Developing Mobile Services for Joomla! 1.5

Master of Science Thesis in the Programme Software Engineering and Technology

Patrik Strömvall Lundqvist
Malte Wannerskog

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, June 2009

Developing Mobile Services for Joomla! 1.5

Patrik Strömvall Lundqvist
Malte Wannerskog

Examiner: Sven-Arne Andreasson

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2009

Abstract

This master thesis has led to the development of a tool Mobile Base for Joomla! which offers a library of functions as well as a graphical interface. This tool is easily installed on a fresh Joomla! installation and can be used by persons that want to develop extensions for Joomla! which should offer mobile services such as sending and receiving SMS and MMS messages.

Mobile gateways can be used in order to send and receive SMS and MMS messages. By using the functions provided by the library included in Mobile Base a developer does not have to understand how to use mobile gateways, instead he can focus on the unique functions that he wants to develop.

The library offers functions to send basic text and multimedia messages as well as more advanced messages such as binary text messages and messages that configure the settings in a mobile phone according to the Over-The-Air Client Provisioning standard issued by the Open Mobile Alliance.

Keywords: Joomla!, Mobile Phone, SMS, MMS, Client Provisioning, Mobile Gateways

Sammanfattning

Det här examensarbetet har lett fram till utvecklandet utav ett verktyg Mobile Base för Joomla! Som dels erbjuder ett bibliotek med funktioner samt ett grafiskt gränssnitt. Verktyget kan enkelt installeras på en ny Joomla! Installation och kan användas utav personer som vill utveckla tillägg för Joomla! som ska erbjuda mobila tjänster som att skicka och ta emot SMS och MMS meddelanden.

Mobilgateways kan användas för att skicka och ta emot SMS och MMS meddelanden. Genom att använda funktionerna tillängliga i biblioteket i Mobile Base behöver en utvecklare inte sätta sig in i hur man använder mobilgateways utan kan istället fokusera på de unika funktioner som han vill utveckla.

Biblioteket erbjuder dels funktioner för att skicka enkla text- och multimediameddelanden men även mer avancerade meddelanden som binära textmeddelanden samt meddelanden som konfigurerar inställningarna i en mobiltelefon enligt standarden Over-The-Air Client Provisioning satt utav Open Mobile Alliance.

Nyckelord: Joomla!, Mobiltelefon, SMS, MMS, Client Provisioning, Mobilgateways

Preface

This master thesis is the result of our studies at Chalmers university of Technology were we have specialized in Software Engineering.

We would like to thank Crepido Systems AB for giving us the opportunity to work with them on this thesis.

Our Special thanks go to Michael Walenius who is the regional manager at Crepido Systems AB. He has been our mentor during this work and it is his ideas that have made this work possible. During the months we have spent on this work he has always supported our own ideas and thoughts and it has been a great pleasure to work with him.

Contents

1 Introduction

1.1 Background

There are several ways in which a mobile device can communicate with a server. Examples are:

- Trough a web browser.
- Trough a client application installed on the mobile device.
- Trough plain text messages (SMS) and multimedia messages (MMS).

While they are all reasonable solutions, using SMS and MMS messages has two advantages over the others. It doesn't require that the mobile device is connected to the internet, and it is safe to say that the majority of the mobile devices on the market today are capable of sending and receiving these messages. In addition to a mobile device it is necessary to have a server capable of sending and receiving SMS and MMS messages. Preferably these services should be available trough an user-friendly graphical interface.

1.1.1 Joomla!

Joomla! is a web content management system, it's open source and free for everyone to use. It allows a user with no or little knowledge about software development to easily manage a web site. The functionality of a Joomla! site can easily be extended by using different extensions; components, modules, plug-ins and templates.

In order to install Joomla! 1.5 an Apace HTTP server capable of running PHP Hypertext Preprocessor (PHP) and a My Structured Query Language (MySQL) database is required.

Joomla! is divided into a front-end and a back-end. The front-end contains the content of the website that is accessible for the regular user.

The purpose of the back-end is to provide a section for administrators of the site to use. From here an administrator can control the content that will be displayed in the front-end as well as change the behaviour of the various extensions that are used on the site.

Components
Components are the most essential extensions; they contain the main content of a page. Joomla! is designed to run one component for each page. A component could for example be an online gallery, forum or an e-commerce system.

Modules
Modules are used for smaller tasks; any number of modules can be displayed on a page. A module can be a stand-alone extension or it could work in conjunction with a component. Usually a module is used for one simple task such as displaying data.

Plug-ins
A plug-in is code that changes the core functionality of the Joomla! framework. Plug-ins can be used when a piece of code is needed throughout a site. Common usage is to format the output of a component or a module.

Templates
The graphical presentation of a Joomla! site can easily be changed with the use of templates. A template defines everything that has to do with the appearance of the site. A template contains at least one HTML file defining the structure of the site, and one CSS file for the design. It defines where on the page different extensions should be displayed, as well as the graphical representation of them.

Application Framework
Software developers that want to develop their own extensions can take advantage of the application framework provided by Joomla!. Joomla! supports the Model-View-Control (MVC) design to control the logic flow in software.

- The model handles the data of the application.
- The View consists of the user interface.
- The Controller controls the flow of data in the application. It retrieves data from the model and sends it to the view to be presented to the user.

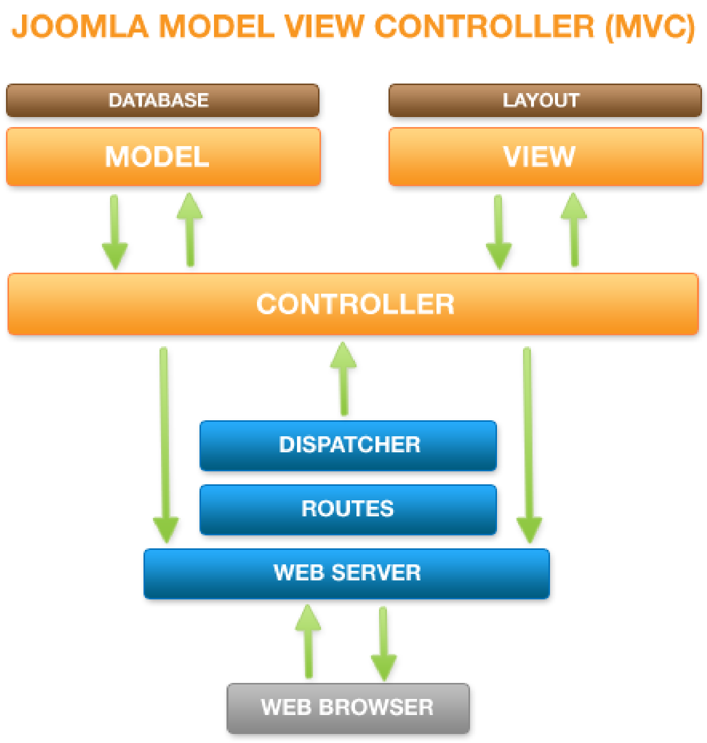


Figure 1.1: Joomla! MVC [6]

By using the framework provided by the Joomla! team it's easier for developers to produce well structured code that is easy to work with.

1.1.2 Mobile Gateways

It is possible to send and receive SMS and MMS messages from and to a computer system by using mobile gateways. There exist mobile gateways that provide this service at no additional fee except the cost of sending the SMS or MMS message. The mobile gateways often provide an API in order for developers to integrate the service in a software system.

1.2 Problem Description

The task is to develop a set of components and modules for Joomla!. These extensions will provide SMS and MMS functionality to a Joomla! site when installed.

In order to be able to provide mobile services, the developed extensions will use a mobile gateway. The gateway chosen for this task is Mobilstart [5].

The extensions that are to be developed must be capable of handling communication between mobile phones and the system in both directions. The system must be able to send a message to a mobile phone as well as being able to receive and process messages from a mobile phone. These messages can be both plain text messages (SMS) and multimedia messages (MMS).

It is important that this communication is done in a secure way.

The system must be able to handle payment for both incoming and outgoing traffic. There exist several methods for this; the problem is to find the most suitable solution of payment for both incoming and outgoing traffic.

When a user sends an SMS or MMS message to the system there should be a way to recognize which user it is depending on the telephone number.

1.3 Purpose

The result of this work is a community site where users will have access to different mobile services. It will be possible for an administrator to change the fee for a certain service, when a user uses a service that is not free he will be charged for this. These mobile services will be available through the use of various extensions. It will be possible to install one or several of these extensions on another site in order to easily provide mobile services on that site

By developing these extensions for Joomla! we will gain a deeper knowledge about Joomla!.

Furthermore the developed extensions will either be of commercial use for Crepido Systems, or can be used internally within the company.

1.4 Scope of Work

The focus of this master thesis is to make an attempt to find a good way to integrate mobile services into Joomla!. While there exists a wide variety of web content management systems we will only consider Joomla! for this master thesis. The main reason for this is that Crepido Systems uses Joomla!. We will not perform an analysis of Joomla! or compare it to other web content management systems.

We are going use existing solutions for sending and receiving SMS and MMS messages by using mobile gateways. While it is possible to connect directly to operators this would be expensive and require more work and is thus not suitable for this master thesis.

2 Analysis and Methodology

2.1 Joomla!

Before we could begin to analyze how to best integrate mobile services into Joomla!, we first had to learn extension development in Joomla!, as well as familiarize ourselves with Mobilstart.

By reading the book Learning Joomla! 1.5 Extension Development [2] we learnt how to develop modules and components for Joomla!. In order to be able to develop good extensions it is important to learn the application framework provided by Joomla!. This allows us to create extensions that have a good structure and that are easy to extend with additional functionality if needed.

In addition to the book, the Joomla! documentation provides excellent examples that was of great help. In particular it contains a very good tutorial of how to develop components that implements the MVC pattern [7].

2.2 Mobilstart

In order to use Mobilstart we first had to register an account on Mobilstart. Once we had done this we could start using their services. Every account on Mobilstart shares the same shortnumber that messages are sent to and received from, Mobilstarts' shortnumber is 72500. This means that when we use Mobilstart to send a message, 72500 will be the number listed as sender.

Mobilstart provides a set of APIs which makes it possible to integrate mobile services in your own webpage [17, 18, 19]. The first step we took was to familiarize ourselves with these APIs by developing some basic scripts.

2.2.1 Send SMS

Mobilstart provides a HTTP POST service which is used to send SMS and MMS messages.

When sending an SMS the following parameters are mandatory and have to be included in the HTTP POST that is sent to the service;

- **msisdn**: Phonenumber of the mobile device that the message is being sent to.
- **msg**: Data or Text.
- **shortnumber**: The shortnumber assigned by mobilstart.
- **username**: Mobilstart username.
- **password**: Mobilstart password.

In addition to the mandatory parameters there are some optional parameters that may be included;

- **sendername**: Name of the sender.
- **destport**: Destination UDHPort.
- **originport**: Originator UDHPort.
- **expiry**: Expiry time.
- **isdata**: If this parameter is sent, the msg parameter will be sent as data, otherwise it will be sent as regular text.

Example 2.1 illustrates how an SMS can be sent by using a simple HTML form.

```
<html>
<body>
<form action="https://extsms.bozoka.com/messaging/api/SmsCPost"
method="POST" accept-charset="utf-8">
Username: <input type="text" name="username"/><br/>
Password: <input type="password" name="userpassword"/><br/>
Shortnumber: <input type="text" name="shortnumber"/><br/>
Phone number: <input type="text" name="msisdn"/>
Text: <input type="text" name="msg"/><br/>
<input type="submit" value="Send SMS">
</form>
</body>
</html>
```

Example 2.1: HTML form for sending SMS through Mobilstart

When a message is sent using the service, an ID is returned. This means that the message is being processed; it does not guarantee that the message has been delivered to the mobile device.

2.2.2 Send MMS

Sending an MMS using Mobilstart is very similar to sending an SMS. The difference is that another HTTP service is used and different parameters are sent.

