provide the same level of data protection as desktop applications. All communication between the server and the client should be encrypted. In the case of mobile devices, there always exists an additional risk of the device being lost or stolen, and therefore all data stored on it should be encrypted as well. This requirement imposes an additional hardship on the secure data storage implementation.

In many business environments the public and private key infrastructure is already in place. Using an existing enterprise PKI would significantly simplify the implementation. However, a major problem is that the asymmetric key encryption scheme works very slowly on the devices with limited computational power, which is the case for all mobile devices. The compromise solution could be using asymmetric encryption for most sensitive information and symmetric key for all other information. The symmetric key could be generated once for the session and sent to the client using the asymmetric key. Following this approach, all the packets within one session between the client and the server will be encrypted with this key, except highly sensitive information such as, for example, payment transactions, which will still use the asymmetric key. In our design, we have decided to use an Elliptic Curve Cryptography [26] because it allows for much shorter public keys with the same encryption strength. A similar combination of the asymmetric and symmetric key infrastructure is used by the popular PGP[27] encryption system. The encryption feature is implemented as a replaceable module.

For more secure applications and policy regulated encryption, this module can be replaced by another component.

5. PRO-ACTIVE DATA FEEDING

An important user acceptance factor for occasionally-connected or loosely-connected applications is the ability to store and retrieve server information when the connection is down. A traditional reactive approach assumes that the information is stored in a cache only if it was already requested. In a proactive approach, a server application would be able to anticipate what information will be required next and push it into a cache. The key to improving the user experience is the ability to predict what information will be needed by a user in the near future and send it to the client when the connection is up but idle.

Regardless of the underlying caching technology, the only way to make the mobile applications framework viable is by proactively supplying the client with the maximum amount of relevant data possible. The majority of business operations have relatively formal transaction chains, allowing the server to analyze user transaction patterns, collect statistics and store them in individual user profiles. When the user is disconnected or has unstable connection, the server part of the application decides which transactions the user will most likely execute, and preload the client with the data required for these transactions.

In order to determine what data will be required by the client application in a multi-client component based mobile environment, we use the client demand forecasting model. We have already introduced a model for mobile client to prune server-side data objects. Some additional information can also help to find out the probability that the object will be used by the client application for a given user profile. The approach uses simple rules to calculate the probability of requesting a particular data object. Objects with the highest probability should be pro-actively pushed to the client. Typically, the business applications reflect business processes which contain multiple steps. However, the business process workflow is not always linear. Every step can trigger multiple further steps. A user can also use navigation to change the natural order of steps.

As an example we will consider a typical m-business application presented in [15]. The application contains many screen forms that display multiple data objects retrieved from the XML-based database. Our goal is to feed the database with those data objects that have the highest probability to be requested by a specific user.

We can describe the application as a graph of nodes where each node represents unique screen. A screen is a combination of the form and data objects that it displays. One form can display more than one object (or list of objects of the same type). For each object only a subset of its attributes should be displayed.

The transitions from one screen to another (navigation) can be performed by clicking different keys on the keypad. In every case we have more than one possible transition for navigating from any screen. We assume that every such possible transition has a certain probability of being chosen by the user. These probabilities can be different for different users. Many factors that characterize a user's behavior (personal interests, geographical location, price sensitiveness) can be organized as sets of user context variables.

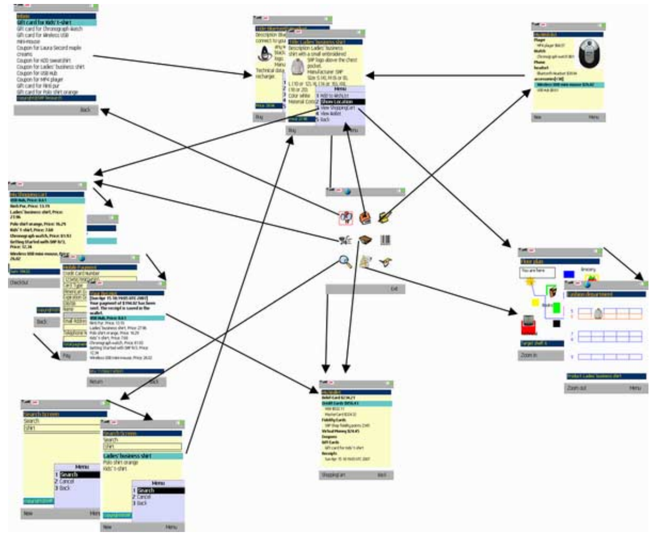

Figure 4. UI Screen navigation in our sample application

With the given probability of every step we can try to determine the probability of the next step which requires the data object to use statistical temporal models (e.g., hidden Markov models (HMMs), dynamic Bayesian networks (DBNs) or dynamic conditional random fields (CRFs)[21]). Authors have not tried all these techniques yet, but we are looking forward to perform a deeper research in this area.

So far we have only implemented a set of heuristic rules for determining the probability of the Business Objects to be requested by the user for a given application context. All data objects can then be sorted by the probability and the objects with the highest probability are pushed into a local database until there is free space. This simple algorithm allows optimizing local data storage load using the knowledge of the data to be requested.

Our goal is to build a model that will allow us to predict the consumption of existing business objects and appearance of the new objects. The demand prediction model based on the business object request probability can be easily converted into simple distribution rules. The distribution rules can in turn be generated automatically by the analyzer for each user profile and re-generated if the user role changes or application is being upgraded.

