

White Paper

---

**Sprint Enterprise Application Messaging <sup>SM</sup>**

**A Reference Architecture and Framework**



June 2, 2003

**An Insight Into Technology**

iQknowledge

*Prepared by:*  
**InQuest Corporation**

*Sponsored by:*  
**Sprint Mobile Computing Services Group**

© InQuest Corporation

## Table of Contents

<b>TABLE OF CONTENTS</b> .....	<b>ii</b>
<b>EXECUTIVE SUMMARY</b> .....	<b>1</b>
<b>1. INTRODUCTION</b> .....	<b>2</b>
<b>2. ENTERPRISE REQUIREMENTS</b> .....	<b>3</b>
<b>3. EXTENSIBILITY TO BUSINESS APPLICATIONS</b> .....	<b>5</b>
3.1. CLIENT REQUIREMENTS.....	7
<b>4. MOBILITY</b> .....	<b>9</b>
4.1. CONSISTENCY IN APPLICATION MANAGEMENT AND DELIVERY .....	10
4.2. MOBILE PRESENCE AWARENESS.....	10
4.3. GUARANTEE AND RELIABILITY OF MESSAGE DELIVERY .....	11
4.4. MOBILE SECURITY.....	11
4.5. APPLICATION WAKEUP .....	12
4.6. VOICE MESSAGING.....	12
<b>5. DIRECTORY INTEGRATION AND ACCESS CONTROL</b> .....	<b>12</b>
<b>6. ARCHIVING, LOGGING, AND COMPLIANCE</b> .....	<b>13</b>
<b>7. REPORTING AND ANALYSIS</b> .....	<b>14</b>
<b>8. MESSAGE ENCRYPTION</b> .....	<b>14</b>
<b>9. POLICY MANAGEMENT</b> .....	<b>15</b>
<b>10. VIRUS PROTECTION</b> .....	<b>16</b>
<b>11. RELIABILITY, AVAILABILITY, AND SCALABILITY</b> .....	<b>16</b>
<b>12. INTEROPERABILITY</b> .....	<b>18</b>
<b>13. A REFERENCE IMPLEMENTATION</b> .....	<b>19</b>
<b>14. CONCLUSION</b> .....	<b>22</b>

---

**Highlights**

---

## Executive Summary

The term *application messaging* is used to denote an instant messaging (IM) architecture that in addition to person-to-person communication extends to support real-time person-to-application as well as application-to-application messaging.

Application messaging extends the existing enterprise web and application architectures to simple text- or forms-based IM clients that could be used from almost any device, making it the “killer” application that wireless carriers, such as Sprint, have been looking for to introduce new customer-facing business solutions that complement their significant investments in next-generation wireless services.

Application delivery over a real-time messaging channel, with “*presence awareness*” is a strong proposition that enables alerting and collaboration functionalities that traditional web technologies alone cannot provide. Enterprises are just beginning to understand the reward of real-time messaging, communication, and collaboration in terms of user productivity and customer experience -- a reward that can be significantly increased once extended to business applications and mobility.

The rise in the ad-hoc use of Consumer Instant Messaging (CIM) in corporations has created a sense of urgency for IT to implement properly managed Enterprise Instant Messaging (EIM) solutions. This, in turn, has motivated development of EIM solutions by independent software vendors (ISVs), some of which have only recently been announced. Examples include Microsoft's® Real-Time Communication Server (RTC Server), AOL® Enterprise AIM® Services, Lotus® Sametime, and Yahoo!® Messenger Enterprise Edition. These EIM solutions extend real-time person-to-person instant messaging to the enterprise. However, in their rush to introduce EIM solutions, ISVs have limited their attention primarily to the same set of instant messaging features that has made CIM solutions a success, leaving a significant gap in the areas of enterprise *application messaging* and *mobility* – two areas where Sprint sees significant demand and opportunity.

As a wireless carrier, Sprint has recognized that the success of its foray into the application messaging market is contingent on the capability of its application messaging framework to integrate with third-party EIM solutions. While Sprint's application messaging framework can provide a complete real-time messaging solution for customers that are new to EIM, it does not require replacement of EIM solutions where they already exist. Rather, it complements the third-party products, particularly when extending application messaging to mobile devices. The flexibility to integrate other EIM solutions into its application messaging architecture has put Sprint on the path to forge new partnerships with companies such as Microsoft for delivery of comprehensive EIM, application messaging, and mobility solutions.

In this white paper we examine the Sprint Enterprise Application Messaging<sup>SM</sup> framework within the context of satisfying enterprise requirements. We also provide examples of how enterprise applications can be extended to real-time messaging clients to provide a universal look and feel to applications, whether accessed from a desktop or a pervasive device.

**Highlights**

***Application messaging allows for person-to-person, person-to-application, as well as application-to-application communication in real-time.***

***Application delivery over an instant messaging channel, particularly with its “presence awareness”, is a strong proposition that enables functionality that even the traditional web technologies alone cannot provide.***

**1. Introduction**

Sprint is a leader in the wireless telecommunications market and has invested heavily in next-generation wireless services. Currently Sprint is the only wireless carrier that provides a nationwide 1xRTT CDMA 2000 footprint that is capable of providing up to 144 Kbps always-on data communication from a PCS mobile device. While this is a significant achievement in the telecommunications landscape, it has presented a challenge to Sprint, as well as other service providers and Internet developers to creatively extend corporate applications to pervasive devices.

To address this challenge, Sprint has organized a team of talented software developers and consultants in its Mobile Computing Services (MCS) group. The focus of this group has been to extend enterprise applications to pervasive devices. Recognizing that corporate partners, employees, and customers present a heterogeneous wireless environment, the MCS group has focused its attention on solutions that are interoperable with other carrier networks as part of an overall Sprint solution.

The challenge faced by Sprint’s MCS group was to extend corporate applications to pervasive devices by making the experience consistent from device to device and without losing any of the following functionality:

- *Interrogation and query* – Discovering information (i.e. searching & web surfing)
- *Action* – Performing a specific task, such as updating a database.
- *Alerting* – Providing the ability to alert on key events.
- *Reporting* – Maintaining a record of past events and generating relevant reports.
- *Analysis* – Analyzing data to draw relevant conclusions.
- *Collaboration* – Sharing information with and garnering opinion from others.

Ideally, a technology had to be selected that would easily interface with existing web and application architectures through small and extensible clients that could be used from almost any device. The chosen solution was to extend the instant messaging technology to corporate applications -- giving rise to the Sprint Enterprise Application Messaging. Application messaging allows for person-to-person, person-to-application, as well as application-to-application communication in real-time. Application delivery over a real-time messaging channel, particularly with its “presence awareness”, is a strong proposition that enables functionality that the traditional web technologies alone cannot provide. Figure 1, provides a comparison of application messaging versus traditional web technologies with respect to the six enterprise application functionalities discussed earlier.

