Push technology automates the information delivery process by not requiring users to request for the information they need. Wireless has experienced explosive growth in recent years, and ‘push’ will be the predominant wireless service delivery paradigm of the future. A wide variety of services, alerts and messages, such as promotional content will be deliver to consumers’ phones or PDA. However, to push information to wireless device can be a challenge because of the problem of intermittent communication links and resource constraint on wireless devices as well as limited bandwidth. This paper explores an efficient multicasting mechanism that ‘pushes’ pre-specified information to groups of wireless devices. The mechanism operates with limited bandwidth. Furthermore, it also overcomes the connectivity problem. Based on the above concept, we have designed and implemented a system to multicast sales information via wireless technology. The system is message-oriented and JMS compliance.

Keywords
Mobile commerce, multicast, wireless devices, push technology, publish/subscribe model

1. Introduction

Today the vast scale and scope of current online information sources on the Internet makes it difficult to find and process relevant information. Finding a specific piece of information on the Internet requires time-consuming searches. Hence, automation to push pre-specified information to the user seems like the next logical step to solve these problems.

We are moving towards third generation wireless technology where wireless handheld devices and hand phones support multimedia applications. It is believed that ‘push’ will be the predominant delivery methodology in wireless devices services because the problem of servers unable to push data to clients who are disconnected would not be an issue in GPRS wireless network. GPRS wireless network provides users with constant connection to the Internet. Hence, we can expect a large number and a wide variety of services, alerts and messages, such as promotional content, to deliverer to consumers’ mobile phone or PDA in real time. In addition, there are some constraints in wireless technology, such as the small memory capacity in the devices, limited bandwidth and the high cost of information searching on the wireless network. Push information to wireless handheld devices will save a great deal of time and money comparing to surfing the Internet via WAP technology. This brings forth the idea to create a ‘wireless push’ channel to push information to wireless devices in real time. This will constantly keep the mobile user informed – in real time. This provides more information for making decision.

Furthermore, from a software point of view, in order to cope with the limited bandwidth problem, this research will study an efficient multicasting mechanism to push sales information to a group of members over the wireless
network. We propose a framework and application on Internet selling process whereby the sales information like product catalog will be multicast to the interested mobile users in real time. Publish/Subscribe messaging model is the underlying message delivery mechanism. The model provides the capability of multicasting information to a specific group of recipients.

2. Phenomenon of Push Technology

Push technology implemented on the Internet is not a new idea and there was a tremendous wave in 1990s. Push technology automates the information delivery process without requiring users to request for the information that they need. The browser that we are familiar is a kind of pull technology in that we must request for a web page before we see it. The significant and prominent examples of push implementation are in broadcast media like radio and television where the information is send out whether anyone is receiving it or not. Push on the Internet means having Web material gathered and sent to them, instead of a client program having to request for it.

Currently, there is a glut of information because the publication of web pages is so easy. The ratio of useful information to useless information is too low. With terabytes of information on the web, we need information filter to provide meaningful information to user in a timely and regularly manner. Those who are pushing information should deliver the information that a user is interested in straight to the user, allowing the user to bypass the millions of irrelevant websites. However not every type of information should be delivered via the push mechanism. The following three criteria must be satisfied in order for a piece of information to be considered suitable for delivery using push mechanism:

- The kind of information desired must be known ahead of time like stock quotes, new headline etc.
- Searching for such information must be an inefficient use of the user's time.
- The user must want this information regularly.

The following points briefly illustrate some of the more common “push” implementations:

- Automatically deliver new or updated product information to customers and prospects.
- Notify a salesperson if their number of opportunities drops below a specific threshold.
- Distribute corporate policies and procedures.
- Distribute market analyses and trend reports.
- Keeping business partners informed about customer service calls and product updates.
- Distribute product literature to business partners.
- Notify customers about the changes to their accounts.