The following parameters are supported;

- **msisdn**: Phonenumber of the mobile device that the message is being sent to.
- **subject**: The ISO-8859-1 encoded MMS subject to be sent.
- **shortnumber**: The shortnumber assigned by mobilstart.
- **username**: Mobilstart username.
- **password**: Mobilstart password.

Additionally at least one file attachment must be sent.

Example 2.2 illustrates how an MMS can be sent by using a simple HTML form.

```
<html>
<body>
<form action="https://extsms.bozoka.com/messaging/api/MmsCPost"
method="POST" enctype="multipart/form-data" accept-charset="utf-8">
Username: <input type="text" name="username"/><br/>
Password: <input type="text" name="userpassword"/><br/>
Shortnumber: <input type="text" name="shortnumber"/><br/>
Phone number: <input type="text" name="msisdn"/>
Subject: <input type="text" name="subject"/><br/>
Attachment 1: <input type="file" name="test1"/><br/>
Attachment 2: <input type="file" name="test2"/><br/>
<input type="submit" value="Send">
</form>
</body>
</html>
```

Example 2.2: HTML form for sending MMS through Mobilstart

2.2.3 Receiving SMS and MMS

Receiving SMS and MMS messages can be easily achieved by using mobilForward which is an application provided by Mobilstart for forwarding SMS or MMS messages sent to 72500 to an HTTP server [19].

When a person sends a message to 72500 a *prefix* must be included in the message. Mobilstart uses this prefix to uniquely match incoming messages to a specific account. Each account has one prefix which can be changed by the owner of the account.

When using mobilForward an order code is added along with an associated script. An order code is a combination of the prefix and another unique code word. When a person sends a message to 72500 including an ordercode mobilForward will check for matching codes and forward the message to the associated script.

2.3 Sending SMS from Joomla!

In order to test how to integrate the functionality to send SMS messages from Joomla! we decided to develop two basic components.

The first was a component where users should be able to send SMS messages from the site to a mobile device. Since we follow the MVC pattern we wanted to let the view present the form to the user while the model sends the message to Mobilstart. This means that the message needs to be forwarded to Mobilstart directly from the code instead of using a form like in example 2.3. This can be achieved by using the PHP library cURL. cURL allows us to connect to and communicate with an HTTPS server located at Mobilstart [8].

Example 2.3 illustrates how an SMS message can be sent from PHP by using cURL.

```
<?php
$url = "https://extsms.bozoka.com/messaging/api/SmsCPost";
$ch = curl_init();
$parameters="username=myusername&userpassword=mypassword
&shortnumber=72500&msisdn=46703123456&msg=Message goes here";
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);
curl_setopt($ch, CURLOPT_POST, 1 );
curl_setopt($ch, CURLOPT_VERBOSE, TRUE);
curl_setopt($ch, CURLOPT_POSTFIELDS, $parameters);
$response = curl_exec($ch);
curl_close(\$ch);
?>
```

Example 2.3: sending SMS through Mobilstart with cURL

We decided that the second component should provide the functionality that users can subscribe to newsletters. An administrator should then be able to send an SMS message to all users who are subscribing. The structure for this component was very similar to the first one with the difference that this also featured an administrator interface in the back-end. Figure 2.1 illustrates the information flow for the component.

After the two components were finished we realized that the models for both components were very similar. Both components contained similar methods for forwarding the message to Mobilstart.

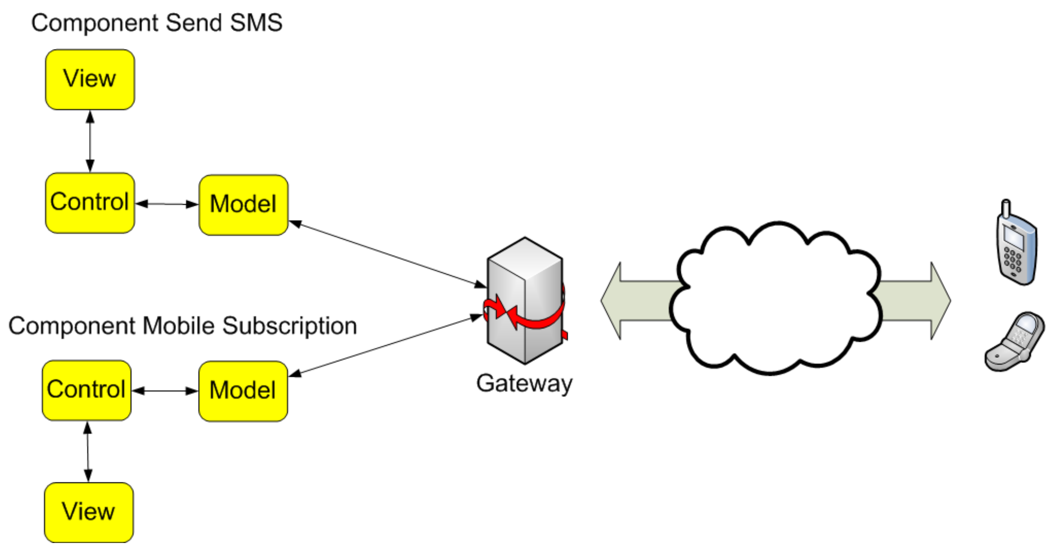


Figure 2.1: Shows how the components communicate with Mobilstart

This redundancy felt unnecessary and we got the idea to develop a library of functions that can be used when developing extensions that supports mobile services. This would increase the development speed for such extensions and save the developer a lot of time. Gathering all functionality for mobile services in a library will also eliminate the need for a developer to learn the gateway APIs, instead he can focus on the functionality of his extension.

2.4 Mobile Base

We decided to develop a component *Mobile Base* which would contain a library of functions. When a developer wants to create a new component that supports mobile services he can install *Mobile Base* and use the provided functions.

We decided that *Mobile Base* should provide the following functionality;

- Send SMS.
- Send MMS.
- Receive SMS.
- Receive MMS.
- Send Binary SMS.

- Send OMA OTA Client Provisioning settings.
- Handle different payment methods for services.
- Support several gateways.
- Log activity.

We decided to focus our work on the development of *Mobile Base* and then continue to rebuild the previously developed extensions as well as develop a set of new extensions which all use *Mobile Base*.

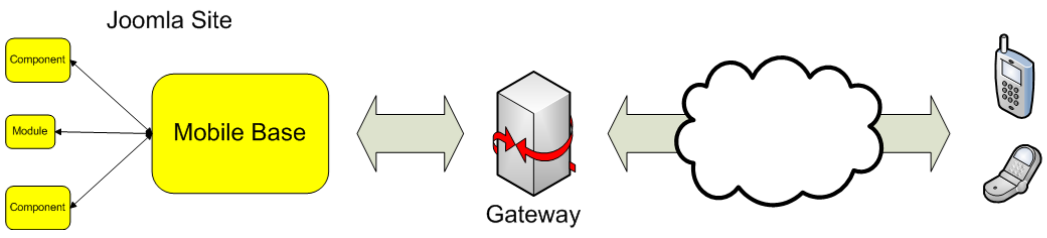


Figure 2.2: Shows how components communicate with Mobilstart through *Mobile Base*

2.4.1 Gateways

One functionality that *Mobile Base* should offer is to support several gateways. The reason for this is that gateways differ in financial aspects and they offer different services. Potential users of *Mobile Base* might find one of the gateways more suitable for their needs. We have chosen that Mobilstart and MO-SMS are the gateways that should be supported.

An administrator should easily be able to change settings of the gateways in the back-end of *Mobile Base* and also be able to specify which gateway to use by default. Another reason why we wanted to support several gateways is that it simplifies the possible future integration of additional gateways since *Mobile Base* already has the structure to support it.

Table 2.1 shows a comparison chart over the different services that Mobilstart and MO-SMS offers. MO-SMS is similar to Mobilstart when it comes to API's and technical usage.

	Mobilstart	**MO-SMS**
Send SMS	Yes (0,60 SEK)	Yes (0,50 SEK)
Send MMS	Yes (1,20 SEK)	Partially[1](1,20 SEK)
Send binary SMS	Yes (0,60 SEK)	No
Receive SMS	Yes (33% provision)	Yes (28% provision)
Receive MMS	Yes (33% provision)	Yes (28% provision)
Price range for incoming SMS and MMS messages	0-199 SEK	5-200 SEK

Table 2.1: Mobilstart vs MO-SMS comparison chart.

Security
Outgoing connections from *Mobile Base* to the gateways are secured over HTTPS, this protects against confidentiality-attacks since the connection is secure and the transferred data is encrypted.

Since incoming connections from a gateway to *Mobile Base* doesn't use username, password or any other identification procedure it can easily be manipulated and forged. The first step to protect against this is to check whether the IP address of the call matches the IP range of the gateway. Since an IP address can be spoofed we need an authentication process to provide the extra security, we achieve this by adding an identification in the call from the gateway. The authentication process checks whether the identification included in the call matches the one stored on the server. Since we now have included an identification in the incoming connection, the need for a HTTPS server has increased compared to before but it is not as important as for outgoing connections since the IP check still applies a fair amount of security. If either of the authentication or IP check fails the connection is ignored.

2.4.2 SMS and MMS Messages

By importing the library included in *Mobile Base* it should be possible for developers to call a function in order to send a message to a mobile phone. There should exist functions that can be used to send SMS, MMS and binary SMS.

[1]An incoming SMS or MMS must first have been registered so MO-SMS knows what operator that phone number uses.

When sending an SMS the parameter for the function should be the message and the phone number of the receiver. The parameters that should be passed when calling the function for sending an MMS are the subject of the message, the telephone number of the receiver, and the attachments of the MMS. When sending a binary SMS the parameters for the function should be the binary encoded message along with the telephone number of the receiver.

When receiving SMS and MMS messages from Mobilstart and MO-SMS the name of the incoming parameters differ. In order to make it as easy as possible for developers we wanted *Mobile Base* to receive these messages and process the data, and output a formatted message that will look the same for all SMS and MMS messages. By doing this the same parameter names will be used regardless if Mobilstart or MO-SMS is used.

2.4.3 Payment and Transaction Methods

Since the services provided by the gateways costs money, *Mobile Base* should be able to provide the owner of the site with the possibility to charge users for using the services on the site. This is solved by giving the users the possibility to transfer money to an account and fill a balance stored on *Mobile Base*.

A users' balance on *Mobile Base* is charged every time the user uses a service. An administrator should be able to set the global price for both SMS and MMS messages in the back-end of *Mobile Base*. It can be set to zero, which means that all services are free of charge. If a price parameter is set when a component calls *Mobile Base* the global price is overridden with the specified price. If a user doesn't have enough money on the account, he will be notified of this and he will not be able to use the service until his account is refilled with money.

It should be easy for a user of the site to transfer money to his account on *Mobile Base*. It should also be easy for the owner of the site to review transactions made by the users. Since money is involved it is important that the transaction is executed in a secure manner. When we discussed different solutions we decided that a good solution would be to use PayPal because PayPal will handle the actual money transfer and we will just handle the communication between *Mobile Base* and PayPal. This guarantees that the most critical part of the transfer is protected by the security of PayPal.

Since neither of us had worked with PayPal integration before this was a new experience but we had a pretty good idea on how the transaction should transpire; a user enters the amount he wants to transfer, then continues to PayPal where the actual money transfer takes place. When the transfer is completed the user is redirected back to our site and the system should now be notified about the transfer.

Buy Now Button with Instant Payment Notification

When investigating how the PayPal transaction should happen the first step we took was to find out how to implement a PayPal button on our site that initiates a payment. We found that the most logical solution to this step would be to add a *Buy Now* [11] button on the site. To add a Buy Now button to the site a button must be created on PayPal where the price and purchase item name is entered. PayPal then generates the code for the button that should go on the website [11]. The transaction flow for the Buy Now button is described in figure 2.3.