<b>Application Capability</b>	<b>Traditional Web</b>	<b>Application Messaging</b>
<b>Interrogation &amp; Query</b>	●	●
<b>Action</b>	●	●
<b>Alerting</b>	○	●
<b>Reporting</b>	●	●
<b>Analysis</b>	●	●
<b>Collaboration</b>	◐	●

Strong
  limited
  Weak

**Figure 1 – Comparison of Web and Application Messaging Functionalities**

---

## Highlights

---

***... application messaging not only matches the traditional web capabilities, but also closes the gap in areas of alerting and collaboration, where inadequacy of web applications is evident.***

***... companies are just beginning to understand the reward of instant messaging ... a reward that can be significantly increased once IM is extended to business applications and mobility.***

***Sprint has recognized that its foray into the application messaging market is contingent on one critical success factor – the capability of its application messaging framework to integrate with EIM solutions from independent software vendors ...***

***The flexibility to integrate other EIM solutions into its application messaging architecture has put Sprint on the path to forge new partnerships with companies such as Microsoft for delivery of comprehensive EIM and application messaging solutions.***

It is clear that application messaging not only matches the traditional web capabilities, but also closes the gap in areas of alerting and collaboration where inadequacy of web applications is evident. The strength of application messaging in these areas is directly correlated to the *real-time* messaging and *presence awareness*. Each user or application that is connected via an application messaging server is aware of the presence of others in real-time. This facilitates real-time alerting and collaboration.

It is also interesting that instant messaging is rapidly displacing e-mail and telephone for daily communication. Gartner Group estimates that by 2005 instant messaging could surpass e-mail as the world's preferred electronic form of communication. Gartner also estimates that by 2005 over 50% of business applications used to interact with customers will be integrated with instant messaging. Many companies are just beginning to understand the reward of instant messaging in terms of user productivity and customer experience -- a reward that can be significantly increased once IM is extended to business applications and mobility. If implemented and managed correctly, IM and application messaging increase a company's ability to monitor and manage the performance of its team members, applications, and services and will significantly improve business productivity.

Sprint has recognized that its foray into the application messaging market is contingent on one critical success factor – the capability of its application messaging framework to integrate with EIM solutions from independent software vendors such as Microsoft's Real-Time Communication Server (RTC Server), AOL Enterprise AIM Services, Lotus Sametime, and Yahoo! Messenger Enterprise Edition. While Sprint's application messaging framework can provide a complete messaging solution for customers that are new to EIM, it does not require replacement of existing EIM solutions. Rather, it complements the ISV products, particularly when extending application messaging to mobile devices. The flexibility to integrate other EIM solutions into its application messaging architecture has put Sprint on the path to forge new partnerships with companies such as Microsoft for delivery of comprehensive EIM and application messaging solutions.

We believe that in its quest to find a “killer” application for mobile devices, Sprint has created a solution that not only incorporates mobility, but also forms the foundation of an “executable Internet” that brings people and applications together in real-time.

## 2. Enterprise Requirements

Before discussing Sprint Enterprise Application Messaging architecture and framework, one must understand the distinct business requirements that differentiate a consumer instant messaging platform, such as AOL, MSN, ICQ, or Yahoo from an enterprise instant messaging platform suitable for application delivery.

The key enterprise requirements for application messaging include:

- Extensibility to business applications and ease of integration
- Mobility
- Directory and infrastructure integration
- Archiving, logging, and compliance with regulatory requirements
- Reporting and analysis capabilities

---

## Highlights

---

***... companies are finding CIM platforms penetrating their IT landscape through informal installation and use of CIM clients by employees.***

- Message encryption
- Ability to support policies for access and usage
- Virus protection
- Interoperability with customer and partner EIM solutions
- Scalability, reliability, and availability
- Centralized administrative controls
- Billing and/or usage tracking

With the exception of “scalability, reliability, and availability”, none of the other enterprise requirements are satisfied by any of the consumer IM platforms. Nevertheless, many companies are finding consumer IM platforms penetrating their IT landscape through informal installation and use of CIM clients by employees.

This ad-hoc intrusion of CIM in the enterprise could result in serious consequences. For example, while corporate e-mail remains on corporate servers and never leaves the enterprise network, a conversation over a CIM network requires that unencrypted messages be sent to the CIM server, hosted by a service provider, and then forwarded to the intended recipient. This holds true even if both users are within the same enterprise and protected by the same firewall. Once the conversation leaves the corporate firewall, it could become subject to eavesdropping. This clearly violates the regulatory requirements imposed by HIPAA for healthcare and GLBA and the Patriot Act for financial institutions. It may also jeopardize sensitive corporate information for others.

***... unavoidable urgency to manage real-time messaging, if addressed within the context of a cleverly designed framework, can help create the infrastructure for application messaging and mobility that brings so many additional benefits to the organization...***

Regardless of the need for mobility and application messaging, the rise in the ad-hoc use of CIM networks creates a sense of urgency for corporations to implement properly managed EIM solutions. This unavoidable urgency to manage real-time messaging, if addressed within the context of a cleverly-designed framework, can help create the infrastructure for *application messaging* and *mobility* that bring so many additional benefits to the organization, examples of which include:

- Increased productivity
- Improved customer experience
- Improved communication with partners and suppliers
- Improved ability to capture corporate knowledge
- Increased collaboration between employees, particularly among geographically disbursed workgroups
- Increased ability to multitask
- Improved accountability through archiving and reporting
- Reduced voicemail “tags” and elimination of unnecessary e-mail traffic

---

**Highlights**


---

***Sprint Enterprise Application Messaging is based on a client/server architecture, where the predominant responsibility for extensibility to business applications lies with the messaging server.***

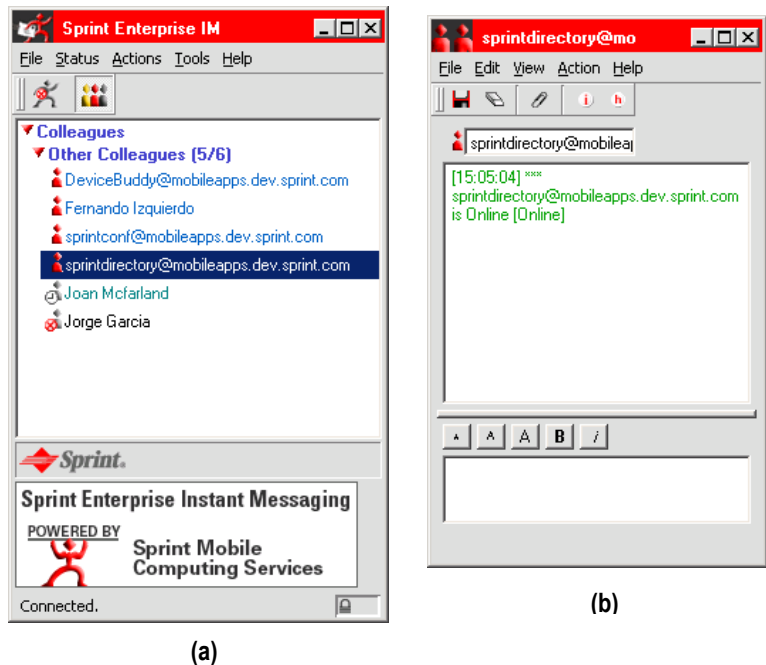
***Business applications simply appear as clients or users of the application messaging server, much like a person who logs in through a desktop IM client.***

### 3. Extensibility to Business Applications

In this document the term *application messaging* is used to denote an instant messaging architecture that has been extended to support person-to-application as well as application-to-application messaging. Future references to *application messaging* imply inclusion of person-to-person messaging capabilities.

Application messaging can be extended to many types of corporate applications, including supply chain, inventory management, order entry, CRM, ERP, financial, directory, telephony, document management, systems & network management, and collaboration tools. The “presence awareness” capability of application messaging makes it particularly ideal for applications that need to communicate with individuals or other applications in real-time. For example, an inventory management system that detects a low inventory can notify individuals whose presence status is “online” of the inventory shortage.

