Message-On-Demand Service in a Decentralized Unified Messaging System

Prem Prakash Jayaraman, Paul Hii, and Arkady Zaslavsky

Abstract— **Variety of service integration concepts have emerged during the last few years. Most of these aim at the concept of integrating telecommunication and the data communication technologies. One form of such a system is Unified Messaging. Unified Messaging enables users to manage their messages independent of location, communication device or communication medium. Most of the existing systems provide centralized message-store based access to messages but lack services like user personalization, message notification and are non-pervasive. In this paper, we define UMS as a user-centric system that provides messaging services based on user demands and preferences. We have proposed a decentralized Unified Messaging System (DUMS) that is pervasive and context-aware. Based on our proposed architecture, we successfully implemented and demonstrated a messaging system called *i-*UMS that ensures a user receives almost instantly all messages such as emails, instant messages and so on in an intelligent and most appropriate manner.**

Index Terms— **UMS, Service Adaptation, Pervasive, Service Personalization, Ubiquitous**

I. INTRODUCTION

The Internet has now become the universal mode of communication letting users to communicate with each other by a number of means like email, IM, Voice over Internet Protocol (VoIP), etc. This development has led to the convergence of geographically separated users. Number of messaging technologies and services has evolved during the past few decades with the advent of new communication technologies and devices. These messaging systems work in common to deliver messages to the user, but differ in their architectures and message formats. The systems use different communication medium to deliver messages and users' use different end-terminals to receive messages. E.g. SMS is sent and received over GSM network while email is sent and received over a transmission control protocol (TCP). This diversity in the various messaging systems has left end users with a plethora of messaging services each of which provides

Prem Prakash Jayaraman is a Research Student with Monash University, Melbourne, Australia 3145 (corresponding author phone: +61-3-9903-1151, email: prem.jayaraman@infotech.monash.edu.au)
Paul Hii is a Research Student with Monash University, Melbourne, Australia 3145 (email: paul.hii@infotech.monash.edu.au)
Arkady Zaslavsky is an Associate Professor with School of Computer Sci & Software Eng, Monash University, Melbourne, Australia 3145 (email: arkady.zaslavsky@infotech.monash.edu.au)

services in a different format. Also users making use of these messaging services have to consider the recipients' location and end-device to obtain prompt reply from the message recipients [Wj04].This approach is unsatisfactory. Hence, this led to the investigation of a message unification service that enables end-users to send and receive messages irrespective of the messaging system format and architecture. Unified Messaging (UM), which is a widely accepted service, aims at this goal of integrating various messages irrespective of their architecture. Unified Messaging can be defined as a system that enables transfer of messages independent of the underlying message architecture, communication medium and devices. Any form of messages such as email, SMS, fax, voicemail, etc. can be delivered via any communication medium [As99]. The UM can be a powerful means of communication as it seamlessly integrates various communication technologies hence, giving the user the flexibility to receive and send any message, anytime and anywhere [Vs00][As99].

UM incorporates services that convert messages like email, fax, voicemail, SMS from one format to another format. Most of the UMs perform this process of message conversion without taking into consideration the user's preferences. The goal of UM is to deliver messaging services to the user's in the most appropriate method. Current UMs does not entirely fulfill the goals of the unified messaging. Hence message conversion is no longer the primary goal of a UM. The system must be able to adapt itself into the user's environment. It needs to take in account user's context [Dc02], user's end-terminals and user's working environment. This is an intelligent approach that creates the foundation in developing a Unified Messaging System (UMS) that adapts itself based on user's location and user preferences hence enabling the Messaging-On-Demand service. The Message-On-Demand in this context can be defined as a service that provides users with messages on demand based on user's preferences, user's context and user's personalized service preferences. The aim of the user in such an environment is no longer to have ubiquitous access [Vs00] to messages instead personalization of the messaging services based on user's messaging demands. Since the system needs to be context-aware and terminal-aware, the environment in which the devices work can be described as a virtual personal area network (VPAN),

where the devices function to deliver the best possible service to the user in a timely and most appropriate manner possible. The devices in the VPAN have enough intelligence built into them to coordinated and exchange information with its peers (others devices that provide messaging services).

The VPAN as depicted in figure 1 is a virtual environment and is a vision of the future. Devices in the VPAN are intelligent devices that have enough knowledge about the user's location and the capabilities of the other devices working within the environment. To implement a Message-On-Demand service in a UMS, the UMS need to be context-aware, terminal-aware and must be able to adapt itself based on user's personalization. Such a system working within a VPAN delivering messaging service to the user based on user demands is more realizable using a decentralized design. The devices constituting the VPAN depicted in figure 1 are decentralized which have enough intelligence to respond to user's location. This paper discusses one such decentralized architecture of a UMS that provides Messaging-On-Demand services to the user. The decentralized UMS is a pervasive [Rg00] system.
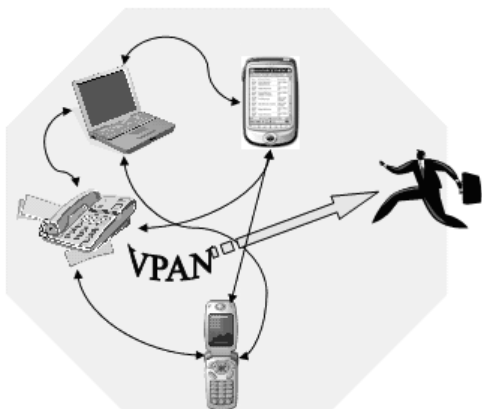

Figure 1: VPAN with user devices delivering message services to the user

Service adaptation and service personalization is one of the key factors of the modern systems, to implement a Message-On-demand service. With the evolution of pervasive computing, users have started to feel the need of service adaptation and personalization in their participating environment.