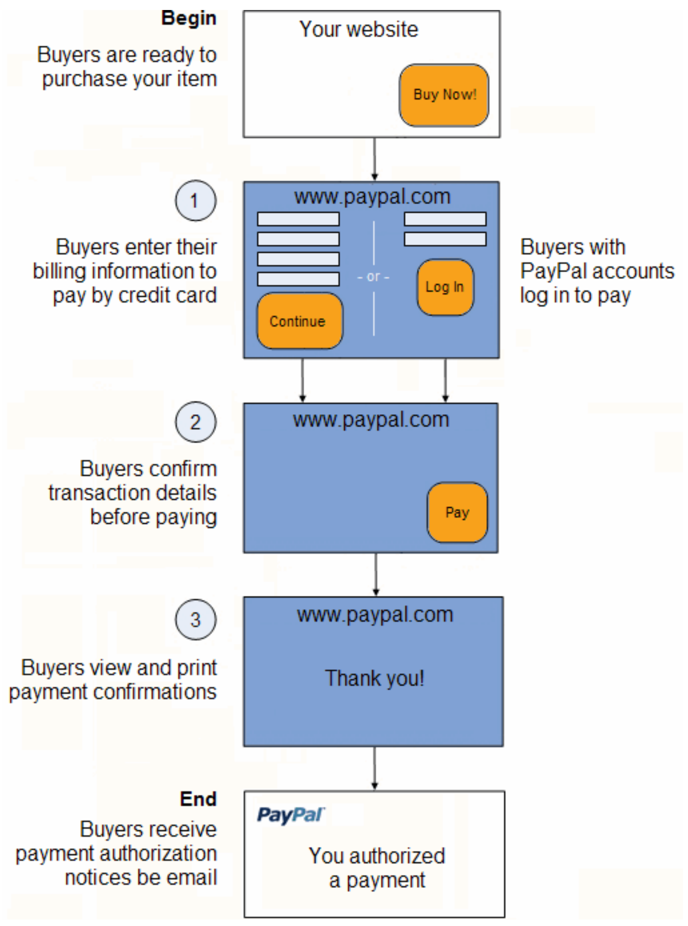

Begin
Buyers are ready to purchase your item
Your website
Buy Now!
1
Buyers enter their billing information to pay by credit card
www.paypal.com
- or -
Continue
Log In
Buyers with PayPal accounts log in to pay
2
Buyers confirm transaction details before paying
www.paypal.com
Pay
3
Buyers view and print payment confirmations
www.paypal.com
Thank you!
End
Buyers receive payment authorization notices be email
PayPal
You authorized a payment

Figure 2.3: PayPal Transaction: Checkout Flow with a Buy Now button [11]

This method also offers a return URL so a user can return to our site after a successful transfer [11]. While this is a good solution for our PayPal transaction implementation it does not completely fulfil our needs. What's missing is that *Mobile Base* has no idea that a transfer has occurred. To solve this problem we use a service offered by PayPal called Instant Payment Notification (IPN). The IPN service notifies *Mobile Base* whenever a payment transaction has been completed and informs who performed the purchase and how much money he or she transferred [12]. The IPN flow is shown in figure 2.4:

1. The payment is completed through the Buy Now button.
2. PayPal sends an IPN message to our system.
3. We respond with the exact same message. This is a security measure.
4. PayPal responds with either a success or failure message.

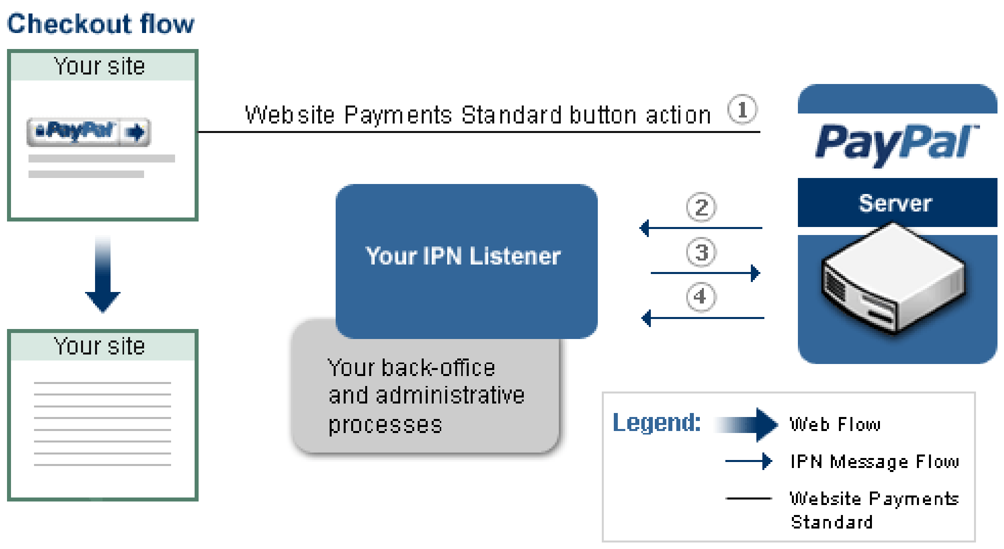

Checkout flow
Your site
PayPal
Website Payments Standard button action
1
PayPal
Server
Your IPN Listener
2
3
4
Your site
Your back-office and administrative processes
Legend:
Web Flow
IPN Message Flow
Website Payments Standard

Figure 2.4: PayPal Transaction: Checkout Flow with IPN Message [12]

With the Buy Now button combined with IPN we have a functional PayPal payment transaction; however this solution is insufficient to use in our system due to these issues:

1. The Buy Now button must have specific shop items and prices linked to it.
2. The Buy Now button must be generated at PayPal before use.

So even though this is an acceptable solution we wanted something that does not require any setup at PayPal except the creation of an account.

Express Checkout
After some more research and reading of the PayPal developer documentation we found a service called Express Checkout. Express Checkout offers more control over the different stages of payment and more customization; most important it solves the issues from the solution with the Buy Now button and IPN.

The Express Checkout has an API with several operations that, when used in conjunction with each other, allows us to control the payment and the overall transaction flow. We use three of the operations; SetExpressCheckout, GetExpressCheckoutDetails and DoExpressCheckoutPayment. [13].

The SetExpressCheckout API operation sets up the Express Checkout transaction. Here we can specify information to customize the look and feel of the PayPal site and the information it displays. The GetExpressCheckoutDetails obtains information about the buyer from PayPal. DoExpressCheckoutPayment completes and finalize the Express Checkout transaction. Each API call requires the caller to pass his or hers PayPal API account information.

Figure 2.5 shows the Express Checkout execution flow:

1. The SetExpressCheckout operation is the first to be called. Here the configuration is set to specify how the PayPal page flow should be; amount, currency, purchase description, where to go if the purchase has succeeded, where to go if the purchase is cancelled and since this is a digital purchase no shipping will be needed.
2. PayPal responds with a Token, which is a temporary identification number for this transaction. A Token's lifetime is approximately three hours.
3. The system redirects the user to PayPal with the Token as a parameter so that PayPal knows which payment that is to be processed.
4. The user then logs in and confirms the details and is redirected back to the system with the Token as parameter.

5. Here the user doesn't have to confirm the order because it has already been done at PayPal so there is no need. However we do call the GetExpressCheckoutDetails in order to get information about the transaction.
6. Finally a call to the operation DoExpressCheckoutPayment is made which finalizes the payment; either successfully or unsuccessfully depending on whether or not the security measures has been fulfilled.

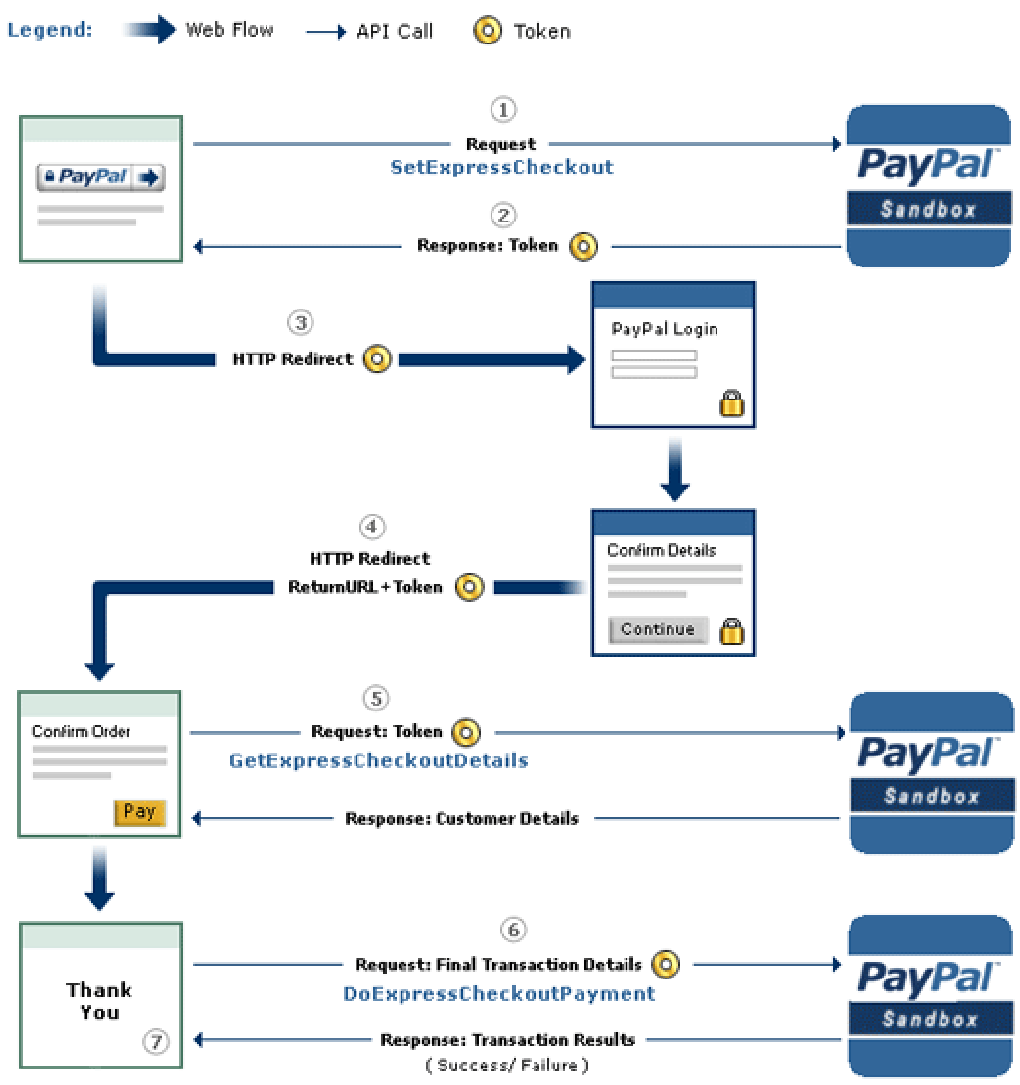


Figure 2.5: PayPal Transaction: Express Checkout Execution Flow [13]

Express Checkout fulfills the demands we had on PayPal payment transactions. All that is needed for Express Checkout is a PayPal account. No interaction with PayPal is necessary such as Buy Now button generation or IPN setups. It is also easy to use; both for developers, administrators and end-users.

Security
As seen in figure 2.5, the communication with PayPal travels back and forth like a real life conversation. This means that the system is exposed to Man-In-The-Middle (MITM) attacks. Even though the traffic from the system to PayPal is encrypted by HTTPS, we cant assure complete security. Even if both directions were encrypted by HTTPS we still couldn't assure complete security and MITM attacks would still be possible. The possible result of a MITM attack could be that the system could be fooled in to thinking that money was transferred when in fact no transaction occurred. Before we realised MITM attacks was possible we thought that a good idea would be that as soon as a transfer was completed the users account should be filled. The result of this could be that an attacker could fill his account with money and spend it on the site before he is caught. To eliminate the problem we changed the acceptance of money transfer to be done manually by administrators. The general idea here is that an administrator checks the PayPal account to make sure that an actual transfer was made. This would result in that administrators would put a lot of work in to checking the PayPal account continuously, but this is the only way to ensure complete security which we think is necessary.