2.1. Data Delivery Mechanism

The paper “Data In Your Face: Push Technology in Perspective"[12] presents some ideas on data dissemination in order to provide a framework for thinking about push technology. The authors have outlined several options for data delivery and the comparison of their characteristics are:

- **Client Pull vs. Server Push** – pull based is request-response operation, which is client initiated transfer of information from server to client. On the other hand, the server initiates the transfer in ‘push’ operation.
- **Aperiodic vs. periodic** -- Aperiodic is an event-driven operation, where event triggers the transfer of information. In periodic delivery, information transfer operations follow pre-arranged schedule.
- **Unicast vs. 1-to-N** -- With unicast data communication, the data to be transferred is sent from one source to one destination. While multicast and broadcast are 1-to-N data communication. In multicast data delivery
mechanism, the transfer data is send to specific subsets of clients. Whereas in broadcast data delivery mechanism, the transfer data is send to an unidentified set of clients that can listen to it.

Data dissemination using periodic push is use in many systems (for example Internet mailing list that regularly send out mail to users). Aperiodic push based on publish/subscribe protocol is becoming a popular way to disseminate information to end-users.

2.2. Pro and Con Push Technology

2.2.1. Benefits of Push Technology

The paper “‘Push’ Technologies: Reborn for Business”[11] summarized the benefit of push technology into one sentence:

"It's mirroring what a really good human assistant would do if all they had to do was sit around and watch out for you."

Push technology has advantages for both end users and content providers. For end users, it significantly improves the response time of accessing Web content. Clients do not have to waste cycles and network traffic to poll servers.

Content providers can target their audiences in a more direct way, which results in a cleaner business model. Servers can use more processor time for data production rather than to process numerous client requests and send much data over the network. Furthermore, servers can better manage the amount of data transferred over the network as it just delivered useful and interested information to clients.

2.2.2. Drawbacks of Push Technology

There is no one perfect technology in this world. Obviously one of the drawbacks of push technology implementation is the requirement for client to be online all the time. Servers cannot push data to clients who are disconnected. This would not be an issue in GPRS wireless network since users are always connected. Besides, push servers need to maintain huge online active connections as well as clients’ status. The servers need to hold the session for every active user and this will consume memory space for keeping track of live connections.

The other downside is that “push” data delivery mechanism consumes a lot of bandwidth if the data is transmitted in “unicast” mode rather than “multicast mode”. In addition, “push” server is potentially exposed to security problems if the clients just listen to the incoming message without doing proper filtering or scanning. New protocols or changes in the exiting protocols are required to make push technology secure. This will enhance users’ trust.

2.3. Constraint and Challenges in Push Technology

“Data In Your Face: Push Technology in Perspective”[12] has identified some issues regarding push technology. Current web casting applies pull technology instead of push due to limitation of HTTP protocol. HTTP protocol is a kind of pull-based protocol. Broadcasting and multicasting are not widely used as it potentially causes bandwidth problems if multi-clients request for the same data through unicast data communication. In addition, the article “What’s All Wrong With Today's Push Technology”[5] has pointed out that push technology should be built on top of multicast data communication. It is not advisable to implement unicasting in push technology, which means sending information one copy to one user at a time as it will eventually outstrip the capacity of networks and clog the Internet. Furthermore, the article “The push technology rage... so what's next”[15] also pointed out the next thing of
push technology is multicasting delivery mechanism which is to enable true push technology to millions of users across the Internet. The Internet still uses the basic one-to-one ratio of one request for information sent to one computer at a time. Therefore Internet service providers (ISPs) will have to update their networks to handle multicasting, which will enable content providers to “broadcast” data to a large number of users rather than sending content using one-to-one model. Furthermore, this article has highlighted a need for standard message in pushing technology as a common way to tell users what is in a content channel and what the users need to view it. This article discusses XML as the standard of W3C.

“When push comes to shove: Push technology is all the rage -- What does this mean for Java?” [6]. This article has discussed about "webcasting" or "push" technology that can dramatically increase the productive of the Internet or Intranet. However, when allowing someone to shove software components onto your computer, security becomes an issue.