This paper presents an overview of the challenges involved in designing a decentralized UMS. A decentralized UMS tries to provide service personalization to the end user hence enabling message notification based on user demands. The system is also location and terminal aware hence, tries to provide the messaging services to the user in the most personalized manner taking into consideration the user's context, location and the end terminal which the user uses to handle the messages. The decentralized UMS also tries to overcome the challenge of delivering messages to the user in a most appropriate manner rather then just converting one

message to another message format as in the traditional universal mailbox systems. The decentralized UMS functions in the VPAN in which the user is the primary objective of the system. Section 2 discusses some of the existing Unified Messaging solutions and how they differ from the system being proposed in this paper. Section 3 proposes the architecture of a decentralized UMS. It then proposes *i*-UMS, a decentralized UMS that has been prototyped as a proof of concept system. The *i*-UMS system uses wired technologies, GSM, GPRS, 802.1X wireless technologies including Bluetooth for communication between user devices for message transfer. Section 5 discusses the results obtained from the implementation.

II. UNIFIED MESSAGING SYSTEMS – AN OVERVIEW

There are a number of commercial Unified Messaging solutions that try to satisfy the goals of the unified system. Current systems depend on a gateway for message delivery which is an obvious approach for most of the UMS. These systems try to unify messages at a single end point message store by collecting messages from different message stores. The system aim at providing message delivery to the users without taking into considerations the users need. In such a system the user has minimum capability to define his reach ability from message senders. Figure 2 depicts a Unified Messaging solution provided by Eicon. Some of the other commercial UM that are available are: Nortel CallPilot [No05] [Lh03], Cisco Unity [Cu04], Global Com's Internet based UMS [Db02] and AVST Call Xpress [Av04].
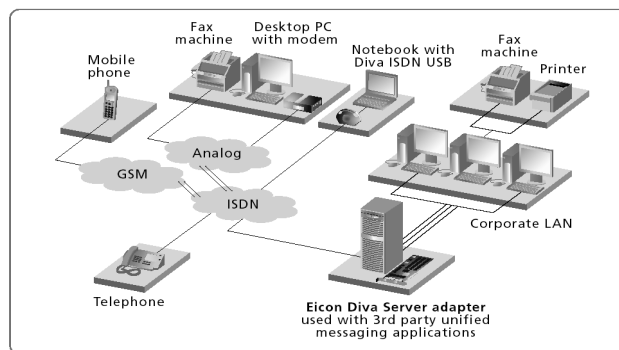

Figure 2: Eicon Diva Server [Dh03]

The basic requirement to achieve for service personalization and service adaptation is the ability to be aware of the user's environment. This primary function of being context-aware and pervasive is absent in the current UMs. The current systems also have the ability to convert messages from one form to another but they do not provide any service adaptation based on the user demands. E.g. conversion of an incoming SMS into an email and storing it in the central message store for retrieval. This example provides message conversion but does not take into consideration the user's context before converting the message into the most appropriate form. This is the same in

case of message notification. The lack of service adaptation also raises a few more questions. Some of the UM solutions use only email or SMS as the one form of message notification. But this will not satisfy the needs of the user since the user may have limited resources and hence cannot access one form of message service from another device [As99]. This approach is not always applicable in all cases especially in our VPAN where the devices work to deliver services to the user in a most appropriate manner possible. The method of using few types of message notification is not enough with the evolution of newer end-terminals and newer messaging technologies. To support this, we need to discuss the properties of few of the existing messaging technologies based on the properties show in table1 [Wj03] [Wj04]:

Time	How long does the message needs to be stored
Direction	Is message Uni or Bi directional
Audience	The potential list of recipients
Address	Does system provide support for multicast, broadcast

Table 1 : EMS properties [Wj03]

Based on the above key properties of messaging service, the following table2 [Wj03] [Wj04] gives a classification of four messaging systems and to which category they fall into.

	Email	SMS	voicemail	IM
Time	Permanent	Permanent	Temporary	Immediate
Direction	Simplex	Simplex	Simplex	Duplex
Audience	World	World	Single	Group
Address	List	List	Single	List

Table 2: Classification of four messaging systems

From the above table, it is clear that email messages fall short in the most important feature of not being duplex. It is a key factor for a pervasive messaging system to use duplex communication which can enable the user to respond immediately. Another problem is that email addresses only a list of users and not the world. Hence, using email as a method of message delivery in a pervasive environment is not viable [Wj03] [Wj04]. Also if we can consider the properties of other messaging systems, we see that each one of them fall short in one particular property. Hence we cannot use just one unique method of unification. This is where the service adaptability plays a major role. The UM must be able to adapt the message delivery based on the user's location and the end-terminal.

Another shortfall of the existing UM solutions is personalisation for the end user. The current UM solutions provides a basic level of personalisation that enables users to filter certain messages. This level of personalisation is so minimal that the system cannot personalize message delivery based on user preferences. E.g. Users can block a particular person from sending them an email irrespective of his/her

location. But this is not satisfactory in terms of a UMS that delivers messages on demand. The system must be able to filter messages depending on user current environment and his environmental preferences. E.g. delivering a high priority message to the user when the user is in a meeting. The priorities are set by the users and vary depending on user's preferences. Similarly present UMs do not provide any form of service personalization. The system needs to choose the appropriate service that can be used to notify the user of a new message. As stated earlier, to implement service personalization based on user's location, the UMS must be context aware and also terminal-aware. Service adaptability and personalisation plays the key in developing a Message-On-Demand UMS. Such a system is user-centric and is pervasive. To implement such a Message-On-Demand service, the central gateway technique that the current UMs follow lacks the basic needs of being pervasive and not user-centric. Hence a system that is user-centric and pervasive can be realized by using a decentralized design rather than a gateway approach. Such a system is terminal-aware (user devices) and is context-aware, hence knowing the needs of the user providing message delivery in the most appropriate fashion possible based on user's demands. It also fulfils the goal of UMS to deliver any messages any where, any time in the most appropriate manner possible.