2.4.4 Open Mobile Alliance Client Provisioning

The system should be able to send e-mail and internet settings to a mobile phone. Both of us were aware that this can be done, but we had no experience in the development of it. The end-user has a mobile phone which he wants to configure with e-mail settings. He can do this manually but if the user is not familiar with how to do this it's both easier and quicker to get the settings sent to the phone instead of entering them manually. The user visits a website and enters his settings such as username, password, mail-server, etc. which is then sent to his phone. Once the mobile phone has received the configuration the user accepts it and it is ready to be used.

We wanted *Mobile Base* to provide an easy way for developers to generate correctly formatted SMS messages that will configure e-mail and internet settings on a mobile phone. A developer should be able to create a HTML-form where the user can enter his desired settings. These values should then be forwarded to the correct function in *Mobile Base* where a correctly formatted SMS will be generated.

In order to understand how to generate such SMS messages we studied the documentation [14, 15, 16] provided by Open Mobile Alliance (OMA). The Open Mobile Alliance is a standards organization which develops open standards for the mobile phone industry. Its mission is best described on their website [3]:

"The mission of the Open Mobile Alliance is to facilitate global user adoption of mobile data services by specifying market driven mobile service enablers that ensure service interoperability across devices, geographies, service providers, operators, and networks while allowing businesses to compete through innovation and differentiation."

Client Provisioning
Client Provisioning (CP) is the process by which a mobile device is configured with a minimum amount of user interaction. The term covers Over-The-Air (OTA) provisioning. This provisioning process is initiated when a mobile device receives a special formatted SMS containing the settings [10, 14].

Over-The-Air Provisioning
In order to send settings Over-The-Air a provisioning document must be created. Provisioning documents are XML formatted documents that contains the settings that are to be sent via SMS. Since it's convenient to keep the size of an SMS limited the Open Mobile Alliance have created a specification that defines a compact binary representation of the XML language called WAP Binary XML (WBXML). The binary XML content format is designed to reduce the transmission size of XML documents, allowing more effective use of XML data on narrowband communications channels without any loss of functionality or semantic information. The binary format encodes the parsed physical form of an XML document, i.e., the structure and content of the document entities [15].

Provisioning Document Example
The XML Document Type Definition (DTD) defines two elements; a CHARECTERISTIC element and a PARM element. The PARM element is used to provide values for the individual parameters. The CHARECTERISTIC element is used to group parameters to a specific characteristic [14].

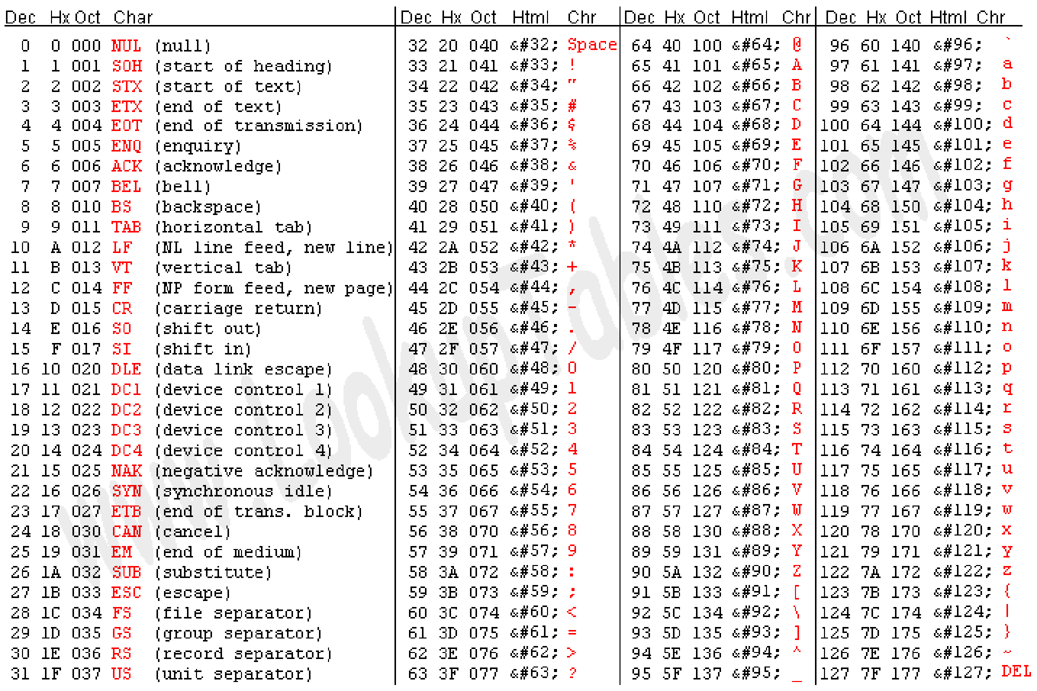
Example 2.4 shows how a provisioning document, which configures the *Data Account* on a Sony Ericsson mobile phone, could look like. The *Data Account* is used to connect to a network connection [10].

```
1. <wap-provisioningdoc version="1.1">
2.    <characteristic type="NAPDEF">
3.        <parm name="NAPID" value="NAP1" />
4.        <parm name="BEARER" value="GSM-GPRS" />
5.        <parm name="NAME" value="Data account1" />
6.        <parm name="NAP-ADDRESS" value="nap.address.com" />
7.        <parm name="NAP-ADDRTYPE" value="IPV4" />
8.     </characteristic>
9. </wap-provisioningdoc>
```

Example 2.4: XML Provisioning Document

1. Defines which version the provisioning document is.
2. Specifies that the data account is to be configured.
3. Sets the id of the data account.
4. Specify that the account uses GPRS to connect to the internet.
5. Sets the name of the data account.
6. Sets the address that is used to connect to a remote entity.
7. Defines the type of the NAP-ADDRESS.
8. End of characteristic tag.
9. End tag for the provisioning document.

As mentioned earlier a provisioning document is often encoded as binary XML before being sent to a mobile phone Over-The-Air. How to encode a provisioning document as binary XML can be found in the provisioning content specification [10]. Each mark-up element in the provisioning document is represented by a hexadecimal token. Strings are encoded as hexadecimal values where each character is represented by its hexadecimal value in the ASCII table [4].



Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH	(start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX	(start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX	(end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO	(shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI	(shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE	(data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1	(device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2	(device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3	(device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4	(device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN	(cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM	(end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB	(substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC	(escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS	(file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS	(group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS	(record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US	(unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Figure 2.6: ASCII table [4]

Table 2.2 shows the WBXML encoding of the provisioning document in example 2.4.

C5	Element <wap-provisioningdoc>
45	Attribute version...
03	...encoded as a string
31 2E 31	The string "1.1"
00	End of string
01	End of attribute list for <wap-provisioningdoc>
C6	Element <characteristic>
55	Attribute type NAPDEF
01	End of attribute list for <characteristic>
87	Element <parm>
11	Attribute name NAPID
06	End of attribute list for <parm>
03	Encode value as string
4E 41 50 31	The string "NAP1"
00	End of string
01	End of element <parm>
87	Element <parm>
10	Attribute name BEARER
06	End of attribute list for <parm>
AB	Encode value GSM-GPRS
01	End of element <parm>
87	Element <parm>
07	Attribute name NAME
06	End of attribute list for <parm>
03	Encode value as string
44 61 74 61 20 61 63 63 6F 75 6E 74 31	The string "Data account1"
00	End of string
01	End of element <parm>
87	Element <parm>
08	Attribute name NAP-ADDRESS
06	End of attribute list for <parm>
03	Encode value as string
6E 61 70 2E 61 64 64 72 65 73 73 2E 63 6F 6D	The string "nap.address.com"
00	End of string
01	End of element <parm>
87	Element <parm>
09	Attribute name NAP-ADDRTYPE
06	End of attribute list for <parm>
85	Encode value IPV4
01	End of element <parm>
01	End of element <characteristic>
01	End of element <wap-provisioningdoc>

Table 2.2: XML to WBXML conversion.

Example 2.5 shows the provisioning document from example 2.4 encoded as binary XML.

```
C5  45  03  31  2E  31  00  01  C6  55
01  87  11  06  03  4E  41  50  31  00
01  87  10  06  AB  01  87  07  06  03
44  61  74  61  20  61  63  63  6F  75
6E  74  31  00  01  87  08  06  03  6E
61  70  2E  61  64  64  72  65  73  73
2E  63  6F  6D  00  01  87  09  06  85
01  01  01
```

Example 2.5: WBXML provisioning document

When adding the functionality to generate a WBXML provisioning document our first step was to create a set of functions which is used to generate the XML provisioning document. This XML provisioning document is then passed to a function which converts it to a WBXML provisioning document. The biggest technical challenge with this was to perform the XML to WBXML conversion. We found a free library called libwbxml [9] but found that this did not fully suit our needs for the following reasons; First of all, not all web servers allow execution of external programs. Furthermore depending on which operating system the server runs on the correct compilation of libwbxml has to be used. We decided to create an XML to WBXML parser in PHP by following the WBXML documentation [15]. Since the parser is in PHP, every web server that is capable of running PHP can use it.

Security
To increase the security when creating a provisioning document, a security header must be added. This header contains the Hash Message Authentication Code (HMAC) and information about what security method that is in use. The HMAC is created from a SHA-1 algorithm based on the WBXML of the provisioning document content and a 4-digit key. The HMAC is then encoded as a string of hexadecimal digits where each pair of consecutive digits represent a byte [16].

2.5 Extensions

Once the first version of *Mobile Base* was finished we could continue to develop extensions for Joomla!.

First we made some changes to the two components we previously created so that they both use *Mobile Base*. This turned out to go very smoothly since, *Mobile Base* handles the mobile functionality, that part of the code in the components could simply be replaced by a call to the proper function in *Mobile Base*.

Once this was done we began to plan other extensions to develop. The purpose of these extensions are to demonstrate different mobile services, therefore we only specified their basic behaviour before we began the development. Extra features of these extensions are of lesser importance.

2.5.1 Intra SMS

We had already developed a SMS component but we felt the need for a more sophisticated SMS service where users can select one or more groups of users and send SMS messages to one or several recipients. This can be very useful for large or small organizations and can for example be installed in an intranet.

2.5.2 Send MMS

A component where a user can select an image, enter a subject and then enter a message and send this as an MMS to a specific phone number.

2.5.3 Send E-Mail and Browser Settings

Today it is common that when a customer is in contact with customer support they can send an SMS following the OMA OTA Client Provisioning standard [14] which will automatically configure the mobile device with the desired settings. With this component a user can send an SMS which will automatically configure SMTP, IMAP and POP3 on a mobile device.

A user should also be able to send an SMS which will automatically configure browser, and internet settings on a mobile device. This can be used to configure a mobile device which currently is not configured to access the internet. After the configuration has been made the device will be properly configured and can access the internet.

2.5.4 Mobile Gallery

We wanted to a create a component where users has the possibility to send an MMS which will be received by the system, the component would then extract the image from the MMS and automatically add it to the gallery. If the user has registered his name along with his telephone number he will be displayed as the uploader of the image. It should also be possible to not allow anonymous uploads (where the user has not registered his name) to be added automatically to the gallery. This allows an administrator to first review an uploaded images and eliminate the risk that users upload inappropriate content.

2.5.5 Mobile Quote

Users should be able to send an SMS to the system. The system will then reply with a famous quote from the component database.

3 Results

3.1 Extensions

3.1.1 Mobile Base

Mobile Base is a special component, it provides a library of functions for developers to call in order to perform mobile tasks such as sending an SMS or MMS. It also acts like a regular component where users can transfer money through PayPal which can be spent when using mobile services.