Sprint Enterprise Application Messaging is based on a client/server architecture, where the predominant responsibility for extensibility to business applications lies with the messaging server. Business applications simply appear as clients or users of the application messaging server, much like a person who logs in through a desktop IM client. As shown in Figure 2(a), users of business applications can simply add applications to their “buddy list”. The “buddy list” in this figure includes three applications and three people. For example, “sprintdirectory@mobileapps.dev.sprint.com” is a corporate directory application, which is invoked by selecting the application from the list. This would then cause a chat window to open (Figure 2(b)) where the user can type a text request and receive a text response.



**Figure 2 – Applications and Users Appear as “Buddies”**

---

## Highlights

---

Forms can also be used to interact with applications and to solicit input from the user. Figure 3, demonstrates the use of forms to solicit input from the user of a medical application on a desktop client. Figure 4, demonstrates the same form presented to the user on a mobile Palm® OS device. Notably, the delivery and presentation of the application are consistent.

The screenshot shows a desktop window titled "Edit Stats" with a standard Windows-style title bar (minimize, maximize, close buttons). The form contains the following fields and values:

Field	Value	Max	Min
Hospital	Big Spring State Hospital		
Patient	Greg Globber		
Room	1098		
HR	200	180	20
BP Sys	50	180	20
BP Dia	50	180	20
Temp	98	104	95
B O2	50		10

At the bottom of the form are two buttons: "Submit" and "Cancel".

Figure 3 – Use of Forms on a Desktop Device

The screenshot shows a mobile application window titled "SprintEAM/Application" with a purple header. The form content is identical to Figure 3, but the layout is adapted for a smaller screen. The fields are stacked vertically, and the "Submit" and "Cancel" buttons are not visible in this view.

Field	Value	Max	Min
Hospital	Big Spring State Hospital		
Patient	Greg Globber		
Room	1098		
HR	200	180	20
BP Sys	50	180	20
BP Dia	50	180	20
Temp	98	104	95
B O2	50		10

Figure 4 – Use of Forms on a Mobile Palm OS Device

As is later discussed, using a client/server architecture for application messaging has a number of benefits, including the ability to log sessions, impose usage and security policies, meet regulatory requirements, and provide interoperability with other IM architectures. While all communication appears as peer-to-peer to the user, in reality the connections are only logical peer-to-peer connections managed by the messaging server. Application messaging clients, including application servers, will never establish direct peer-to-peer connections.

***Sprint Enterprise Application Messaging framework uses XML switching, which can easily be extended to many applications and clients.***

Think of the application messaging server as a switch fabric not much different from Storage Area Network (SAN) or Ethernet switches that facilitate real-time communication between connected clients. In addition to creating logical peer-to-peer connections, these switches enable many administrative, policy enforcement, and management functions that would otherwise not be possible. The extensibility of the messaging switch fabric to applications will depend on the switched protocol. Sprint Enterprise Application Messaging framework uses XML (eXtensible Markup Language) switching, which can easily be extended to many applications and clients.

---

## Highlights

---

***... the core XML switch provides the ability to interact with most common protocols and APIs through component libraries and software development kits (SDKs).***

***... rather than creating a new propriety solution, Sprint has adopted Jabber and XMPP (eXtensible Messaging and Presence Protocol) as the core building blocks for its application messaging architecture.***

***The biggest risk to corporations is losing the richness of application functionality. Sprint has effectively addressed this issue with inclusion of natural language processing and workflow capabilities in the core application messaging architecture.***

The interaction between the messaging server and application servers is accomplished through open APIs (Application Programming Interfaces). Many modern business applications expose their APIs through open protocols such as XML, ODBC, JDBC, or SOAP (Simple Object Access Protocol). When older applications with proprietary interfaces are used, it is likely that vendors or IT organizations have already developed adapters that expose the proprietary systems by using open protocols. In Sprint's application messaging framework, the core XML switch provides the ability to interact with most common protocols and APIs through component libraries and software development kits (SDKs).

Sprint's messaging architecture is based on well-known open standards and avoids introduction of new proprietary protocols into the IT environment. Open standards cultivate broad-based support from the developer community and tend to exhibit greater staying power. Therefore, rather than creating a new propriety solution, Sprint has adopted Jabber and XMPP (eXtensible Messaging and Presence Protocol) as the core building blocks for its application messaging architecture. XMPP is an open XML protocol for near-real-time messaging, presence, and request-response services. The basic syntax and semantics were developed originally within the Jabber open-source community, mainly in 1999. In 2002, the Internet Engineering Task Force (IETF) XMPP working group was chartered with developing an adaptation of the Jabber protocol that would be suitable as an IETF instant messaging and presence technology.

As noted earlier, application messaging replaces the elaborate presentation layer of most applications with a simple text- and forms-based client. There are benefits and disadvantages to a simplified user interface. Rapid response, reduced clutter, and extensibility to mobile devices are among the key benefits. The biggest risk to corporations is losing the richness of application functionality. Sprint has effectively addressed this issue with inclusion of natural language processing and workflow capabilities in its core application messaging architecture.

Natural language processing is a server-side capability that allows users to enter simple phrases or abbreviated text that can then be translated into more elaborate query or action commands by the server. From an application development standpoint, natural language processing eliminates the complexities associated with menus and graphical interfaces, in which the developers have to predetermine the types of information to display.

In summary, the server-side capabilities of Sprint's application messaging architecture that facilitate integration with corporate applications include:

- Ability to act as a switch fabric for a well accepted protocol, XML
- Component libraries and SDKs that extend the core switching capability to other protocols and APIs
- Inclusion of open standards that avoid introduction of proprietary or closed protocols
- Provision of natural language processing and workflow capabilities

### **3.1. Client Requirements**

A real-time messaging architecture should provide support for a heterogeneous user environment that may extend beyond the corporate network to employees, partners, and even customers. This emphasizes the need for a variety of clients to support desktops, mobile devices, and browsers. For example a browser-based client can be used occasionally by customers to contact customer support, or used by employees for quick access to corporate resources from non-corporate terminals.

---

## Highlights

---

***Any real-time messaging architecture should provide support for a heterogeneous user environment that may extend beyond the corporate network to employees, partners, and even customers. This emphasizes the need for a variety of clients to support desktops, mobile devices, and browsers.***

***These clients need not include any application-related business logic and should be completely immune from changes to the back-office corporate application landscape.***

The client's primary responsibilities are to establish a secure communication channel to the messaging server, display messages, render graphics and forms, and provide extensibility for voice, video, broadcast and future improvements. Application messaging clients should be small and must require minimum system resources. These clients need not include any application-related business logic and should be completely immune from changes to the back-office corporate application landscape. Simply put, adding to or upgrading corporate applications should in no way require changes to the application messaging clients.

Figure 5, illustrates a Palm OS client running on a Handspring Treo<sup>TM</sup> 300 device. Notable are the similarities between the Palm OS client and the desktop client shown in Figure 2. The messaging server has made available the same user profile and "buddy list" (persons and applications) to both clients.