III. DECENTRALIZED UNIFIED MESSAGING SYSTEM DESIGN AND ARCHITECTURE

A. System Overview

Figure 3 illustrates the overall architecture of the proposed decentralized UMS (DUMS). The DUMS is a peer to peer architecture. Each device can initiate and accept a connection request. The system receives incoming messages, notifies its peer devices about the messages, gets a response from the peer devices and replies to the sender.
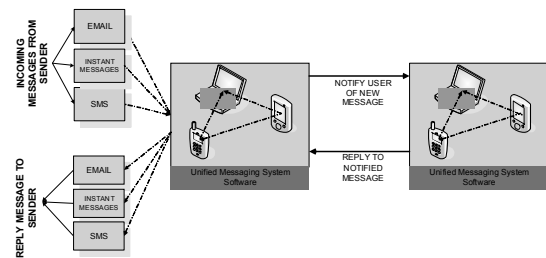

Figure 3: Overview of decentralized UMS architecture

As we can see, the system obtains messages from external sources, processes them and delivers to the user depending on user's context and end-terminal information. The devices work within the VPAN delivering messaging service on demand to the user in the most appropriate manner possible. The decentralized architecture provides the option for the user to reply to a message from any of the end-terminal irrespective of the message origination type. The DUMS takes

care of the message conversion before dispatching it.

B. Architectural Description

Figure 4 illustrates the DUMS architecture. The system, as seen, is split into a number of blocks, each of which performs a specific function. The entire system constitutes the DUMS.
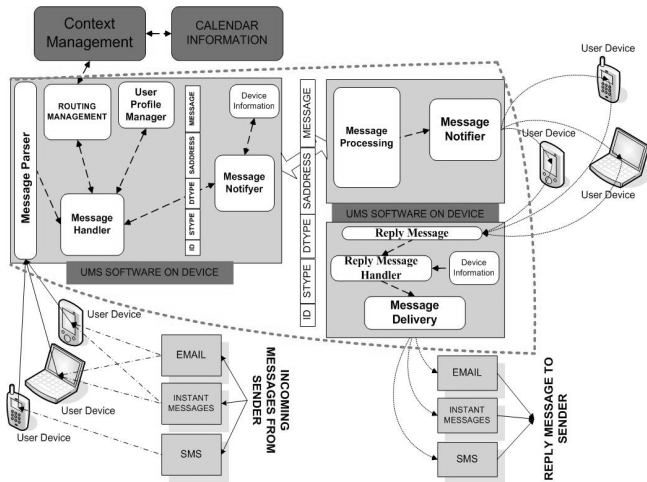

Figure 4: DUMS Architecture

The DUMS software on each device frequently checks for new messages. When a new message arrives, the incoming message handler and notifier is the first module that processes the message.

Message Parser

DUMS has the capability to convert messages from one format to another based on user's preferences and context. The message parser is a key component which can understand the various messaging systems message formats. It parses the message to obtain information like sender's address, phone number, message body, subject and other relevant information. The message parser also works on any new message notification/ response received from peer. In this case, it tries to retrieve information from the DUMS message header. The pseudo code in figure5 describes the function of the message parser.

```
if (incoming message = any message type)
then
    parse the information
    obtain sender information message body, subject
    forward this parsed information to the message
        handler
else if ( incoming message is of type DUMS
message)
    then
    obtain message data from the DUMS message
    forward this information to the message handler
```

Figure 5: Message Parser

Routing Management

The routing management is the component that obtains context information, user preferences and end-terminal information to decide on the most appropriate message format and the most appropriate device to be used for message notification. The routing management comprises a routing manager that obtains context information from a context manager that is external to the DUMS. The context information provides details on user's location, his environment details and end-terminal information. Context information can be obtained by using a number of technologies most of which are still under research. Since context by itself is a major field of research, this paper does not step into the development of a robust context management system.

Message Handler

The message handler handles the inputs received from the message parser and the routing management. The message handler is the one that is responsible for the Message-On-Demand service. It involves service personalization by taking into consideration users preferences. Once message handler receives the context and end-terminal information from the routing manager, it uses the input from the profile manager to obtain user preferences. Based on user's preferences and his/her context information, the message handler decides whether or not to notify the user of the message. The data flow between the message handler and the other components is illustrated in figure6.
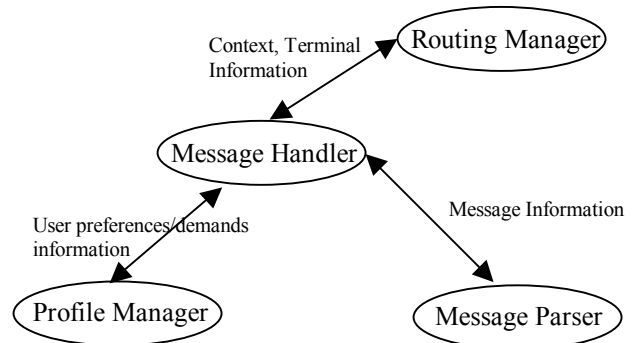