6. CONCLUSION

The enterprise solution vendors have yet to fully address the mobile enterprise applications market for cell phones connected to Enterprise Web Services. The innovative techniques employed by the Mobile Framework, as well as the pro-active user interface and the alignment with the SOA strategy prevalent in business applications, make the Framework a viable solution that merits further development and support.

A large portion of operations currently performed by desktop applications can be performed by mobile applications just as well. The Mobile Application Framework presented in this paper allows to overcome the main limitations imposed by the mobile environment: occasional connectivity and limited memory. The main advantage of our design is using the knowledge of the business object structure and data access statistics to store only the data required by the user. In addition, the Framework uses a compressed RDF format to efficiently exchange, store and query data, perform local querying on the indexed data, pro-actively preload relevant information from the server and allow for the application to function in a disconnected mode.

The methodology described in this paper provides a structured approach for the migration of the existing desktop applications to mobile devices by pruning unused parts and using compressed RDF files. This approach doesn't require intensive Java coding and significantly reduces application development and maintenance cost. It is compatible with all the latest industry standards and makes migrating existing business systems to mobile devices a feasible task. The world of technology is clearly moving towards wireless solutions and our mobile framework greatly contributes to this trend, allowing ubiquitous data access and enabling the advent of a new generation of mobile business applications.

REFERENCES

[1] A. Ankolekar, P. Hitzler, H. Lewen, D. Oberle, and R. Studer. Integrating Semantic Web Services for Mobile Access. ESWC, 2006.

[2] M. Cannataro, G. Carelli, A. Pugliese, and D. Sacca. Semantic lossy compression of XML data. In Knowledge Representation Meets Databases, 2001.

[3] B. Choi. Document decomposition for XML compression: A heuristic approach. In DASFAA, number 3382 in LNCS, Singapore, 2006.

[4] D.A. Huffman. A method for the construction of minimum-redundancy codes. In Proc. of the I.R.E., 1952.

[5] P. Kropf, G. Babin, and A. Hulot. Réduction des besoins en communication de CORBA. NOTERE'98, Montréal, Canada.

[6] H. Liefke and D. Suciu. XMill: An efficient compressor for XML data. In Proc. of the ACM SIGMOD Int'l Conf. on Management of Data, pages 153–164, 2000.

[7] David J.C. MacKay. Information theory, inference and learning algorithms. CUP, 2003.

[8] J. K. Min, M. J. Park, and C. W. Chung. XPRESS: A queriable compression for XML data. In Proc of the ACM SIGMOD Int'l Conf. on Management of Data, 2003.

[9] Y. Natchetoi, H. Wu, and G. Babin. A context-dependent xml compression approach to enable business applications on mobile devices. Euro-Par. 2007.

[10] C.E. Shannon. A mathematical theory of communication. Bell Syst. Tech. Journal, 27:398–403, 1948.

[11] Y.Natchetoi, H.Wu, G.Babin, S.Dagtas EXEM: Efficient XML Data Exchange Management for Mobile Applications, Information Systems Frontiers ISF 2007

[12] P. M. Tolani and J. R. Haritsa. XGRIND: A query-friendly XML compressor. In IEEE Proc. of the 18th Int'l Conf. on Data Engineering, pages 225–234, 2002.

[13] A.Arion, A.Bonifati, I.Manolescu, A. Pugliese XQueC: A Query-Conscious Compressed, XML Database, Proceedings of the 2004 International Conference on Extending DataBase Technology

[14] C.J. Augeri, B.E. Mullins, D. A. Bulutoglu, R. O. Baldwin, L. C. Baird An Analysis of XML Binary Formats and Compression

[15] H.Wu, Y.Natchetoi Mobile Shopping Assistant: Integration of Mobile Applications and Web Services, Proc. WWW 2007

[16] Sun Microsystems. J2ME Web Services Technical White Paper, July 2004.

[17] Microsoft Developer Network (MSDN). Consuming Web Services with the Microsoft .NET Compact Framework, March 2003.

[18] Bradford C. Johnson, James M. Manyika, and Lareina A. Yee. The next revolution in interactions. The McKinsey Quarterly 2005 Number 4

[19] D. Schall, M. Aiello, and S. Dustdar. Web Services on Embedded Devices. (2006) International Journal of Web Information Systems 2(1):1-6.

[20] Wolf-Tilo Balke, Jorg Diederich. A Quality- and Cost-based Selection Model for Multimedia Service Composition in Mobile Environments. icws, pp. 621-628, 2006.

[21] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data. 18th International Conference on Machine Learning, 2001

[22] OSGi. http://www.osgi.org/.

[23] L.Pajunen, A.Ruokonen, Modeling and Generating Mobile Business Processes, IEEE International Conference on Web Services (ICWS), Salt Lake City, Utah, USA, July 2007

[24] Ecospace IP. http://www.ip-ecospace.org/.

[25] OASIS Open: OASIS web services business process execution language (WSBPEL)TC. (2006).

[26] M. Smith M.D. Malan, D.J. Welsh, A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In Sensor Communications and Networks, 2004.

[27] Hankerson D. Hernandez J.L. Kirkup M.Menezes A. Brown M., Cheung D. PGP in constrained wireless devices. In 9th USENIX Security Symposium., 2000.