**Figure 5 – A Palm OS<sup>®</sup> Client on a Treo<sup>TM</sup> 300**

The following list summarizes some key application messaging client requirements:

- Ability to be customized with "skins" that reflect the corporate brand.
- Support for open communication standards such as TLS (Transport Layer Security) for data encryption and XML for data communication.
- Support for a heterogeneous user environment with needs extending beyond the corporate network to employees, partners, and even customers.
- Facilitation of corporate communications much the same way that consumer IM clients display ads and messages.
- Ability to support APIs that allow plug-ins for added capability and customization. For example, voice, video, co-browsing, whiteboards, or text-to-speech capabilities could be added as plug-ins.

---

**Highlights**


---

## 4. Mobility

Extending business applications to mobile devices can significantly enhance employee productivity and operational efficiencies. Extending customer-facing applications to mobile devices can also improve customer satisfaction. While the benefits of mobile computing are clear, the roadmap for extending applications to pervasive devices has been blocked by several obstacles, such as:

- Bandwidth and coverage
- Device limitations including screen size, processor speed, and input method
- Multiple presentation layer form factors
- Security and data encryption
- Presence awareness
- Guarantee and reliability of information delivery
- Cost

With advances in telecommunications technology and next-generation wireless services, the issue of bandwidth and coverage will soon be resolved. Improvements to mobile devices have also been significant. Integration of Palm OS, Windows® CE, and J2ME™ (Java™ 2 Micro Edition) into mobile communication devices has increased the device intelligence and capability. However, screen size, presentation form factor, and input capability will continue to remain an issue for the foreseeable future.

The initial attempts to extend applications to mobile devices were primarily focused on

- WAP (Wireless Access Protocol) and WML (Wireless Markup Language)
- Proprietary solutions, such as the Palm web clipping architecture
- CHTML (Compact HTML) and iMode used primarily in Japan
- Application-focused devices such as the BlackBerry™ for e-mail

While these solutions will continue to play their roles, they share several common disadvantages. For example, they all require dedicated servers, infrastructure, or customization that is only applicable to mobility and cannot be leveraged for other purposes. In addition, the screen limitations have made the menu driven format of these solutions difficult to use and applicable to only a small subset of business applications.

***... with simple text interaction and server-side natural language processing, the small screens of handheld devices can provide richness in application functionality that is comparable to the desktop.***

Application messaging however is a multipurpose solution. Mobility may or may not be the primary driver. The rise in the ad-hoc use of CIM in corporations has created a sense of urgency for IT to implement properly managed EIM solutions. Simple enhancements to the EIM architecture, and introduction of IM clients for Palm OS, J2ME, and Pocket PC devices can easily extend corporate applications to mobile devices at minimum cost. Furthermore, with simple text interaction and server-side natural language processing, the small screens of handheld devices can provide richness in application functionality that is comparable to the desktop.

---

## Highlights

---

In order to extend application messaging to mobile devices, Sprint has incorporated the following six key requirements into its architecture:

- Consistency in application management and delivery
- Mobile presence awareness (i.e. via mobile or desktop)
- Guarantee and reliability of message delivery
- Security
- Application wakeup
- Voice messaging

### **4.1. Consistency in Application Management and Delivery**

The pioneering methods of extending applications to pervasive devices (i.e. WAP and web clipping) have proven to be difficult and expensive to manage. There is a constant need to update and maintain application menu structures as the applications evolve or new device form factors emerge. Management of the ancillary dedicated infrastructure that is required for these solutions also adds to the cost and complexity. Furthermore, it is difficult to adapt these solutions to the emerging web architectures such as Web Services.

Application messaging however provides a consistent method for application delivery that supports and complements most modern application delivery architectures, including Web Services. Applications' presentation layers appear similar on all devices, eliminating the need to create custom menus for each device form factor. This consistent method of application delivery and management helps reduce complexity and cost.

### **4.2. Mobile Presence Awareness**

Before a message is delivered to a mobile user, the messaging server determines whether the user is present on a desktop device or has "gone mobile". To accomplish this, users must define their mobile profiles. This includes their mobile telephone number and other attributes that are required by the policy management framework, such as hours and weekdays when mobile notifications are permitted.

Users have the option to set their presence status from an IM client. For example, user's presence could be set to *online*, *mobile*, or *offline* among others. Users who wish to leave their desktop and receive future messages on a mobile device would set their presence status to *mobile*. Users who wish to receive all of their messages on a mobile device when *offline* would also be able to configure their preferences to that effect.

The application messaging server provides the user's presence status to others. For example, a small cell phone icon indicates that the user is capable of receiving messages on a mobile device. There are two possible presence situations where a cell phone icon is used.

1. User has "gone *mobile*" with no active mobile connection - A *mobile* presence status does not guarantee that the user is available to receive or to respond to any messages. It only indicates that an attempt will be made to reach the user on his mobile device.

---

## Highlights

---

*... Sprint's architecture provides a store-and-forward proxy that accepts messages on behalf of the user and relays those messages to the user when his status becomes online via a desktop or mobile device.*

*A robust application messaging architecture should provide a granular level of mobile security that can be managed by the user as well as a policy management engine.*

*Encryption and store-and-forward proxy work together to guarantee the privacy of messages.*

2. The user is *online* from a mobile device - This indicates that the user is present to send and receive messages in a secured environment via a mobile device.

### 4.3. Guarantee and Reliability of Message Delivery

The next requirement that is addressed by Sprint's application messaging architecture is the guarantee of message delivery. Cell phones could be turned off or the user may enter an area where coverage does not exist. To address this issue, Sprint's architecture provides a store-and-forward proxy that accepts messages on behalf of the user and relays those messages to the user when his status becomes *online* via a desktop or mobile device.

### 4.4. Mobile Security

Security is addressed by leveraging TLS (Transport Layer Security), WTLS (Wireless Transport Layer Security), IPSec or similar security protocols, or VPNs (Virtual Private Networks) for encryption. A robust application messaging architecture provides a granular level of mobile security that can be managed by the user as well as a policy management engine. For example, users must be able to request secure communication with some "buddies" and unencrypted communication with others. Before a further discussion of this issue, let's see how messages can be delivered to a user.

Encryption and store-and-forward proxy work together to guarantee the privacy of messages. A message sent to a mobile user can be delivered either immediately via a protocol such as SMS, SMTP, or WAP Push, in which case security is not guaranteed; or alternatively the user could be notified via an alert that a message is waiting. In the latter case, after receiving the alert, the user must authenticate to the messaging server via a mobile or desktop client to securely retrieve the message from the proxy. After logging in to the messaging server, regardless of the device used, the user's presence status will become *online*. At this point the user can securely communicate with other users instantly or access corporate applications.

Should the communication line drop, the store-and-forward proxy continues to collect messages for the user until the user returns *online* or opts out of the mobility service.

Situations may arise when convenient and timely access to information may require that the message be sent directly to a pager or a mobile phone. In that case, security of the message cannot be guaranteed. However, corporate policies or users must determine whether such communication is allowed. For example, a message from an assistant informing a supervisor about a lunch appointment can be more conveniently handled unencrypted. Another example includes messages sent to system administrators via monitoring agents or system administration tools such as Tivoli®. The organization should have the flexibility to either establish central policies for encryption or to delegate such decisions to users, or both.

The previous discussion is applicable to messages sent to the user while his presence status is set to *mobile*. If the user wishes to access applications, or respond to a message, authentication to the messaging server becomes necessary, which will generate an *online* status and a secure connection.