Figure 6: DFD for Message Handler, Parser and Routing Manager

Message Notifier

The message notifier handles two tasks. New message notification that needs to be passed on a peer device and notifications received from a peer device. The message notifier based on the inputs from the message handler takes into consideration, user's end-terminal device. The message notifier is responsible to establish a connection with the peer device, sending message notification to the peer device, waiting for an acknowledgement and disconnecting the connection. If the message acknowledgement is not received, the message notifier contacts the message handler to obtain information of any other device that is within the user's

environment. If the message notifier receives a message notification from the peer device, it takes into the consideration the device capability on which it is functioning and provides an appropriate notification to the user. The pseudo code illustrated in Figure7 describes the message notifier functionality.

```
Repeat
    if (message notification is PDA)    then
        notify PDA
        wait for ack.
         If ack = true
            return success
        else pass message back to message handler
    else if (message notification is smart phone) then
        notify smart phone
        wait for ack.
        If ack = true
            return success
        else pass message back to message handler
    else if (message notification is laptop) then
        notify laptop
        wait for ack.
         If ack = true
            return success
        else pass message back to message handler
while success
```

Figure 7: Message Notifier

Reply Message Handler and Delivery

The DUMS architecture provides a function that allows the user to reply to any incoming message irrespective of the device. This functionality is achieved by the Reply Message Handler and the Message Delivery components. The reply message handler is an integral component of the DUMS architecture it can handle any reply messages that the user wishes to send. The reply message handler obtains device information to check if the same device can be used to send the reply message. If it cannot use the same device, it then chooses the appropriate peer device which has the capability to send the message. For example, replying to an incoming SMS message notification as an SMS from the laptop. If the reply message handler decides to use a peer device to dispatch the reply, it uses the message delivery sub system to accomplish the task. The message delivery component uses reply message information from the reply message handler, communicates with the corresponding peer device and dispatches the user's reply message. The format of the reply message by default is the incoming message type. The pseudo code in figure8 describes the reply message handler and delivery functions.

```
Obtain and process reply message
Construct reply message format with appropriate
headers
Check reply message type
If require forwarding
Dispatch reply message to the peer device to be sent to
be dispatched to the message sender
else
dispatch reply message to message sender
```

Figure 8: Reply message handler and Delivery

C. DUMS Message Format

The system uses the DUMS message format for transferring messages between the peer devices. The DUMS message as illustrated in figure9 is made up of a set of predefined headers that encapsulates the original messages.

ID	STYPE	DTYPE	SADDRESS	MESSAGE

Figure 9: DUMS Message Format

ID: This represents the Message ID. The message ID is a unique identifier used to uniquely identify each message handled by the DUMS.
STYPE: This determines the source type of the message. The source type identifies in what format the message has actually arrived from the sender.
DTYPE: The destination type identifies in which format the message needs to be interpreted by the peer device. The DTYPE takes the same values as the STYPE. E.g., DTPYE will take value SMS if the destination device chosen to deliver the message is the users Smart Mobile Phone.
SADDRESS: This identifies the sender of the message.

IV. IMPLEMENTATION AND RESULT DISCUSSION

The DUMS architecture differentiates itself from other system by being distributed, decentralized and pervasive in nature. The system is aware of the user's devices hence, allowing the system to notify the user of new messages promptly regardless of location and time. The system is also adapts itself based on user's location and messaging preferences. The goal of the system is to send and receive any message anytime anywhere. The pervasive system requires information about user's context and environment. Context information varies depending on the environment in which the user can be in. E.g. In a meeting, at home or on the move like walking in a shopping mall or even driving a car. Since context information varies based on user location and environment, the application of DUMS is classified into a few scenarios based on the user's environment. These scenarios are:

At Home
In the office / meeting
On the Move such as driving a car

The main objective of the DUMS is to ensure prompt delivery of messages to the user in the most appropriate manner possible. It is a major challenge to implement a decentralized system. Since the design involves a number of technologies that are still under research and are at various stages of maturity, the proof of concept implementation is based on a few assumptions. Also, as discussed earlier, since context information can change depending on user's environment, the implementation to establish the DUMS architecture has been developed in specific to the Home Scenario. The system that has been developed works on the proposed DUMS architecture discussed in section3. This decentralized UMS called the intelligent-UMS (*i*-UMS) is pervasive, context-aware and provides Message-On-Demand service. The implementation of the prototype has been done using the Microsoft .Net framework. The reason behind choosing the .Net framework was, the devices, namely laptop, PDA and smart phone were all based on Windows operating system. Microsoft .Net provides rich programming api's for programming mobile devices and devices with restricted resources like the smart phone. The *i*-UMS developed was tested within the Home Scenario based on few real time situations. The following section will provide a detailed description of the Home Scenario and the various test case scenarios with along with the results obtained.

A. Home Scenario Description

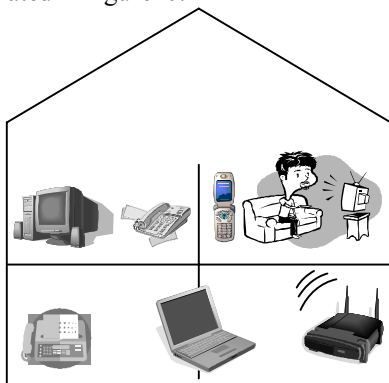Home Scenario is one of the typical cases of the usage of *i*-UMS illustrated in figure10.