Library
The library part of Mobile Base is a collection of functions that other components can call. The library provide the following functionality:

- Support for Mobilstart and MO-SMS
- Send SMS
- Send MMS
- Send binary SMS

- Send OMA OTA Client Provisioning settings
- Receive SMS
- Receive MMS
- Generate OMA OTA XML for IMAP, SMTP, AP, Internet and Browser settings.
- OMA OTA XML to WBXML conversion.

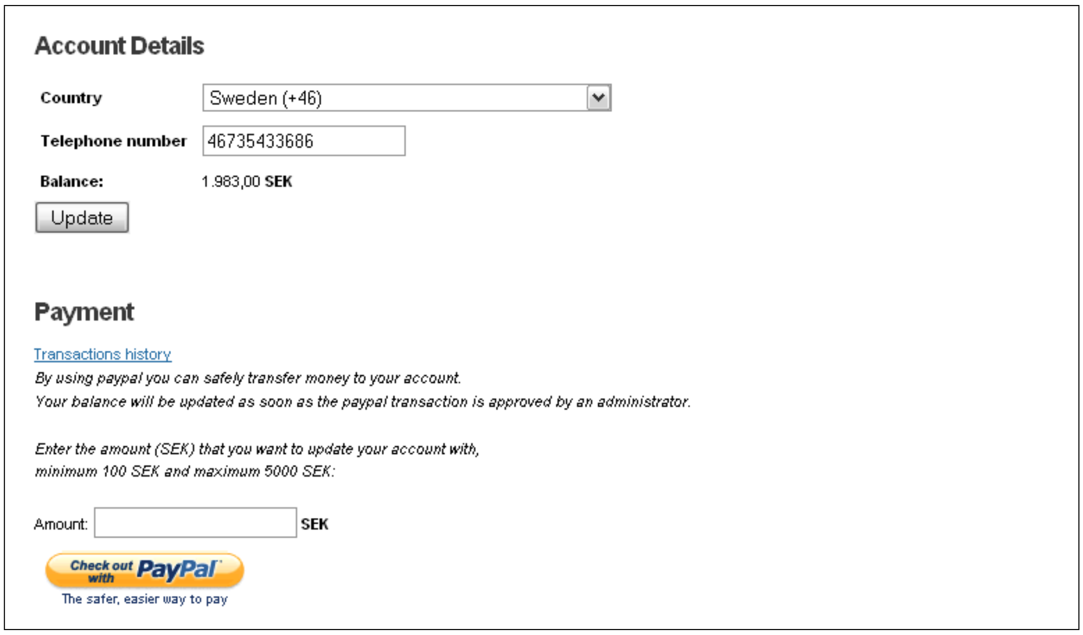
The functions of the library are connected to the gateway accounts that are set up in the back-end of Mobile Base. Each call to a send function takes a phone number, a price and message specific information as parameters. The message specific information varies depending on if it's an SMS, MMS or binary SMS. SMS needs a message, MMS needs a list of attachments such as text, image, audio or video, and binary SMS needs a WBXML string. Each call will log the phone number, date, price and type of service, it will also charge the user of the price. If the user doesn't have enough money on his account, an error code is returned and the message wont be sent.

The receive functions are a bit different, first a Forwarding Rule must be specified in the back-end. When a message is received, Mobile Base will extract the information and forward the call to the correct component depending on the Forwarding Rule.

The OMA OTA XML generation functions takes POST calls as input. The POST calls should contain information about what the mark-up elements should be set to. The functions return blocks of XML data so a user can build the complete XML document from blocks of his choice. The document needs a start-block and a end-block, everything in between is generated by the user.

The OMA OTA XML to WBXML conversion takes an OMA OTA XML formatted string as input and outputs the ready-to-be-sent WBXML string.

Front-End
In order to be able to use the services that cost money provided by *Mobile Base* a user must set up an account on *Mobile Base*. This is easily done by first selecting a country from the drop-down list and then by entering the phone number. Once the information has been registered the user can see his current account balance. If the balance is zero the user will not be able to use any of the services that cost money.


Account Details

Country	Sweden (+46)
Telephone number	46735433686
Balance:	1.983,00 SEK

Payment

Transactions history

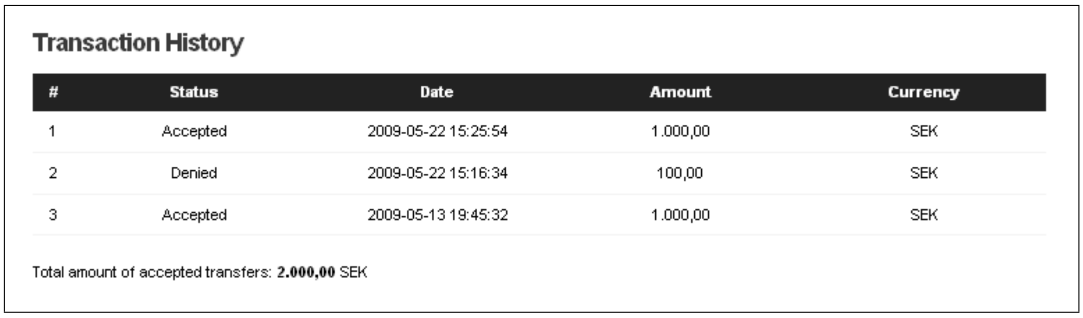
By using paypal you can safely transfer money to your account.
Your balance will be updated as soon as the paypal transaction is approved by an administrator.

Enter the amount (SEK) that you want to update your account with,
minimum 100 SEK and maximum 5000 SEK:

Amount: SEK

Figure 3.1: Mobile Base Front-end - Personal Settings

Payment is handled by PayPal; by entering an amount and clicking on the checkout button the user is directed to PayPal where he fulfils the payment over a secure channel. When the payment is completed the user will be redirected back to the system and an appropriate message will be displayed whether the transaction was successful or not. The user can also see his transactions history and the total amount of money transferred. Once the transfer has been approved by an administrator the users' account balance will be updated.


Transaction History

#	Status	Date	Amount	Currency
1	Accepted	2009-05-22 15:25:54	1.000,00	SEK
2	Denied	2009-05-22 15:16:34	100,00	SEK
3	Accepted	2009-05-13 19:45:32	1.000,00	SEK

Total amount of accepted transfers: **2.000,00** SEK

Figure 3.2: Mobile Base Front-end - Paypal Transactions

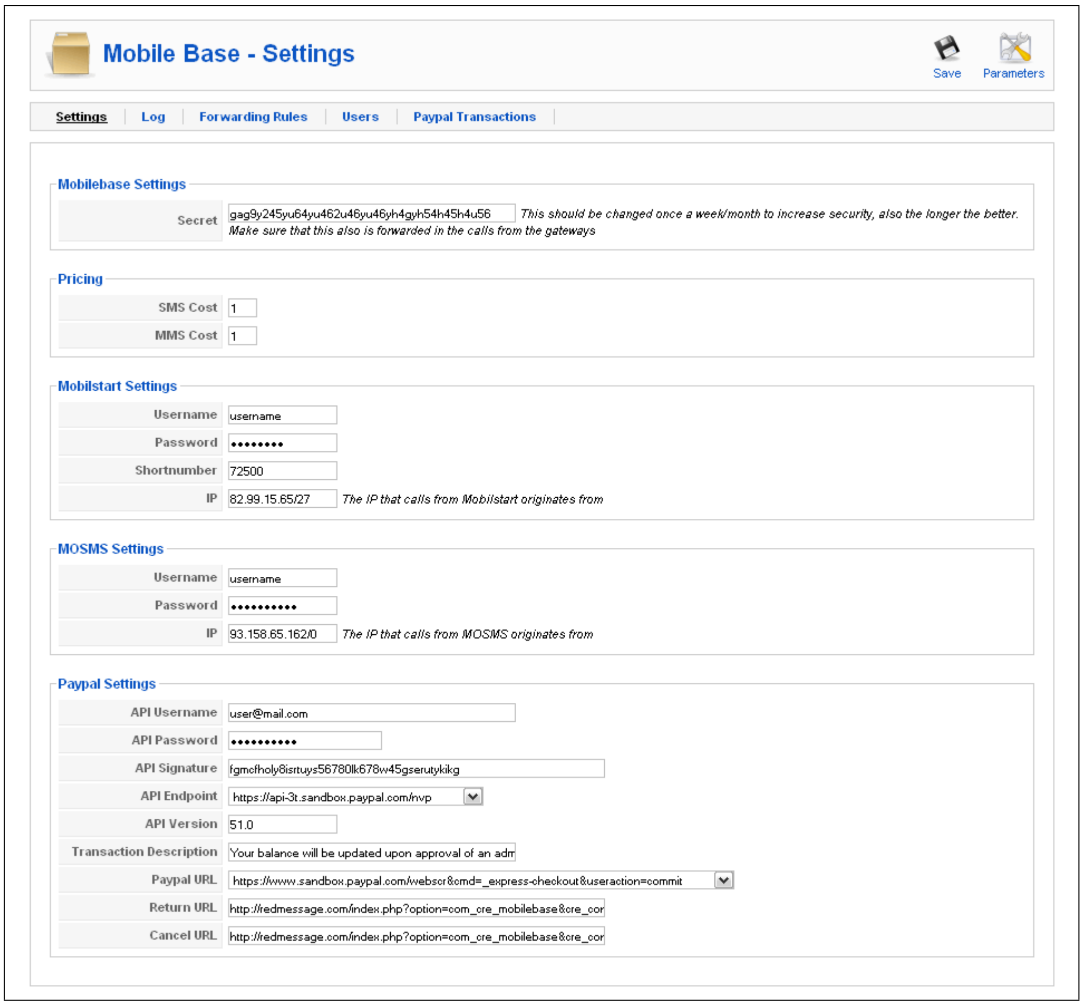
Back-End
The back-end provides five interfaces; *Settings*, *Log*, *Forwarding Rules*, *Users* and *Paypal Transactions*.

In the settings section an administrator can change the global configuration that will be used by Mobile Base. Here an administrator can choose which gateway that will be used as default, Mobilstart or MO-SMS. He must also enter his account information for the gateway that he intends to use.

In order to increase the security a secret is used when a message is forwarded from a gateway to mobile base.

The cost for outgoing SMS and MMS messages are also set, if no price is specified for a service this is the price that will be used.

In order to be able to use PayPal to handle payments the correct settings for PayPal must be entered.