---

**Highlights**

---

#### **4.5. Application Wakeup**

As described previously, in order to securely view messages users must activate their messaging client on a mobile device when a notification from the proxy server is received. Newer J2ME devices do, however, allow specific applications to be launched via SMS messages, therefore enabling automatic launch of the client when a new message is waiting.

Mobile Information Device Profile (MIDP) is a component of J2ME that facilitates development of downloadable applications and services for mobile devices. MIDP version 2.0 has introduced a “push registry,” which enables applications to set themselves up to be launched automatically, without user initiation. The push registry allows for network- and timer-initiated application wakeup.

MIDP 2.0 defines new connection types beyond HTTP, including TCP sockets and UDP datagrams that can be set up as inbound connections, making them suitable for push. In addition, the Wireless Messaging API (WMA) makes SMS-based push activation possible as well. It is this feature of J2ME that enables activation of real-time messaging clients on mobile devices.

#### **4.6. Voice Messaging**

Messages stored by the proxy can also be delivered via voice notification. In this scenario, the user would be alerted of a pending message via SMS, SMTP or other network or device dependent protocol. The user then dials into a central text-to-speech (TTS) server to listen to the message. The authentication to the TTS server can be accomplished via caller ID or a pin. The extension of voice capability to the XML switch fabric is easily accomplished via component libraries and SDKs.

Integration of a modern PBX or IP telephony system with application messaging is yet another possibility. From the messaging client, users would be able to:

- Receive voice mail notifications
- Browse their voice mailbox to view number of messages and, date, time, message length, and caller-id information.
- Forward messages to another voice mailbox
- Forward voice messages via e-mail
- Delete or save messages
- Initiate conference calls
- Configure greetings and other voice mail preferences

### **5. Directory Integration and Access Control**

Directory services, such as Microsoft’s Active Directory™ or Sun™ ONE Directory Server, and directory access protocols, such as the Lightweight Directory Access Protocol (LDAP), form a predominant method for managing and accessing user profiles and authentication information. Secure authentication and access control frameworks such as RADIUS are also in use in many organizations.

---

## Highlights

---

***Rather than maintaining a separate user database and increasing the burden of system administration and security management, Sprint's enterprise application messaging framework provides connectors to the existing corporate directories as well as authentication and access control frameworks.***

***... the archive engine should log the information encrypted so that it is only retrievable through the appropriate administrative and security channels. This requires a tight integration ... with the "reporting and analysis" and "message encryption" engines.***

Rather than maintaining a separate user database and increasing the burden of system administration and security management, Sprint's enterprise application messaging framework provides connectors to the existing corporate directories as well as authentication and access control frameworks.

This level of integration provides the capability to

- Authenticate against existing directory infrastructure
- Eliminate multiple passwords and facilitate single sign-on
- Create, maintain, and delete user profiles
- Perform inquiries for inclusion of individuals or applications in "buddy lists"

## 6. Archiving, Logging, and Compliance

Maintaining archives and logs of IM conversations is mandatory for most organizations. Just as e-mail archives allow for enforcement of e-mail usage policies, similar policies must be implemented as the user population transitions to real-time messaging.

Archiving is necessary for the following reasons:

- Industry-specific federal mandates such as HIPAA for healthcare institutions and GLBA and the Patriot Act for financial organizations require compliance.
- As IM conferencing becomes more popular, it is vital that the conferencing sessions be captured to retain corporate knowledge.
- Compliance with corporate usage policies must be enforced. For example, sessions established by suspected employees or contractors could be monitored, or logs could be scanned for questionable keywords or phrases.

Since the captured messages may include private or sensitive data, the archive engine should log the information encrypted so that it is only retrievable through the appropriate administrative and security channels. This requires a tight integration of the archive engine with the "reporting and analysis" and "message encryption" engines.

With respect to privacy needs, messages can be generalized into three basic categories.

- Secured – No other users, including the administrative and security personnel should have access to secured messages. Preferably, secured information such as user and application passwords should not be logged at all.
- Private – This includes data exchanged between users and/or applications for private consumption of those entities. As necessary, authorized corporate personnel or application agents should be able to access the information to ensure compliance with regulatory as well as corporate policy requirements. Private data should also be retrievable by those involved in the conversation. However, private messages should be stored in an encrypted format.
- Public – These are messages that are intended for consumption by the corporate community, therefore, their logs may not require privacy protection. Examples include training sessions, or general corporate communications.

---

## Highlights

---

***Sprint's application messaging framework should provide the necessary reporting and analysis application agents to replay, manipulate, analyze, and summarize the archives.***

***... giving users the ability to retrieve past conversations, replay IM conferences, ... can enhance productivity and will place the corporate knowledge where it is needed the most.***

***Sprint's instant messaging framework assumes that security threats from within the organization are just as likely as the external ones.***

## 7. Reporting and Analysis

Sprint's applications messaging framework provides the necessary reporting and analysis application agents to replay, manipulate, analyze, and summarize the archives.

Real-time messaging significantly improves a company's ability to satisfy reporting and compliance requirements as compared to other modes of communication, such as telephone conversations.

Reporting and analysis is of great value to both the company and the user community. Companies tend to view reporting as a defensive strategy that ensures compliance with corporate usage policies and regulatory requirements. However, giving users the ability to retrieve past conversations, replay IM conferences, and view application access logs, also enhances productivity and places the corporate knowledge where it is needed the most.

When used for policy enforcement, reporting and analysis engines must be as automated as possible to minimize human interaction, which is the "weakest link" in any security framework. For example, robots or application agents should be used to scan the logs for phrases and generate reports only when violations are detected. Reporting tools include:

- Application agents for scanning the logs
- Report generators for capturing IM sessions
- Administrative tools for customizing and analyzing reports
- Session playback capability for end users, preferably through an IM client

## 8. Message Encryption

The premise behind message encryption is to maintain the privacy of messages by preventing unauthorized access from both inside and outside the corporate firewall. Message encryption and privacy is completely absent in CIM frameworks, leaving messages susceptible to eavesdropping and interception.

Sprint's application messaging framework assumes that security threats from within the organization are just as likely as the external ones.

Most messaging architectures require users to log in to a central message server. The primary purpose of the message server is to maintain the presence and connection information, store user profiles and "buddy lists", relay messages between users, manage chat rooms and conferences, enforce policies, provide interoperability, and archive messages.

In order to archive a message, the message server must have access to a clear-text copy of the message. It can then encrypt the message using its own public key or symmetric ciphers for archival.

In general a conversation between two application messaging clients can be encrypted in two ways:

- Peer-to-peer – where data is encrypted between the two clients and can only be decrypted by the clients.

---

## Highlights

---

***...conversation between two application messaging clients can be encrypted in two ways:***

- ***Peer-to-peer***
- ***Hub-by-hub***

***... corporations need to implement and manage real-time messaging policies.***

***To help educate the users about the corporate policies, any alerts caused by violation of those policies could be immediately brought to the users' attention through the application messaging client.***

- **Hub-by-hub** – where data is encrypted between the client and the message server. In this model, a message received by the server is decrypted using the sender's key, and re-encrypted in the recipient's key before being transmitted to its final destination.

If two clients were to establish a peer-to-peer encrypted session, the message server would not be able to archive a useable copy of the message. This is similar to two e-mail clients using PGP (Pretty Good Privacy) to exchange messages. Referring back to Section 6, and the three "secure", "private", and "public" message categories, it is evident that peer-to-peer encryption is suitable for "secure" messages, such as application passwords, since this information must be protected at all cost. For "private" and "public" messages, hub-by-hub encryption is desirable, since it would facilitate access to messages for archiving, policy enforcement, and reporting.