Figure 10: Home Scenario VPAN

A simple case could be: The user might be in watching television having just the mobile phone with him/her. An instant message arriving on his/her laptop from an important contact is received and processed by the *i*-UMS. The system then contacts the location manager to obtain the location information of the user and the device nearest to the user. On obtaining this information, the *i*-UMS sends a message notification to the user's mobile phone based on user's preference. The user on receiving the message has the option of replying to the message through the mobile phone. The home scenario is VPAN described previously where devices are user-centric trying to deliver messaging services to the user irrespective of time, location and message type.

B. *i*-UMS Implementation

The *i*-UMS prototyped was tested on the following types of message types: email, Instant Message, voicemail and SMS. The *i*-UMS makes uses of Bluetooth, GSM, IEEE 802.11 based wireless networks and wired networks for communicating between the users end-terminals. The end-terminals used for the prototype implementation were the laptop, PDA and Smart Phone. The main implementation focus was to achieve the previously discussed goal of message delivery in the most appropriate format possible irrespective of time, location and message type. To achieve this goal, the implementation will discuss mainly on the two primary functions that is used to achieve this goal. They are message retrieval and message delivery. The following sections will provide the implementation details for the laptop, smart phone and the PDA.

Message Retrieval and Parsing

The *i*-UMS on the laptop retrieves messages from the user's mailbox. The mailbox can be a POP3 server or an IMAP server. This operation is performed by using a third part tool called the PowerTCP Mail for .NET [PT04]. The *i*-UMS also retrieves IM from MSN messenger. The working of the *i*-UMS on the laptop is very simple. Whenever the messenger is set in away mode, i-UMS recognizes that the user is not near the system. Hence, it immediately monitors incoming messages. Once it receives a new message, it then retrieves the message and passes it on to the message notification module as depicted by the code in figure11. The retrieval of IM from MSN messenger is done by using the MSN messenger class module of the .NET class library. It provides rich class libraries to query and retrieve MSN messenger status and IM [DM04]. The PDA can retrieve email, IM and can notify the laptop or the smart phone depending on the context information. The functionality of the *i*-UMS on the PDA for message retrieval is almost similar to that of the laptop with a few exceptions. In home scenario, the assumption is that the laptop is connected to the network and, hence, has the ability to check for new messages frequently. Since the PDA is a low powered device, the need for the PDA to check for new messages in this scenario is not necessary. But the PDA does have the option of checking for new messages which is more applicable in case of office or on the move scenario. The message retrieval on the smart phone involves the retrieval of SMS messages. *i*-UMS on the smart phone has the added ability to retrieve any new SMS message and then parse the SMS message into a DUMS message format. Figure12 provides code snippet for monitoring and retrieving SMS messages from the smart phone.

```
Public Class Messenger
    Dim WithEvents msn As New
    MessengerAPI.Messenger
    Dim wnd1 As
    MessengerAPI.IMessengerConversationWnd
    Dim contacts As MessengerAPI.IMessengerContacts
    Dim contact As MessengerAPI.IMessengerContact
End Sub
'Event to obtain instant message
'this is a turn around method that grabs incoming
message
Private Sub msn_OnIMWindowDestroyed(ByVal
pIMWindow As Object) Handles
msn.OnIMWindowDestroyed
    Dim wnd As
MessengerAPI.IMessengerConversationWnd
    wnd = pIMWindow
    contacts = wnd.Contacts()
    Dim str As String = wnd.History
    For Each contact In contacts
        username = contact.SigninName
    Next
    If msn.MyStatus <> 2 Then
        'Construct the DUMS message
        Dim msg As New UMSData
        UMSMSG.message = msg
        If str.IndexOf(msn.MySigninName) >= 0 Then
            Exit Sub
        End If
        UMSCollection.Add(UMSMSG)
```

Figure 11: IM retrieval

```
'Retrieves new SMS message
    Private Sub sms_InboxChanged(ByVal sender As
Object,
    ByVal e As
SmartSMS.NETCF.InboxChangedEventArgs)
    Handles sms.InboxChanged
        'TextBox1.Text = " "
        Dim Msg As SmartSMS.NETCF.SMSMessage
        Dim date1 As New DateTime
        Dim date2 As New DateTime
        Dim dt() As String
        For Each Msg In e.Messages
            Dim timeString As String =
            DateTime.Now.AddMinutes(-2)
            date1 = DateTime.Parse(Msg.Time.ToString)
            date2 = DateTime.Parse(timeString)
            If DateTime.Compare(date1, date2) > 0 Then
                messageDispatcher(Msg)
            End If
        Next
    End Sub
```

Figure 12: SMS retrieval

Message Notification and Replying

Message Notification is one of the important modules of *i*-UMS. *i*-UMS does a number of processing before it actually notifies the user of the incoming message. *i*-UMS first obtains the priority of the message based on the user's profile. It then contacts the context manager to obtain the location of the user and the device that is near to the user. Once *i*-UMS figures out the appropriate device to be notified, it then checks for the availability of the devices and the technology that can be used to communicate with the device. E.g., if the device to which the message needs to be forwarded to is a PDA, then *i*-UMS checks if the PDA is in Bluetooth coverage. If it is not, then it uses wireless IEEE 802.11 to communicate with the PDA. This intelligence built into the *i*-UMS, completely makes uses of the decentralized architecture which facilitates the device to move from one location to another, but still keep in contact with the other devices. Once the notification message is sent to the user's device, the message pops up on the user's device indicating the arrival of the new message. The user can view and reply the message on the device. Once again *i*-UMS on the device uses the same intelligence to find the best possible way to forward the reply. This message dispatcher process is described in figure13.

```
Public Class MessageDispatcher
    Public Function DispatchMessage(ByVal msg As
UMSMessage) As Boolean
        Dim route As New MessageRouting
        Dim response() As String
        Dim msgSend As New SendSocket
        response = route.getContextInfo("0").Split(",")
        If response(0) = "1" Then
            System.Console.Write("Notifying the PDA")
            Return msgSend.send(msg, response(1), 22000)
        ElseIf response(0) = "2" Then
            SmsMessage(msg)
            System.Console.Write("Notifying the smart
phone")
            Return True
        End If
    End Function
End Class
```