Mobilebase Settings

Secret	gag9y245yu64yu462u46yu46yh4gyh54h45h4u56 *This should be changed once a week/month to increase security, also the longer the better. Make sure that this also is forwarded in the calls from the gateways*

Pricing

SMS Cost	1
MMS Cost	1

Mobilstart Settings

Username	username
Password	••••••••
Shortnumber	72500
IP	82.99.15.65/27 *The IP that calls from Mobilstart originates from*

MOSMS Settings

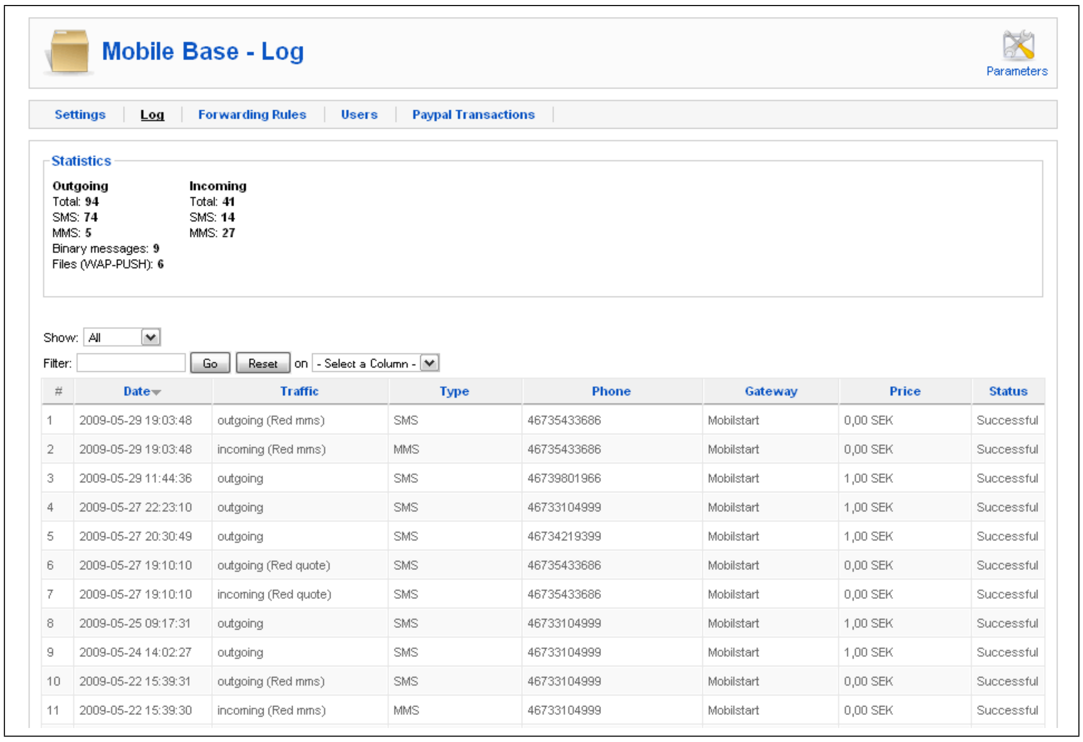
Username	username
Password	••••••••••
IP	93.158.65.162/0 *The IP that calls from MOSMS originates from*

Paypal Settings

API Username	user@mail.com
API Password	••••••••••
API Signature	fgmcfholy8isrtuys56780lk678w45gserutykikg
API Endpoint	https://api-3t.sandbox.paypal.com/nvp
API Version	51.0
Transaction Description	Your balance will be updated upon approval of an adm
Paypal URL	https://www.sandbox.paypal.com/webscr&cmd=_express-checkout&useraction=commit
Return URL	http://redmessage.com/index.php?option=com_cre_mobilebase&cre_cor
Cancel URL	http://redmessage.com/index.php?option=com_cre_mobilebase&cre_cor

Figure 3.3: Mobile Base Back-end - Settings

All incoming and outgoing traffic is logged. In the log section an administrator can view information about which type of message was sent or received, which gateway was used, what price was charged for the message and if it was successfully sent or received, the phone number of the person using the service is also displayed.



Statistics

Outgoing
Total: **94**
SMS: **74**
MMS: **5**
Binary messages: **9**
Files (WAP-PUSH): **6**

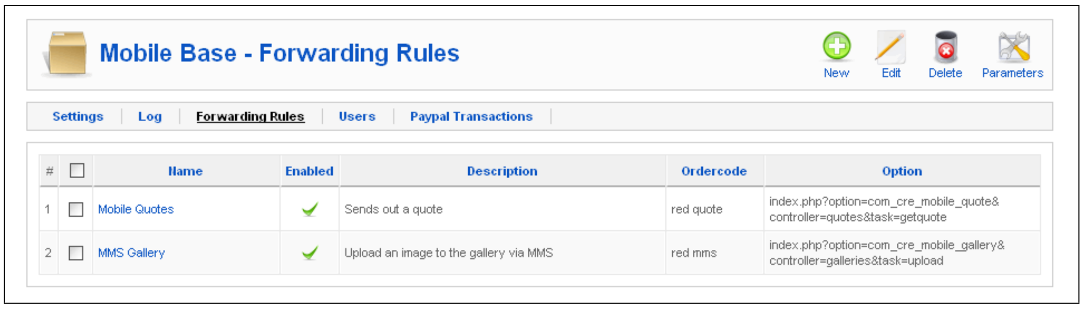
Incoming
Total: **41**
SMS: **14**
MMS: **27**

Show: All
Filter: [Go] [Reset] on - Select a Column -

#	Date	Traffic	Type	Phone	Gateway	Price	Status
1	2009-05-29 19:03:48	outgoing (Red mms)	SMS	46735433686	Mobilstart	0,00 SEK	Successful
2	2009-05-29 19:03:48	incoming (Red mms)	MMS	46735433686	Mobilstart	0,00 SEK	Successful
3	2009-05-29 11:44:36	outgoing	SMS	46739801966	Mobilstart	1,00 SEK	Successful
4	2009-05-27 22:23:10	outgoing	SMS	46733104999	Mobilstart	1,00 SEK	Successful
5	2009-05-27 20:30:49	outgoing	SMS	46734219399	Mobilstart	1,00 SEK	Successful
6	2009-05-27 19:10:10	outgoing (Red quote)	SMS	46735433686	Mobilstart	0,00 SEK	Successful
7	2009-05-27 19:10:10	incoming (Red quote)	SMS	46735433686	Mobilstart	0,00 SEK	Successful
8	2009-05-25 09:17:31	outgoing	SMS	46733104999	Mobilstart	1,00 SEK	Successful
9	2009-05-24 14:02:27	outgoing	SMS	46733104999	Mobilstart	1,00 SEK	Successful
10	2009-05-22 15:39:31	outgoing (Red mms)	SMS	46733104999	Mobilstart	0,00 SEK	Successful
11	2009-05-22 15:39:30	incoming (Red mms)	MMS	46733104999	Mobilstart	0,00 SEK	Successful

Figure 3.4: Mobile Base Back-end - Log

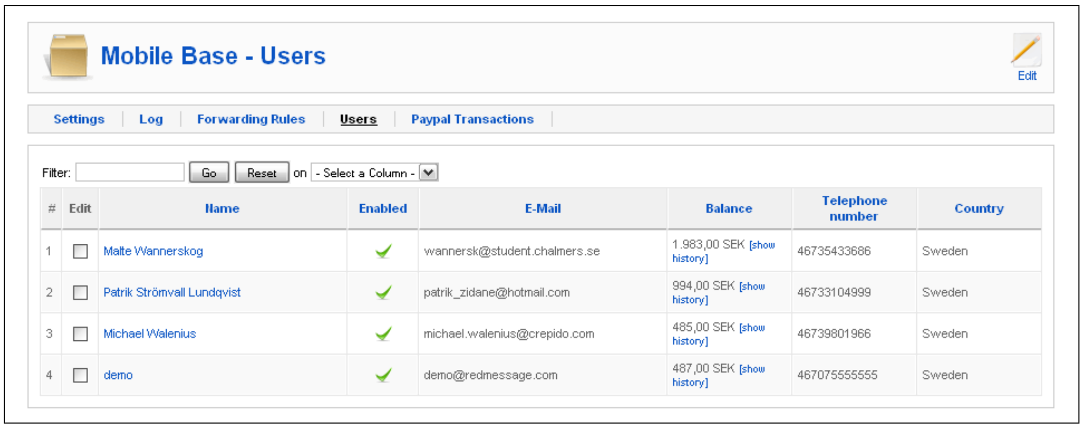
The forwarding rules interface contains a list of rules that defines where incoming messages should be forwarded to. Here an administrator can add, edit and delete forwarding rules.



#	☐	Name	Enabled	Description	Ordercode	Option
1	☐	Mobile Quotes	✓	Sends out a quote	red quote	index.php?option=com_cre_mobile_quote&controller=quotes&task=getquote
2	☐	MMS Gallery	✓	Upload an image to the gallery via MMS	red mms	index.php?option=com_cre_mobile_gallery&controller=galleries&task=upload

Figure 3.5: Mobile Base Back-end - Forwarding Rules

The users interface contains a list of the users registered on the Mobile Base component, the information shown is the name of the user, if he is enabled, e-mail, balance, telephone number and country.



#	Edit	Name	Enabled	E-Mail	Balance	Telephone number	Country
1	☐	Malte Wannerskog	✓	wannersk@student.chalmers.se	1.983,00 SEK [show history]	46735433686	Sweden
2	☐	Patrik Strömvall Lundqvist	✓	patrik_zidane@hotmail.com	994,00 SEK [show history]	46733104999	Sweden
3	☐	Michael Walenius	✓	michael.walenius@crepido.com	485,00 SEK [show history]	46739801966	Sweden
4	☐	demo	✓	demo@redmessage.com	487,00 SEK [show history]	467075555555	Sweden

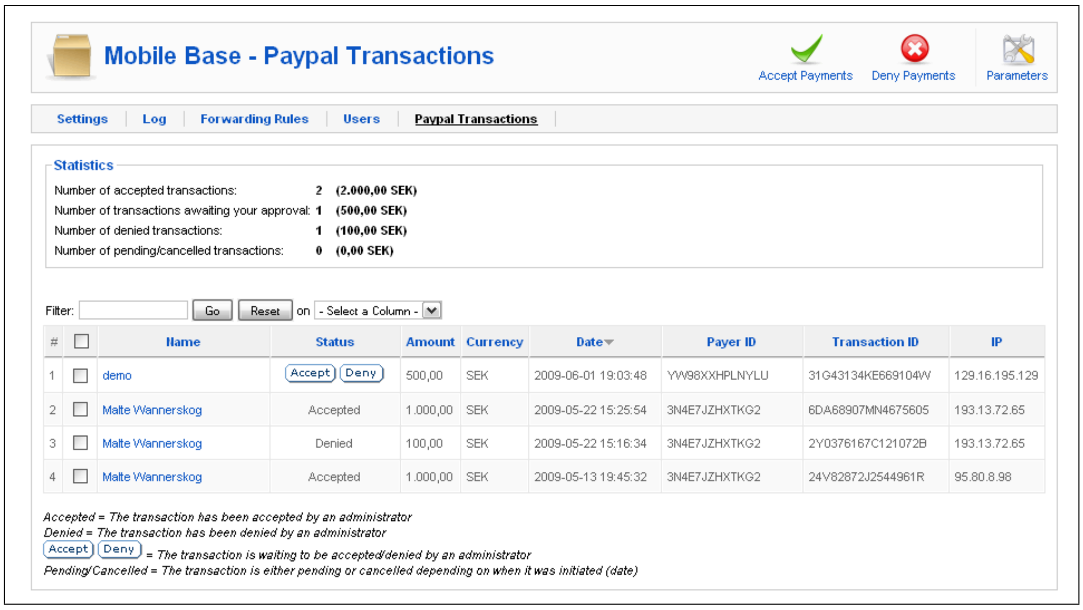
Figure 3.6: Mobile Base Back-end - Users

The PayPal transactions interface contains a list of transactions done by the users of the system. The list contains all the necessary information about a transaction; including username, status, amount, currency, date of transfer, PayPal Payer ID, PayPal Transaction ID and IP of the payer.

When a user initiates a transaction at the front-end the status of the transaction will be *Pending/Cancelled*. Since the lifetime of a PayPal transaction is limited the transaction is *Pending/Cancelled* depending on how much time has passed from the initiation. If the transaction is completed within the lifetime of the transaction and the money has been transferred to the PayPal account the status will be updated to *Completed*.

If the status of the transaction is *Completed* an administrator can either accept or deny the transaction, if accepted the user account balance will be updated and the status changed to *Accepted*, if denied the user account balance will remain the same and the status changed to *Denied*. The accept/deny functionality is an extra security feature that prevents any potential tampered PayPal transactions to be accepted by the system automatically since a transaction must be confirmed by an administrator before money is added to the account of the user.

The interface also shows statistics of the total amount of *Accepted*, *Denied*, *Completed* and *Pending/Cancelled* transactions as well as the amount of money in each status-category.