It is possible for an application messaging infrastructure to include more than one message server, each responsible for a domain or a group of applications and users. Communication between all messaging servers is encrypted, as is communication between application servers and message servers.

## 9. Policy Management

As for most communication tools, such as e-mail, corporations need to implement and manage real-time messaging policies. These policies should be monitored, maintained, and enforced by a policy management framework. At the first glance, policy management may not appear to be a critical issue. However, once instant messaging is extended to corporate applications and customer support its importance becomes apparent.

An effective policy management framework should support three classes of policies, including:

- **Individual Policies** – These are user communication preferences, "buddy lists", presence visibility to others, disallowed lists, mobile preferences, such as weekdays and times that mobile notification is allowed, and more.
- **Security Policies** – These are access control, encryption, group segregation, archive & logging, and other security related policies. For example, consider a healthcare or financial institution customer support representative who requires the ability to query customer information while at work, but not outside work hours. The policy management framework should curtail access to customer information outside the scheduled work hours.
- **Usage policies** – These are for monitoring compliance with corporate usage policies for conduct, and enforcement of communication standards with customers and partners. Real-time monitoring of customer support sessions is an example of "usage policy" enforcement. Policy management application agents could in real-time monitor conversations for phrases and comments that may not be in the best interest of the company. If a situation is detected, a supervisor could be invited to join and monitor the conversation. It is important to note that most users wish to respect corporate policies. To help educate users about corporate policies, any violation of those policies could be brought to the users' attention immediately through the application messaging clients.

A robust policy management framework requires at least three levels of granularity for application of policies. These include:

---

## Highlights

---

- Individual users and applications
- Groups and Subgroups of users or applications
- The application messaging domain

The organization controls the domain, group, and subgroup policies. For example a corporate policy may establish that all members of the “morning shift” of the “teller” group should have access to the banking applications from 7:30 AM to 5:00 PM. In general each lower level of granularity will inherit the policies established for its parent. However, the parent should have the flexibility to delegate the policy to a lower level. For example, a domain policy may establish that all communication between users must be encrypted. It could, however, provide the flexibility to allow groups to override that policy. A group may also allow individuals to override policies. This level of flexibility in policy management enables organizations to balance their need for control and convenience.

Individual users and applications will inherit the policies applied to their groups or subgroups. At the discretion of the organization, groups may have the flexibility to delegate some policies to the users. Should a user belong to multiple groups, the policy management framework calls for enforcing the most stringent requirements.

Since each application messaging domain could consist of multiple servers, the policy information should be stored in a data repository and be accessible by all servers via LDAP. This architecture facilitates scalability of the environment without the need to configure each individual server. Another benefit of leveraging LDAP is that any existing directory infrastructure could be used as a repository for the policy information.

## 10. Virus Protection

Additional productivity tools that enable document sharing, co-browsing, file transfers, and access to corporate document management systems enrich the core application messaging capabilities.

Document sharing capabilities are currently offered by many CIM platforms. As these features enter the corporate environment, there is a compelling need to implement the necessary security features to minimize the threat of viruses. Therefore, Sprint’s application messaging framework facilitates inspection of transferred or shared documents by virus protection tools.

## 11. Reliability, Availability, and Scalability

Reliability, availability, and scalability (RAS) are three concepts that we have come to expect from almost all IT services. For application messaging, these are even more critical. Since the messaging server acts as a protocol switch fabric that connects people and processes, it must exhibit a high degree of resiliency and redundancy. This switch fabric must remain operational 24x7x365.

The application messaging architecture must also support hundreds of thousands of concurrent users and yet continue to perform flawlessly. In order to scale to a large number of users, and support organizations with multiple divisions and administrative domains, the application messaging architecture must support a distributed environment where many individual messaging servers could be interconnected. This requires a modular architecture

***... there is a compelling need to implement the necessary security features to minimize the threat of viruses.***

***In order to scale to a large number of users, and support organizations with multiple divisions and administrative domains, the application messaging architecture must support a distributed environment where many individual messaging servers could be interconnected.***

---

## Highlights

---

***In an application messaging environment there are two methods to track presence information, a persistent TCP socket connection and polling.***

***The core Jabber framework that is adapted by Sprint for its application messaging architecture leverages TCP socket connections to maintain highly accurate presence information for users and applications.***

***To further enhance the scalability of the system, Sprint's architecture distributes client TCP sessions across multiple processors known as "Connection Managers". Connection Managers run on independent hardware platforms from the core XML switches.***

***Unparalleled scalability can be achieved by increasing the number of Connection Managers and core XML switches in the network.***

in which a series of XML message switches operate in concert, while allowing for redundant servers servicing the same domain or groups of users.

Scalability to hundreds of thousands of users while maintaining accurate presence information for every connected user is a serious challenge, but not one that has gone unsolved. In an application messaging environment, there are two methods to track presence information, a persistent TCP (Transmission Control Protocol) socket connection and polling.

Polling could be accomplished through a number of different protocols, such as HTTP. HTTP has some obvious benefits, such as the ability to easily bypass firewalls or implement browser-based clients. Polling, however, is extremely chatty. While this may not be an issue for the IM client, it could create a significant amount of traffic on the server side where thousands of users may be connected. Since polling is stateless, each transaction must re-establish the operating status of the session. This requires searching for information such as cookies, and establishing any or all changes that have been made since the last transaction, which results in a significant amount of overhead traffic. The increased traffic could degrade performance when large numbers of users are connected.

Establishing a persistent TCP socket would resolve many of the chattiness issues that are associated with polling. With persistent TCP connections, very little overhead traffic is required to maintain the connection. Data is sent across the connection only when state changes, or XML payloads are transmitted. A key advantage of persistent socket connections is the congestion control capability that is built into TCP. TCP's congestion control does not work well for short-lived connections, such as those that might be established for polling. Long-lived socket connections, however, allow TCP to gather enough transmission history to learn about the behavior of the connection.

The core Jabber framework that is adapted by Sprint for its application messaging architecture leverages TCP socket connections to maintain highly accurate presence information for users and applications. To further enhance the scalability of the system, Sprint's architecture distributes client TCP sessions across multiple processors known as "Connection Managers". Connection Managers run on independent hardware platforms from the core XML switches. This allows for specifically tuning the performance parameters of the operating system to handle large numbers of active TCP sockets, up to possibly 25,000 or more on a single processor. Each Connection Manager condenses client TCP connections into a single connection to the XML messaging switch. Therefore, the messaging server (the core XML switch) is not required to handle a large number of connections, leaving its resources available for parsing and streaming of XML messages. Unparalleled scalability can be achieved by increasing the number of Connection Managers and core XML switches in the network.