In short, the main challenges in push technology are:

- Avoid clogging of network traffic by multicast or broadcast information to those users who are interested.
- Standardize the push message to all channels especially to push information to different platform or devices, like various manufacturers’ wireless phone, Palm OS, IPAQ, Visor etc.
- Security will be an issue when user allows downloading of information or software component onto his local devices.

3. J2ME vs JMS Middleware in Wireless

The importance of mobile phones in consumers’ every day lives will continue to increase as the number and diversity of available services grows. Consumers will increasingly demand easy access to information whilst on the go in a timely and cost effective manner. The introduction of Java™ technology enabled phones will open many more possibilities and provide a major impetus for the expected growth. In order to cope with this growth in mobile devices, currently several technologies are use to develop wireless applications like Sun’s Java™ 2 Micro Edition (J2ME™) and Qualcomm's Binary Runtime Environment for Wireless (BREW). J2ME has emerged as a standard for developing wireless application with Java.

3.1. J2ME

J2ME is the newest addition to the Java family and provides a mechanism for delivering Java solutions to resource-constrained consumer and mobile devices. J2ME has not designed specifically for a target device. It is a standard specification for various mobile devices. J2ME model provides three separate layers: Profiles, Configurations, and Java Virtual Machines. The Configurations and Profiles layers customize the Java Runtime Environment to suit different kind of mobile devices. These layers sit on top of the host operating system. The following paragraphs described the top two layers.

3.1.1. Configuration

A configuration defines the basic J2ME Runtime Environment as a virtual machine and a set of core classes that run on a family of devices that have similar capabilities. Two configurations are currently defined: the Connected Limited Device Configuration (CLDC) and the Connected Device Configuration (CDC). The former targets at devices that have extreme limitations when compared to the memory and power requirements of a conventional Java implementation such as cellular phones, two-way pagers, and organizers.
3.1.2. Profiles

A profile addresses the specific demands of a certain “vertical” market segment or device family. A profile’s goal is to define application-programming interfaces (APIs) for devices that have similar uses. A device can support only one configuration but multiple profiles as long as each of those profiles uses the same configuration as its foundation.

3.2. Message Oriented Middleware – JMS

Middleware means software that enables communications between the distributed components of an application. Two models of middleware is defined in terms of communications architecture in distributed application (Figure 3.1):

- Remote Procedure Call (RPC)
  This is the model of CORBA, RMI, and DCOM. The communication between distributed components is synchronous and tightly coupled.
- Message Oriented Middleware (MOM)
  The communication between distributed components is asynchronous and loosely coupled. The respective programming model is call messaging. Message Oriented Middleware (MOM) is a software that acts as transporter for transporting messages from a source component (originator) to target components (recipients), usually running on different computers and distributed application. They allow separate business components to combine into a reliable and flexible system. Java Message Service (JMS) is the MOM standard for Java. JMS has two messaging models, Point-to-Point and Publish-and-Subscribe messaging, but both use exactly the same message format.

![Figure 3.1: The Distributed Components: Tightly Coupled vs. Loosely Coupled](image)

3.3. Publish-and-Subscribe Messaging (Pub/Sub)

The Pub/Sub messaging model is where there is only one sender and one or more receivers. Rather than queuing for sending and receiving messages, there is a topic, which is equivalent to an individual newsgroup. Interested clients subscribe to receive all messages sent to those topics and they can publish to those topics if they wish to. The number of publishers and subscribers grow and shrink over time. Figure 3.2 illustrates the process of Pub/Sub messaging.
where “P” is publisher and “S” is subscriber. This model is suitable for push model application as consumers are delivered messages without having to request for them.

3.4. Wireless JMS

JMS provides an asynchronous and message based transportation. Using message queues hosted on both the client and the server side, JMS applications can be operated in disconnected mode and data synchronization occurs transparently and immediately without user intervention. In addition, it provides an ideal abstraction layer for developing mobile applications and increases the scalability of system. Many mobile devices can simultaneously send messages to a server. When messages arrive at the server, they join an inbound queue and can is dealt with when resources are available, or can be forwarded to other servers for load sharing. In practice, JMS allows wireless services to operate more responsively, to recover from sporadic network outages easily, and to allow mobile applications to be operated offline. Finally, JMS messaging sits elegantly atop Bluetooth, Wireless LAN, GPRS, UMTS, and Mobitex etc.