Figure 13: Message Notification/Dispatcher

The message notification process of the *i*-UMS is same for each device. The device communicates with the context manager to obtain the location information of the user and the device, and then checks on the user's profile to see the message priority and user preferences. Once it obtains this information, *i*-UMS decides on the device which needs to be notified and sends the notification. The user has the option of replying to the message from the handheld device. E.g., the user can reply to an IM notification arriving to his/her PDA from the PDA which sends the reply back to the laptop which in turn forwards the message to the message sender. The user

has also the option of replying to the message in any format. E.g., the user can reply to an IM as SMS message.

C. Implementation Results

i-UMS was implemented based on the Home Scenario environment discussed previously. The user devices that were used for implementation were the PDA, laptop and the smart phone. The laptop is an IBM T42 laptop that runs a Windows XP professional edition. The PDA is a HP iPAQ 2200 running a Windows Mobile 2003 Operating System. The smart phone is an I-Mate smart phone 2 running a Windows Mobile 2003 smart phone Edition OS. The laptop, PDA and smart phone are all Bluetooth enabled devices. The test environment uses the context manager that is assumed to be a web service that provides *i*-UMS with the context information of the user and the user's devices. Figure14 illustrates a state transition diagram of the *i*-UMS.
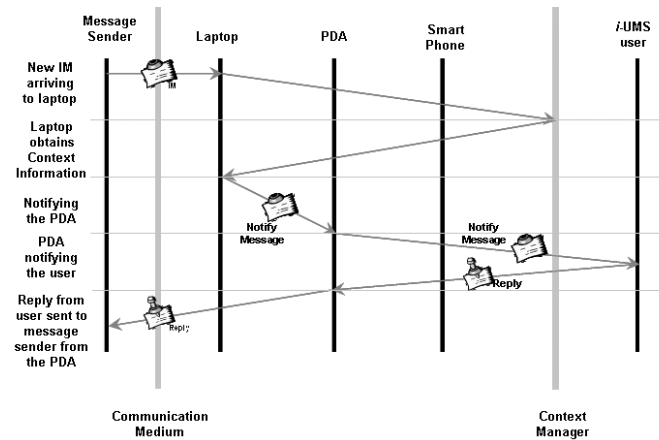

Figure 14: *i*-UMS state transition

1) Scenario 1

In this test Scenario, *i*-UMS demonstrates its message retrieval and notification functionality for an incoming instant message received from MSN Messenger. The laptop receives the new instant message, communicates with the context manager to obtain user's context information. It then obtains user preferences from the user profiler. Once the system obtains this information, it makes a decision on whether to notify the user or not. Based on the outcome of the decision, it chooses the most appropriate device to be notified based on context and user preference information. The user on receipt of the message can reply to the message from the device. The reply is then forwarded to the laptop or forwarded to the sender of the message based on the reply type. Figure15 and figure16 provides a few screenshots of the results obtained for scenario 1. In this test the destination device chosen was the PDA.
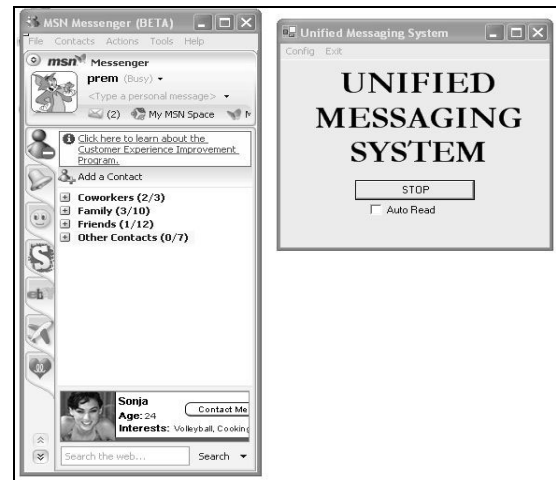

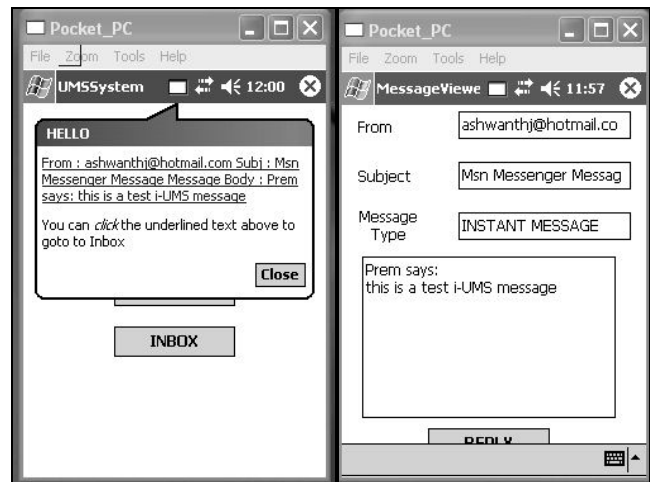**Figure 15: *i*-UMS on laptop - Instant Messenger (Busy State)**