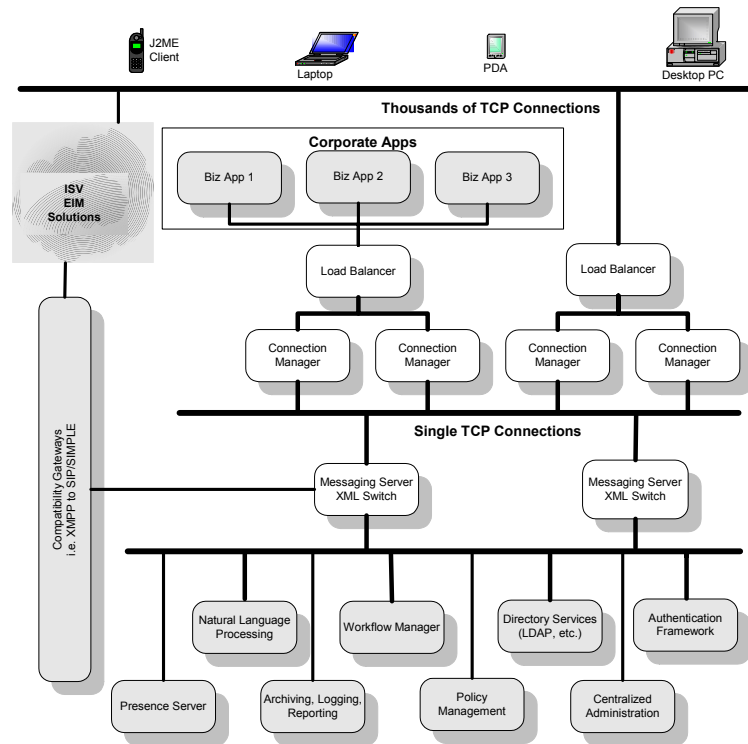
The Connection Manager architecture, as depicted in Figure 6, also plays a crucial role in reliability and availability of the application messaging system. As can be seen in this figure, a high degree of redundancy can be built into the system. For instance, if one connection manager fails, new connections could be routed to other connection managers. It is conceivable that users could get temporarily disconnected; however, they can immediately reconnect to continue operations. The same degree of redundancy exists for the core XML switch. If one switch fails, another can immediately take over.

As is shown in Figure 6, in an effort to improve availability, it is possible to segregate application and user traffic by dedicating one or more Connection Managers to applications.

---

**Highlights**


---



**Figure 6 – A High Level Framework For Sprint Enterprise Application Messaging**

***For Sprint, interoperability meets two objectives. First, it enables Sprint's customers to communicate with their partners and clients who may be using a different EIM platform. Second, it facilitates coexistence with other EIM platforms within the same enterprise, thereby allowing customers to leverage the enhanced application messaging and mobility capabilities of Sprint's architecture to complement any existing or planned EIM solutions.***

## 12. Interoperability

In the CIM market, interoperability has been a significant issue. Major ISPs had business incentives to keep their CIM environments closed. As these ISPs and other ISVs begin to release enterprise versions of their IM applications, it is becoming clear that lack of interoperability would hamper successful adoption of these solutions by the enterprise. For instance, significant consequences could arise if two companies on two different e-mail systems were not able to exchange e-mail messages. As inconceivable as this may be, just under ten years ago many internal corporate e-mail systems were not interoperable.

For Sprint, interoperability meets two objectives. First, it enables Sprint's customers to communicate with their partners and clients who may be using a different EIM platform. Second, it facilitates coexistence with other EIM platforms within the same enterprise, thereby allowing customers to leverage the enhanced application messaging and mobility capabilities of Sprint's architecture to complement any existing or planned EIM solutions.

Referring to Figure 6, Sprint's application messaging architecture utilizes interoperability gateways that facilitate presence awareness, session initiation, protocol conversion, and encryption for connecting to other ISV EIM platforms.

For session initiation, the IETF Session Initiation Protocol (SIP) is the most common protocol in use. SIP is used to initiate calls or sessions between packet-switched networks,

---

## Highlights

---

including Voice-over-IP (VoIP) and instant messaging sessions. Instant messaging, however, requires the ability to detect another party's presence in real-time, a feature that SIP did not provide but was later introduced in SIP Instant Messaging and Presence Language Extensions (SIP/SIMPLE).

Most ISV EIM solutions, such as the newly announced Microsoft RTC Server and Lotus Sametime 3.0a, will leverage SIP/SIMPLE for session initiation and presence awareness. Sprint's interoperability gateways provide translation of SIP/SIMPLE protocol to and from XMPP.

In addition to session initiation and presence awareness, real-time communication requires transmission of text, voice, or video payloads between applications and users. As discussed previously, Sprint Enterprise Application Messaging uses streaming XML payloads. Other EIM solutions may use other protocols to carry their payload. For example, Microsoft's RTC uses the Real-time Transport Protocol (RTP). In this case, the translation between RTP and the streaming XML payloads will again be performed by the interoperability gateway.

To facilitate protocol conversion and translation, the interoperability gateways will also act as security checkpoints where the payloads will be decrypted, re-encapsulated into a new protocol, and encrypted again for transmission to the messaging server.

### 13. A Reference Implementation

To demonstrate application messaging, we have selected a popular application that has been integrated with Sprint Enterprise Application Messaging for Sprint's own internal use - the Sprint corporate directory.



Figure 7 – Instant Messaging Client on a Palm OS Device

The demonstration of this application will utilize screenshots from a Palm OS messaging client. The user experience from other clients, including desktop clients, will be similar. Figure 7, illustrates the Sprint Enterprise Instant Messaging client icon on a Palm OS device. Selecting the SprintEIMs icon will activate the instant messaging client.

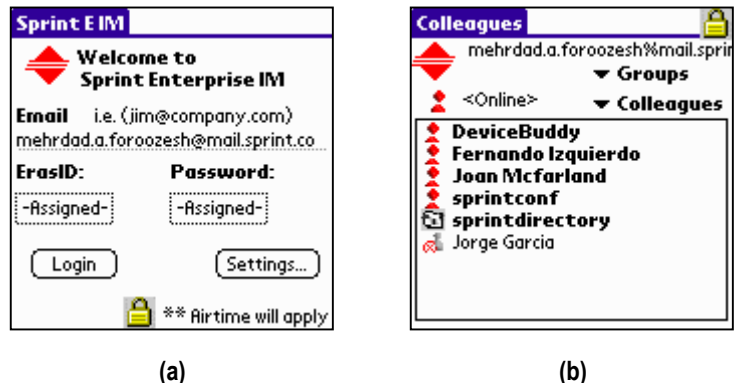
Figure 8(a), shows the login window for the instant messaging client. The user's e-mail address serves as the user ID. The internal Sprint implementation also requires that an ErasID and password be provided to allow VPN or TLS access to internal Sprint resources. The entire user-provided information is authenticated through the corporate RADIUS

---

**Highlights**


---

servers and LDAP, thereby eliminating the need for maintaining any authentication information specific to the enterprise messaging architecture.



**Figure 8 – Instant Messaging Client Startup**

Once the user has successfully authenticated to the system, the main screen of the messaging client appears (Figure 8(b)). This screen contains a list of “buddies” that includes individuals and applications selected by the user. The “buddy list” is maintained by the messaging server and is presented to the messaging client once the user authenticates. The lock icon pictured in both Figure 6(a) and Figure 6(b), indicates that the established session is encrypted.

At this point, the user has the option to initiate a chat session with any of the applications or individuals that are online. “Buddies” that appear in bold text are online and available for immediate access.

Sprint’s corporate directory is a single source of information for locating contact information, title, reporting structure, and employment status of employees and contractors. Previously, this information was only available through the corporate portal via a web browser. Currently, the corporate directory is accessible by all Sprint associates through desktop and mobile IM clients. The following demonstrates a simple session with the corporate directory application.