3.5. Ibus

Softwired, based in Zurich, Switzerland, is the leading supplier of highly scalable e2e™ JMS technology for Intranet, Extranet and Internet as well as end-to-end Wireless and embedded solutions. The following are Softwired’s product that we used in our system: iBus//MessageBus, iBus//MessageServer, iBus//Mobile.

iBus™ is an industrial-strength publish/subscribe and point-to-point enterprise middleware written entirely in Java. It is fully compliant to JMS standard. It allows application developers to quickly build robust, scalable and distributed applications that can share data atop of any transport protocol, such as reliable IP multicast, TCP, HTTP, SSL or various wireless media. iBus//MessageBus is the only reliable IP multicast-enabled JMS middleware on the market today. iBus//MessageBus fully utilizes the benefits that IP Multicast offers. The system links publish/subscribe topics to individual IP multicast groups. This is achieved either automatically or controlled via administrative action.


In this section, we first discuss the design principles of our information dissemination system for wireless devices using multicasting messages based on publish/subscribe model - such a system must overcomes the constraints that are posed by the characteristics of wireless devices. Detailing the design architecture, message delivery mechanism, and the internal workings of the resulting system follows this. We also provide a few snapshots of screens to further illustrate the system functions. This section ends by discussing some issues that we faced during the development process.
4.1. Mobility Issues

In this section, we will look at the mobility issues of developing mobile applications. Mobile applications present many challenges for software developers and require them to deal with the issues not present in wire line system. The following summarizes the issues:

- Intermittent communication links.
  Mobile devices often lose network coverage. To deal with this intermittent communication links, the mobility application should provide disconnected operation and guarantee important data reaching mobile devices.
- Resource constraints.
  In order to fit into small ROM and RAM footprints as well as for low usage of CPU cycles and battery power, mobile applications are aggressive optimized.
- Multiple bearers.
  Currently there are many different bearers in wireless network like SMS, GPRS, Infrared, Bluetooth and HTTP. Develop one mobile application and run in different bearers become the challenge of developer.
- Multiple platforms.
  There are various platforms in the mobile communicator device market such as Palm OS, Pocket PC, Symbian, and Windows CE etc. It is advantageous to develop applications in such a way that they are able to run on various platforms, without requiring substantial modifications.

Building mobile applications using a middleware with JMS implementation can solve most of these issues and problems. iBus//Mobile JMS middleware has been selected to prove the concept on this paper. The following sections will elaborate on the implementation details of the application model of “Push Selling - Multicast Message to Wireless Device Base on Publish/Subscriber Model”.

4.2. Architecture Design

Figure 4.1 illustrates the application framework. This ‘push’ based application implements Internet selling process whereby the product catalog created by the seller will be multicast to mobile users. The proposed model mainly comprises of three types of nodes:

1. **Host** provides the base data for dissemination. In our example system, we assume the core business of the system is to sell health care products and services. The system will disseminate the products’ information in the form of catalog. The system provides a GUI for creating the product catalog. This GUI is a JSP (Java Server Page) interface which running on top of Tomcat Servlet engine. The “Information broker” agent creates and sends out product catalogs.

2. **Information Broker** plays a vital role in acquiring information of product catalog, adds on additional value or data to that information, and then pushes this information to clients. Information Broker runs on top of iBus//MessageServer and iBus//MessageBus. In the event of this broker is being triggered, it will create a topic and all users who are subscribing to this topic should able to receive the messages. This broker behaves like “Publisher” to multicast the product catalog to those active subscribers. Publish/subscribe model enables data/information delivery to a specified set of interested clients. In addition, this node also keeps and maintains the clients’ sessions and configuration data.