Statistics

Number of accepted transactions:	2	(2.000,00 SEK)
Number of transactions awaiting your approval:	1	(500,00 SEK)
Number of denied transactions:	1	(100,00 SEK)
Number of pending/cancelled transactions:	0	(0,00 SEK)

Filter: Go Reset on - Select a Column -

#		Name	Status	Amount	Currency	Date	Payer ID	Transaction ID	IP
1		demo	Accept Deny	500,00	SEK	2009-06-01 19:03:48	YW98XXHPLNYLU	31G43134KE669104W	129.16.195.129
2		Malte Wannerskog	Accepted	1.000,00	SEK	2009-05-22 15:25:54	3N4E7JZHXTKG2	6DA68907MN4675605	193.13.72.65
3		Malte Wannerskog	Denied	100,00	SEK	2009-05-22 15:16:34	3N4E7JZHXTKG2	2Y0376167C121072B	193.13.72.65
4		Malte Wannerskog	Accepted	1.000,00	SEK	2009-05-13 19:45:32	3N4E7JZHXTKG2	24V82872J2544961R	95.80.8.98

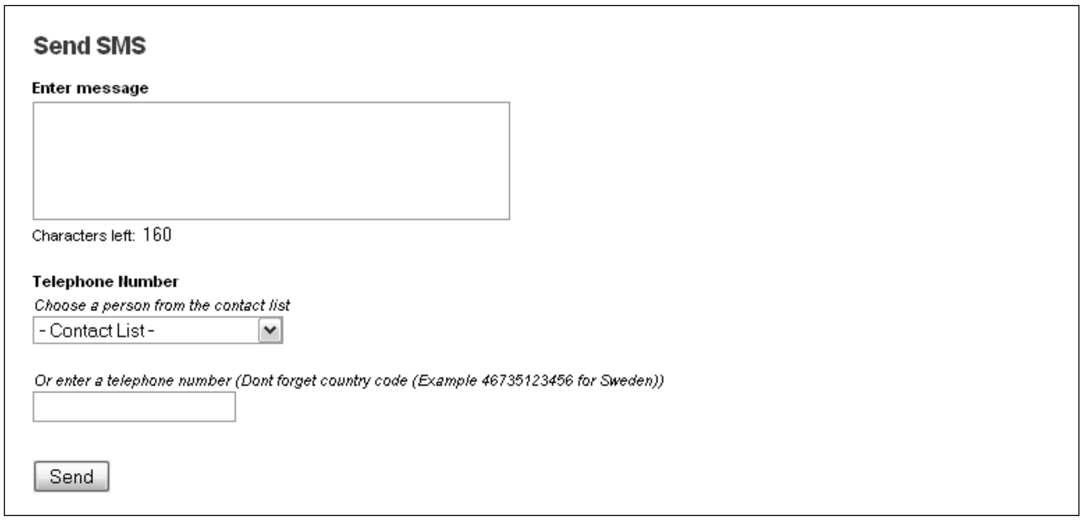
Accepted = The transaction has been accepted by an administrator
Denied = The transaction has been denied by an administrator
Accept Deny *= The transaction is waiting to be accepted/denied by an administrator*
Pending/Cancelled = The transaction is either pending or cancelled depending on when it was initiated (date)

Figure 3.7: Mobile Base Back-end - Paypal Transactions

3.1.2 Send SMS

The component *Send SMS* allows the user to send an SMS from the system.

Front-End
The front-end provides a simple interface for a user to send an SMS. The user writes the message in the message field and may then choose to either enter the receivers' phone number manually or by selecting a person from the contact list. When the user presses send an attempt to send the message as an SMS to the receiver is made and an appropriate message is displayed to the user showing whether the SMS was successfully sent or not.


Send SMS

Enter message

Characters left: 160

Telephone Number

Choose a person from the contact list

- Contact List -

Or enter a telephone number (Dont forget country code (Example 46735123456 for Sweden))

Send

Figure 3.8: Send SMS - Front-end

3.1.3 Send MMS

The component *Send MMS* allows the user to send an MMS from the system.

Front-End
The front-end provides a simple interface for a user to send an image as an MMS. The user browses for an image, enters a subject, a message and may then choose to either enter the receivers' phone number manually or by selecting a person from the contact list. When the user presses send an attempt to send the MMS to the receiver is made and an appropriate message is displayed to the user showing whether the MMS was successfully sent or not.

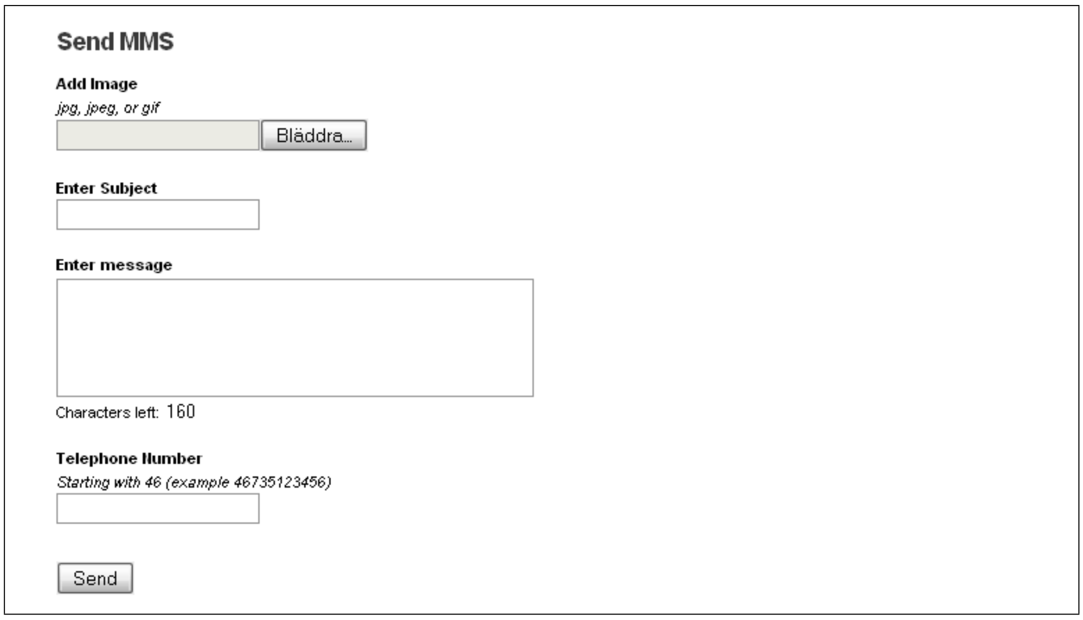


Figure 3.9: Send MMS - Front-end

3.1.4 Intra SMS

The component *Intra SMS* allows the user to send an SMS from the system to one or several recipients.

Front-End
The front-end provides an interface for a user to send an SMS to one or several recipients. In order to use the component a user must register his phone number. Upon registration the user is added to the default group. The phone number can be edited at a later stage by the user. The interface is divided into three sections; *groups*, *available recipients* and *recipients*. The user first selects the group or groups in which his recipients belongs to, the users of this group or groups are then displayed in the *available recipients* section. The user then selects which ones of these that should be moved to the *recipients* section. The user is also able to move back recipients at any time.

When the recipients' selection is done, the user writes the message and presses send; an attempt to send the SMSes to the receivers is made and an appropriate message is displayed to the users showing whether the SMSes were successfully sent or not.

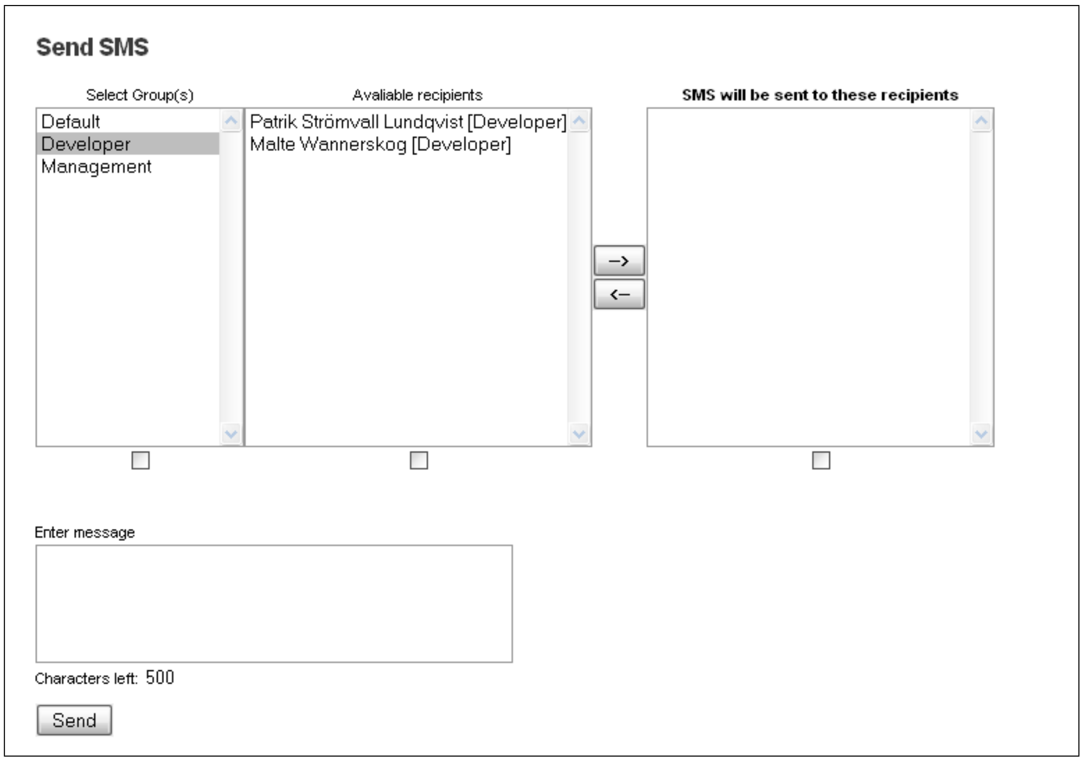
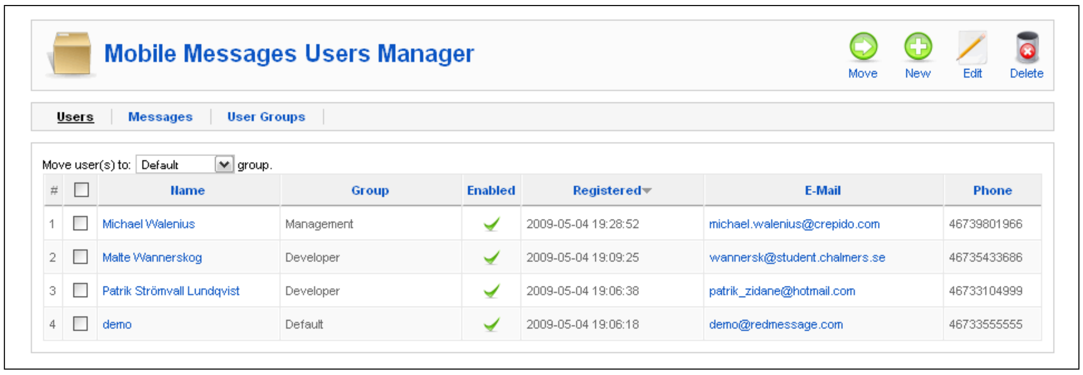


Figure 3.10: Intra SMS - Front-end

Back-End
The back-end provides three interfaces; *Users*, *User Groups* and *Messages*. The user interface contains a list of registered users along with their personal information. Here an administrator can move users between groups, add, edit and remove users.

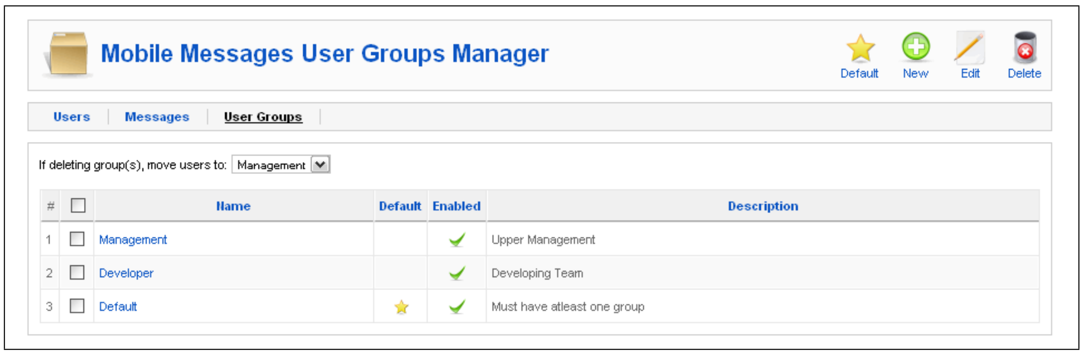


Move user(s) to: Default group.