Figure 16: *i*-UMS on PDA – Receiving Notification

2) Scenario 2

Scenario 2 describes the functionality of i-UMS on the smart phone and how the system responds to any incoming SMS messages. The smart phone can retrieve SMS messages and based on user's context and user preferences, it notifies its peer device. *i*-UMS does not notify all the messages. It retrieves the messages and based on context, decides on the most appropriate device to be notified. Figure17 and figure18 provide the screen dumps of the results obtained in Scneario2. The peer device that needs to be notified in this case is the Laptop.
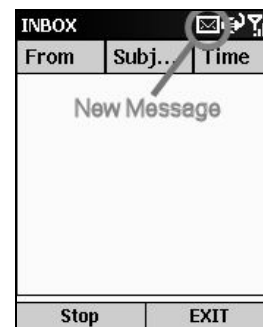

Figure 17: *i*-UMS on Smart Phone retrieving New SMS Message
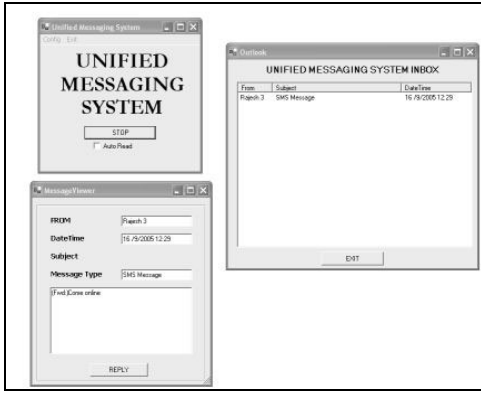
Figure 18: *i*-UMS on Laptop receiving SMS message notification

D. Evaluation Results

The evaluation of *i*-UMS is mainly based on its efficiency to handle incoming messages and message responses from the user. The *i*-UMS has been evaluated based on the following calculations.

Total Delivery Time (**T$_d$**):

$$T_d = \sum_{m=1}^{n} f\,(M_{ret} + M_{prcs} + M_{delivery})$$

M$_{ret}$ = Time to retrieve message in seconds
M$_{prcs}$ = Time taken for message processing in seconds
M$_{delivery}$ = Time taken to deliver message to the user's device in seconds
n = Total number of messages.

Message Retrieval Time(**M$_{ret}$**):

$$M_{ret} = \sum_{m=1}^{n} T_f\,(M_{access} + N_c)$$

T$_f$ = Time Function
M$_{access}$ = Message access time. Time to access the message store in seconds
N$_c$ = Network/ Communication Medium delays.

Message Processing Time (**M$_{prcs}$**)

$$M_{prcs} = \sum_{m=1}^{n} T_f\,(M_{analysis} + T_{context} + T_{UP} + T_{decision})$$

M$_{analysis}$ = Time to parse and analyze the message in seconds
T$_{context}$ = Time to retrieve context information in seconds
T$_{UP}$ = Time to retrieve user preferences in seconds
T$_{decision}$ = Time taken to process context and user preferences to arrive at a message delivery decision in seconds.

Message Delivery Time (**M$_{delivery}$**)

$$M_{delivery} = \sum_{m=1}^{n} T_f\,(T_{MT}) + T_f\,(M_{peer\ processing})$$

T$_{MT}$ = Time for peer to respond for message transfer.

M$_{peer\ processing}$ = Time taken by the peer to obtain message notification and notify user.

The following calculation illustrates the time taken to process a reply message that is created and sent by the user. The time is measured in seconds and it considers the time taken to process the reply and dispatch the message and does not include the external time factor required to deliver the message to the message recipient.

$$T_{Message\ Reply} = \sum_{m=1}^{n} f\,(P_{reply} + T_{dispatch})$$

P$_{reply}$ = Time to process the reply from the user
T$_{dispatch}$ = Time taken to dispatch the reply message(s) to the message sender. This might involve another time cost in using a peer device to deliver the message.

In Scenario 1, the system retrieves message from instant messenger and based on context and user preference information notifies the PDA of the new instant message. The total delivery time T$_d$ calculated in this case was 11 seconds for a single message. When the system was tested for multiple messages from m=1 to 60, the average time to deliver messages increased to 13.5 sec per message. The M$_{ret}$ is the time involved in retrieving the message which was less than a second for a single message. The message processing which involved the message analysis, retrieval of context and user information and the delivery decision took 2 seconds for a single message. Finally the message delivery which involves the time to establish connection, transfer the message and the time for the message to be processed by the peer took around 10 seconds or a single message. This was the same in case of Scenario2. Since the notification device was the laptop in Scnerario2, the message processing time on the laptop was much faster when compared to the PDA. Td in this case summed up to 9.8 seconds for 1 message in queue and increased to 11.5. The results of the time to message ratio is graphically represented in figure19.
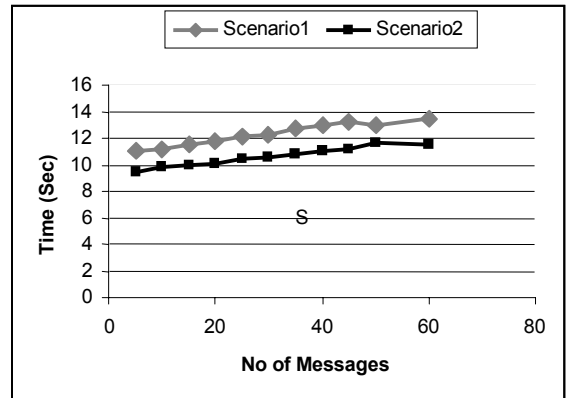

Figure 19: Performance results of i-UMS Message-Handling and Delivery Times.