3. **Clients** are net consumers who will receive such selling information without requesting for it via iBus//Mobile. iBus//Mobile provides a gateway for transferring the messages to wireless devices. From the
JMS provider's point of view, the gateway is a regular JMS client. From the mobile device's point of view, the gateway is a communications hub and message format translator. Light-weight JMS Client Library is used to develop the MIDLET application. iBus/Mobile provides the library. The MIDLET will be loaded in client’s PDA and runs as client’s agent to receive the incoming push information. PALM operating system is the platform for MIDLET application.

**Figure 4.1: Architecture Design Of Push Selling-Multicast Message To Wireless Devices**

### 4.3. Message Delivery Mechanism – Publish/Subscribe Model

Publish/Subscribe model is the basis for building the message delivery mechanism of this system. Figure 4.2 illustrates this design. The transfer data is ‘pushed’ to specific subsets of client. The following describes the message flows:

- Once a seller has created a product catalog, the system will trigger a Servlet to send this product catalog in the form of message to Information Broker.
- The Broker will create a topic, named it as “push_info”. In order to receive this message, the mobile users need to subscribe to this topic.
- The Broker will start publishing its first message, then iBus middleware will start sending the message to all subscribers. The middleware ensures that all subscribers receive message of this topic "push_info".
- The Broker can publish another message before the previous message has been ‘pushed’ out to all subscribers.
4.4. Sequence of events

The following sequence diagrams illustrate the interactions occurring between a mobile device and a JMS middleware. Figure 4.3 shows the sequence diagram for the seller and publisher that perform the Create Product Catalog and Push Messages use cases realization. The seller uses UI provided in JSP to create a new health care product catalog. The seller can create multiple items per catalog. Then the JSP will forward the catalog message to a Servlet and the Servlet will trigger the Information Broker, which acts as publisher. The publisher will create a topic and starts pushing the message to all subscribers.

Figure 4.4 shows the sequence diagram for the subscriber that performs the use case of Receive Message and Display Message realization. The subscriber uses the Initial Subscribe UI to subscribe to the topic. Besides keying the topic information, the subscriber needs to enter the URL that point to the Server. Then the subscriber starts listening to the incoming message. If there is a message, the system will display the message in the screen of mobile devices.

Figure 4.3: Sequence Diagram For The Publisher

Figure 4.2: Publish / Subscribe Messaging In 1-N Push Technology
4.5. Result

This section presents the results of the implementation as a proof of concept. Figure 4.5 shows a User Interface for sellers to create their product catalog. The GUI is developed in JSP and running in Tomcat Servlet Engine. The page consists of several input textboxes for user keying the catalog information like product code, product name, product description and the price. The seller can create multiple items per catalog by clicking on “Add Product” button. The seller publishes the catalog under a specific topic by keying the topic name at the bottom of the page. Once “Create Catalog” button has been clicked, the system will create a product catalog and the page will trigger a Servlet known as “triggerPushAgent”. The Servlet writes the catalog message to a file called “productcatalog.txt” and then it will instantiate the Information Broker. The broker will read the message from that file and multicast message to all subscribers.

Figure 4.6 shows the login interface for client application in Palm OS Emulator. The subscriber needs to key in the necessary data as illustrated below:

- Protocol: Socket means TCP connection, while Datagram means UDP connection.
- Host: Host name or IP address points to Gateway Server.
- Port: The port number of the Gateway.
- Topic: Topic name he wishes to subscribe. His device will then receive all messages under this topic.
- Session ID: Session ID must be uniquely defined and will be kept and maintained by the Gateway Server. If desired, unique session ID can be dynamically generated by the system.

Once the subscriber logs on to the system, the application will be in listening mode and waiting to receive incoming message. Figure 4.7 shows the palm device in listening mode.