Figure 9(a), is the introductory screen of the application, shown as a typical chat window. In this case, the user wishes to search for individuals whose last names begin with characters “Foro”. The user simply types “foro” in the prompt line and selects “Send”. The messaging server receives and logs the message, and then, using natural language processing, creates a query command and forwards the result to the directory server. The directory server identifies “Mehrdad A Foroozesh” as the only individual that matches the query and returns that information to the messaging server where it is logged and then forwarded to the client (Figure 9(b)).

At this point, the user can input a number of commands and shortcuts to query specific information about the identified individual. For example, by typing “phone” in the prompt line, Figure 10(a), the user requests the individual’s phone number. The natural language processing engine remembers the results of the last query and uses that information to conclude that the user is seeking the phone number for Mehrdad A. Foroozesh. A new

Highlights

query string is created to retrieve the phone number from the directory, the results of which are shown in Figure 10(b).

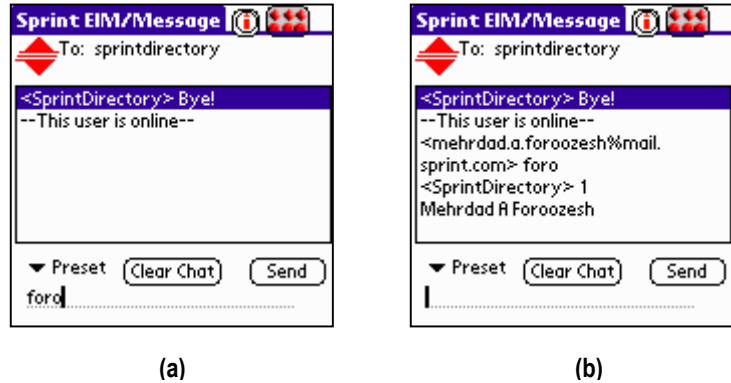


Figure 9 - Requesting Information From The Corporate Directory

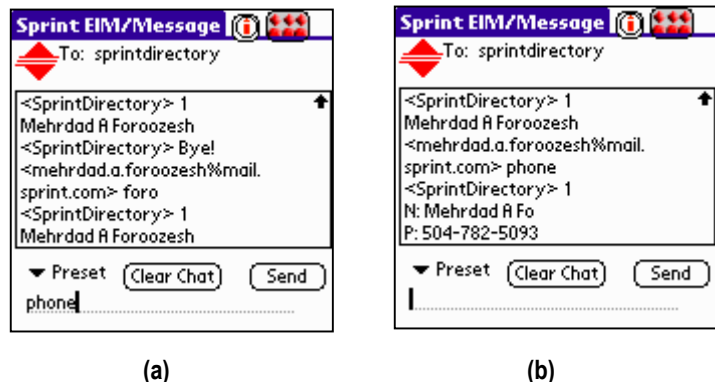


Figure 10 – Requesting a Telephone Number From the Directory

Figure 11(a), demonstrates the use of abbreviated text to retrieve information. For example, to retrieve the e-mail address of the individual, the user is only required to type enough characters to form a unique request. In this case the character “e” would be sufficient to request the e-mail address.

The corporate directory application has many features and capabilities that are beyond the scope of this discussion. By typing “help” on the prompt line users can see all the options available to them.

---

**Highlights**


---

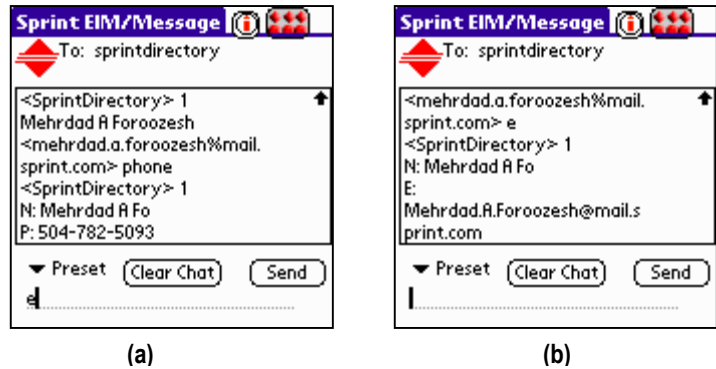


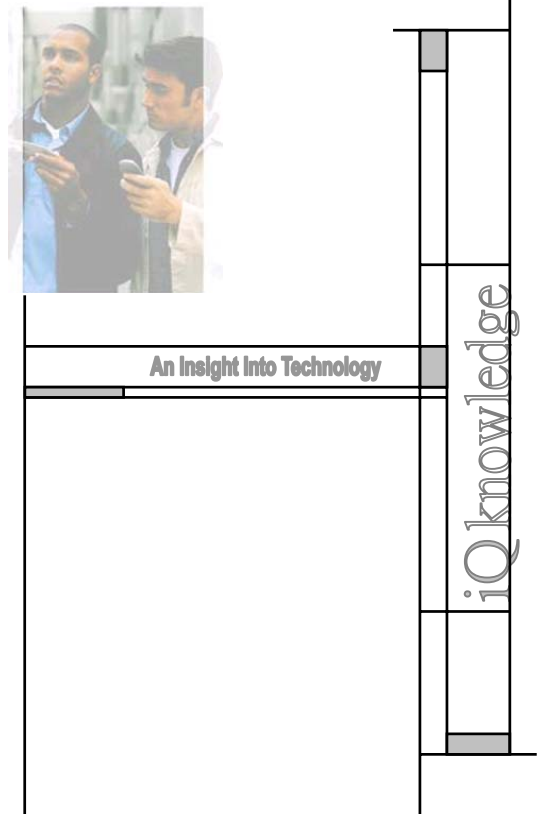
Figure 11 – Using Abbreviated Text To Query Information

## 14. Conclusion

Many corporate employees have discovered the benefits of real-time messaging in a business environment and have taken individual initiatives to incorporate CIM as a tool to communicate with colleagues, partners, and customers. While employees have seen productivity gains from the use of CIM tools, corporations are concerned with its lack of security, control, and policy compliance. Corporations are left with two options, either ban IM as a tool and forgo its benefits, or embrace the technology with even greater functionality that is suitable for the corporate environment.

The latter is the obviously the most beneficial choice for most organizations. Businesses depend on productivity gains to offset costs and competitive pressures. They cannot afford to ignore one of the fastest growing technological tools that can be deployed to bring people and applications together in real-time on any device. To unleash the true power of presence awareness and real-time messaging, businesses should embark on implementing a well-designed messaging solution that can extend instant messaging to applications and mobility. Sprint Enterprise Application Messaging provides the necessary architecture and framework to accomplish this objective.

InQuest Corporation is pleased to present iQknowledge®, a series of whitepapers to assist IT Professionals in making good technology decisions to support their business needs.



**InQuest Corporation**  
5137 Belle Drive  
Metairie, LA 70006 USA  
Tel: 504-456-7380  
[www.inquest-corp.com](http://www.inquest-corp.com)

This document may be reproduced and distributed in whole only when it includes the cover page and this notice. Any reproduction, use, appropriation, or disclosure of this information, in part, without the specific prior written authorization of InQuest Corporation is strictly prohibited.

Copyright © 2003 InQuest Corporation. All rights reserved. Unpublished rights reserved under U.S. copyright laws. InQuest, iQ Knowledge, and InQuest logo are trademarks of InQuest Corporation. All other trademarks are property of their respective owners.