V. Conclusion

Unified Messaging is not a completely new concept. But Message-On-Demand service is now the key one in developing pervasive messaging systems. The advent of new technologies like WLAN, Bluetooth and hardware like Personal Digital Assistant, smart phones have transformed Mark Weiser's[1] vision of pervasive computing into reality. The enormous growth of distributed computing has led to mobile computing which has allowed users to access and retrieve data irrespective of location [Sd03] [Rg00]. Such advancements in technologies have created the urge for systems that are pervasive in nature, i.e., systems that get integrated into user's environment and remain transparent.

Hence the ultimate aim of the UMS is no longer to integrate messages into one format but to deliver messaging service to users based on their demands and preferences. Service personalization and adaptation is one of the key factors to establish Message-On-Demand service. The pervasive UMS also facilitates message notification based on user's context. The system is end-terminal aware and is context aware. This kind of a service personalization can be made feasible in a decentralized design where the user's end terminals are intelligent enough to obtain user's context information. They also react to incoming messages based on the user's context. This paper has proposed DUMS, a decentralized UMS architecture that is pervasive, context-aware and user-centric. To realize the proposed DUMS architecture, the paper has presented *i*-UMS, a decentralized UMS based on DUMS architecture. The *i*-UMS has been implemented within the VPAN environment of the home scenario. Through the implementation, we have proved the feasibility of such a decentralized messaging system.

The implementation of *i*-UMS is based on a number of assumptions. These assumptions are necessary due to the primitive nature of some technologies used to realize such a system. Hence the Message-On-Demand in a decentralized UMS has still got a lot of doors open for further research. There are several extensions that we have identified to the proposed model that can render it working in the real world. Some of them are Context Awareness, Speech recognition and implementation of *i*-UMS based on DUMS architecture for the other scenarios. *i-UMS* is a next generation UMS which is pervasive, context-aware and user-centric.

References

[As99]   Arbanowski, S.; Van der Meer, S, Service personalization for unified messaging systems. Proceedings. IEEE International Symposium on Computers and Communications, ISCC'99, Read Sea, Egypt, 1999 July 6-8; Page(s): 486

[Av04]   AVST, Unified Messaging in Today's Business Environment A Guide to Voice, Fax and Email Anytime, Anywhere. Vendor White Paper, 2004 September; Page(s): 20

[Cu04]   Cisco Unity, Product Literature, [homepage on the Internet] Cisco Systems © 1992—2005 [Cited 20 January 2004]. Available at: http://www.cisco.com/warp/public/cc/pd-/unco/un/prodlit/

[Db02]   Declan Barber, GlobalCom: A unified messaging system using synchronous and asynchronous forms. Proceedings of the inaugural conference on the Principles and Practice of programming, and Proceedings of the second workshop on Intermediate representation engineering for virtual machines, Dublin, Ireland. 2002; Page(s): 141-144. Work in progress

[Dc02]   Chalmers, D.; Contextual Mediation To Support Ubiquitous Computing, In PhD Thesis, Imperial College London. 2002.

[Dh03]   Daniela Horn, The Importance of an All-in-OneCommunication Adapter in a Unified Communications Solution. Eicon Networks, C 2003. Available at http://www.eicon.com/worldwide/solutions/docs/UM_WP.pdf

[DM04]   DotMSN, [homepage on the Internet], Xih Solutions (c) 2003-2005. [Cited November - December 2004] Available at http://www.xihsolutions.net/dotmsn

[IA01]   InfoActiv, Inc. Increasing Demand for Unified Communications. 2001, February 1: Page(s):11. Available at: http://itresearch.forbes.com/detail/RES/992971989_481.html

[Lh03]   Larry Hettick, Inside Unified Messaging: A comprehensive shopper's guide. Nortel Networks. Copyright© 2003. Available at http://www.nortelnetworks.com/products/01/callpilot/collateral/nn106 800-021104.pdf

[Mj04]   Michael James, Liz Rice. An Introduction to Unified Messaging. Internet Applications Group, Data Connection Limited, 2004 August.

[No05]   NORTEL, Nortel CallPilot Unified Messaging, Product Brief, 2005 Available at:http://www.nortel.com/products/01/callpilot/collater al/nn101801-041105.pdf

[PT04]   Power TCP Mail for .NET, [homepage on the Internet] Dart Communications, © 2004. [Cited November - December 2004]. Available at: http://www.dart.com/dotnet/mail.asp

[Rg00]   Grimm, R.; Anderson, T.; Bershad, B.; and Wetherall, D.; A System Architecture for Pervasive Computing. Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system, Kolding, Denmark. 2000; Page(s): 177 – 182

[Sd03]   Saha D, Mukherjee A. Pervasive computing: a paradigm for the 21st century. IEEE Publication, 2003 March.

[Vd99]   Van der Meer .S, Arbanowski. St, Magendanz T, An approach for a 4th generation messaging system. Proceedings. The Fourth International Symposium on Autonomous Decentralised Systems, Tokyo, 1999 March.

[Vs00]   Van der Meer .S, Arbanowski. St, Steglich. St, Flexible control of media gateways for service adaption. Proceedings of IEEE Intelligent Network Workshop, 2000 May

[Wj03]   Wams, J.M.; van Steen, M, Pervasive Messaging. Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom 2003). 2003 23-26 March; Page(s): 499 - 504

[Wj04]   Wams, J.-M.S.; van Steen, M. Unifying user-to-user messaging systems. IEEE Internet Computing. 2004 March-April; Page(s): 76 – 82

---

[1] Source from Saha D, Mukerjee, Pervasive computing: a paradigm for the 21st century