Figure 4.8 shows the two subscribers: User001 and User002 have received the product catalog.
Figure 4.5: User Interface For Creating Product Catalog

Figure 4.6: Client’s Application – Login User Interface
4.6. Issues and Problems

We implemented the system components discussed so far on Pentium III, 256MRAM PC platform. The platform carries the components of iBus/MessageServer, iBus//MessageBus, iBus//Mobile, Tomcat Servlet engine and emulators of Palm OS. Since all components are running in single machine, these have incurred the performance issue during testing. The second and subsequent emulators would receive the message with slight delay. The latency is about one second. In addition, another reason that might cause the latency is the TCP/IP communication protocol that is between Palm OS and Gateway. Socket connection will introduce some delay, as the system will send the
message to all subscribers and waiting for acknowledgement back from remote handheld devices. Therefore the multicast message mechanism is only able to achieve high performance provided the system is implemented in a network and Internet Service Providers (ISPs) that provide multicast-enabled routers and gateway across their network and Mobile IP system.

5. Recommendations and Future Works

Although the developed system is able to multicast messages to a specific subset of clients who have indicated their interest in receiving the selected information, the current wireless network is limited by the transmission protocols of TCP, UCP, HTTP, WAP etc. Thus, the communication layer is still transmitting data in “unicast” mode, this can potentially cause bandwidth problems if the system multicasts messages simultaneously to thousands of receivers. In order to enable true multicast messages, network upgrade is necessary - that means installing multicast-enabled routers across the networks. Furthermore, a research on IP multicast for mobile hosts (especially in wireless environment like mobile IP) is also necessary to enhance the system and to realize true multicasting.

Besides multicast delivery mechanism, we also recommend a study on broadcast delivery mechanism. Broadcasting is a form of push based data delivery mechanism. The basic idea is the server periodically broadcasts the sales information, while client agents monitor the broadcasted messages and only receive messages that they require. The advantage of this approach is that other clients who are also monitoring the broadcast do not directly affect the performance of any other client receiving data from the broadcast.

Furthermore, we are also researching into using mobile agents in wireless handheld devices because they will overcome wireless network limitations such as low bandwidth and intermittence disconnection problem. Agents can act as buyers or sellers and they communicate and negotiate with each other autonomously.

6. Conclusion

This push-based selling system enables mobile users to receive products information in real-time. The crucial element of the system is that the sender assigns messages to topics and not to a particular remote object and the receivers couple the messages based on the topics. The system provides a company as content provider to simultaneously push its selling information to multiple handheld devices such as PDA. Therefore, from the perspective of sender, it is more effective and efficient in transmitting large amounts of data to a group of receivers in real time. No additional user interfaces or efforts are required in selecting or customization the group of receivers. In comparison, conventional delivery mechanisms like email require the sender to know the receivers’ email addresses before sending the information. The latter requires extra efforts in keeping and maintaining clients information.

Furthermore, from sellers’ perspective, it allows for targeting their audiences in a more direct way, which results in a cleaner business model. The server can better manage the amount of data transferred over the network as it just delivers useful and interested information to clients. This significantly enhances scalability and uses the available network bandwidth with maximum efficiency.

From mobile users’ point of view, they do not need to spend a lot of time and money in order to get information they want as the information is automatically ‘pushed’ into their wireless handheld devices.

To conclude, the system overcomes the problems of current push technology in commercial activity. A framework is proposed and a trail application developed to multicast commercial information to wireless device where the delivery mechanism is base on publish/subscribe model. This application enables sellers to multicast their product information to a group of interested users in real time. It also allows the sellers to target their audiences in a more direct way
rather than waiting for audience to search for the relevant information. Furthermore, the system do not need to maintain the clients’ profiles because the message will be able to reach the audiences as long as the audiences keep noticed of the gateway address and subscribe to their interested topics.

**Acknowledgment**

We gratefully acknowledge Softwired Inc. for providing us with Ibus Gateway, Ibus Server and Ibus Mobile.

**References**


[14] Dharap, C., and Freeman, M., Information Agents for Automated Browsing, ACM Digital Library, 296-305,


[32] Talking Point, ChatterBox Whitepaper: The Agent Infrastructure For Wireless Portals,