#	☐	Name	Group	Enabled	Registered	E-Mail	Phone
1	☐	Michael Walenius	Management	✓	2009-05-04 19:28:52	michael.walenius@crepido.com	46739801966
2	☐	Malte Wannerskog	Developer	✓	2009-05-04 19:09:25	wannersk@student.chalmers.se	46735433686
3	☐	Patrik Strömvall Lundqvist	Developer	✓	2009-05-04 19:06:38	patrik_zidane@hotmail.com	46733104999
4	☐	demo	Default	✓	2009-05-04 19:06:18	demo@redmessage.com	46733555555

Figure 3.11: Intra SMS Back-end - Users

The user groups' interface contains a list of groups that registered users can belong to. Here an administrator can add, edit and delete groups and also set the default group in which newly registered users are added to.

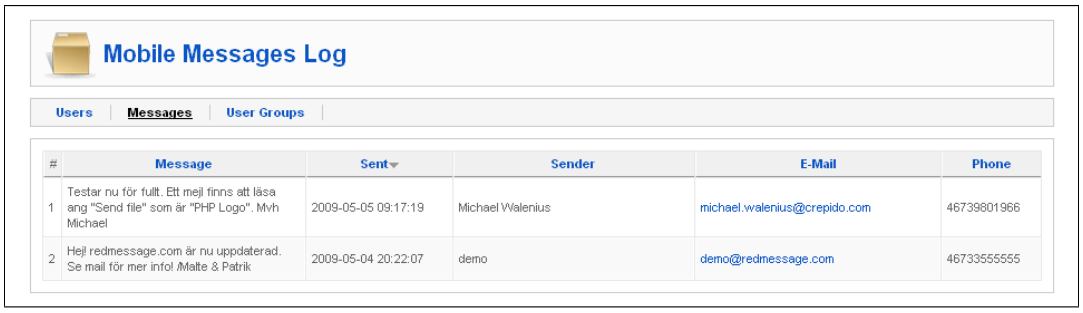


If deleting group(s), move users to: Management

#	☐	Name	Default	Enabled	Description
1	☐	Management		✓	Upper Management
2	☐	Developer		✓	Developing Team
3	☐	Default	★	✓	Must have atleast one group

Figure 3.12: Intra SMS Back-end - User Groups

The messages interface is a log of sent messages. The system logs the message, date and information about the sender.



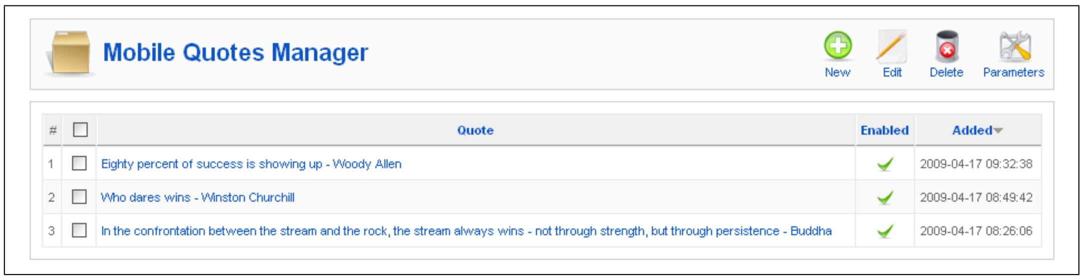
#	Message	Sent	Sender	E-Mail	Phone
1	Testar nu för fullt. Ett mejl finns att läsa ang "Send file" som är "PHP Logo". Mvh Michael	2009-05-05 09:17:19	Michael Walenius	michael.walenius@crepido.com	46739801966
2	Hej! redmessage.com är nu uppdaterad. Se mail för mer info! /Malte & Patrik	2009-05-04 20:22:07	demo	demo@redmessage.com	46733555555

Figure 3.13: Intra SMS Back-end - Messages

3.1.5 Mobile Quote

The component *Mobile Quote* allows the user to send in an SMS to the system and get a famous quote in return.

Back-End
The back-end provides an interface where an administrator can add, edit and delete quotes.



#	☐	Quote	Enabled	Added
1	☐	Eighty percent of success is showing up - Woody Allen	✓	2009-04-17 09:32:38
2	☐	Who dares wins - Winston Churchill	✓	2009-04-17 08:49:42
3	☐	In the confrontation between the stream and the rock, the stream always wins - not through strength, but through persistence - Buddha	✓	2009-04-17 08:26:06

Figure 3.14: Mobile Quote Back-end

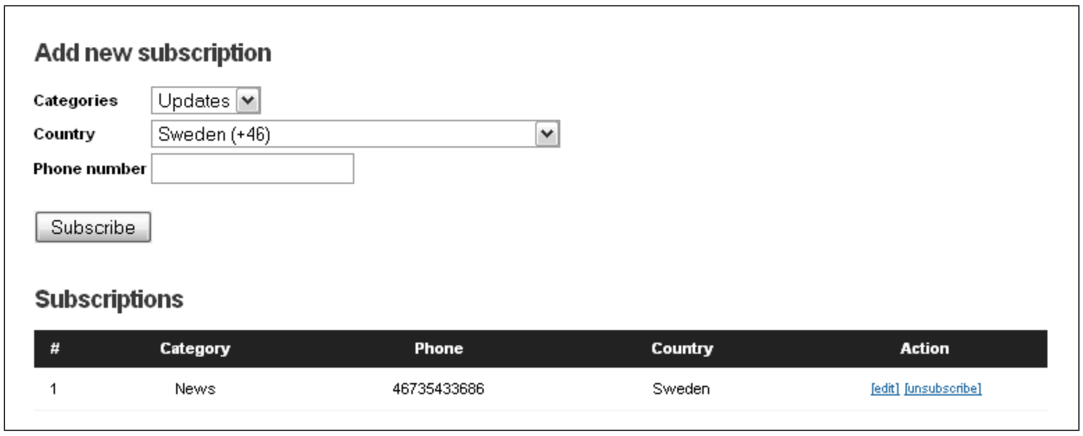
3.1.6 Mobile Subscription

The component *Mobile Subscription* allows a user to subscribe or unsubscribe to various categories which will be sent to a mobile phone.

Front-End
From the-front the user chooses which category that he wants to subscribe

to along with the telephone number of the mobile phone that he wants to receive the information.

The different categories that a user subscribes to are displayed and the user has the possibility to change his telephone number for a specific topic, he may also chose to unsubscribe to a specific topic.



Add new subscription

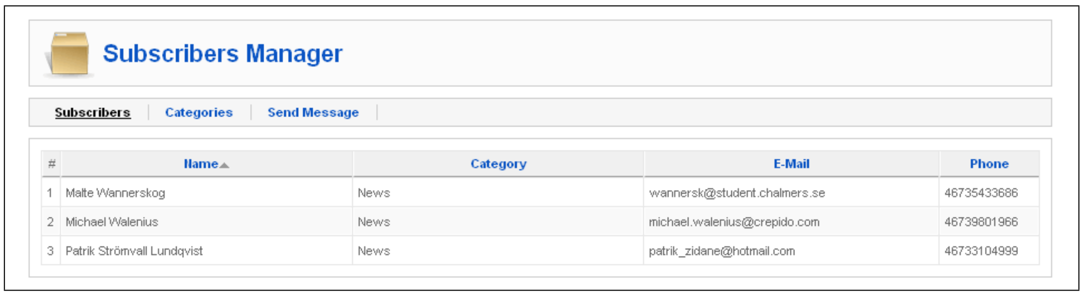
Categories	Updates
Country	Sweden (+46)
Phone number	

Subscriptions

#	Category	Phone	Country	Action
1	News	46735433686	Sweden	[edit] [unsubscribe]

Figure 3.15: Mobile Subscription Front-end

Back End
In the back-end an administrator can view all subscribers along with information about which categories they subscribe to and to which telephone number they will receive the message.

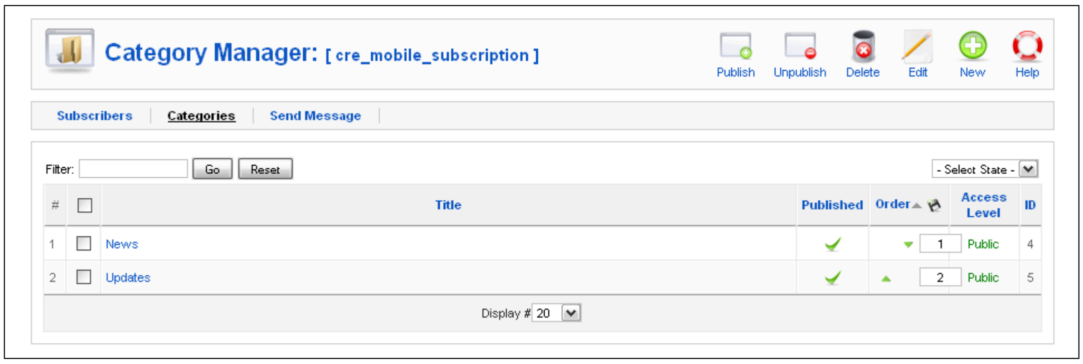


Subscribers Manager

#	Name	Category	E-Mail	Phone
1	Malte Wannerskog	News	wannersk@student.chalmers.se	46735433686
2	Michael Walenius	News	michael.walenius@crepido.com	46739801966
3	Patrik Strömvall Lundqvist	News	patrik_zidane@hotmail.com	46733104999

Figure 3.16: Mobile Subscription Back-end - Subscribers

An administrator can create new categories which users can subscribe to. Existing categories can be created or edited as well as published / unpublished.

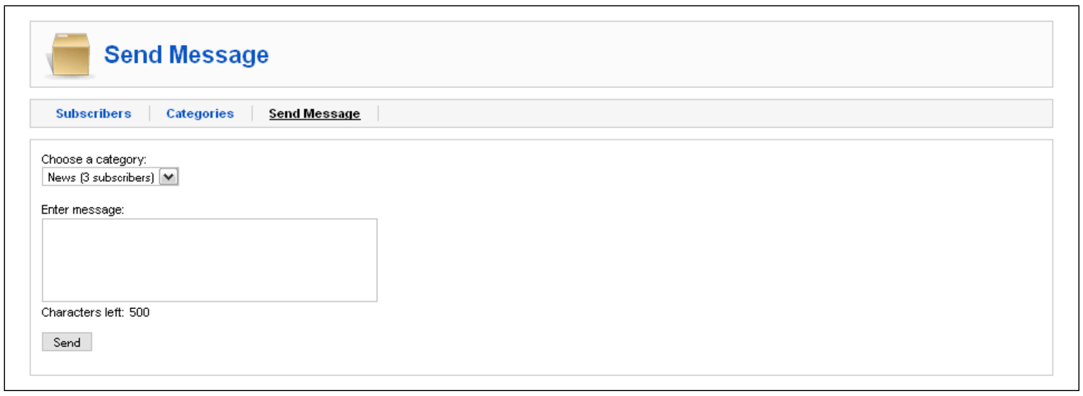


Category Manager: [cre_mobile_subscription]

#		Title	Published	Order	Access Level	ID
1	☐	News	✓	1	Public	4
2	☐	Updates	✓	2	Public	5

Figure 3.17: Mobile Subscription Back-end - Categories

From the back-end an administrator can chose a topic and send a message. An SMS will then be sent to all users who subscribe to that specific topic.



Send Message

Choose a category:
News (3 subscribers)

Enter message:

Characters left: 500

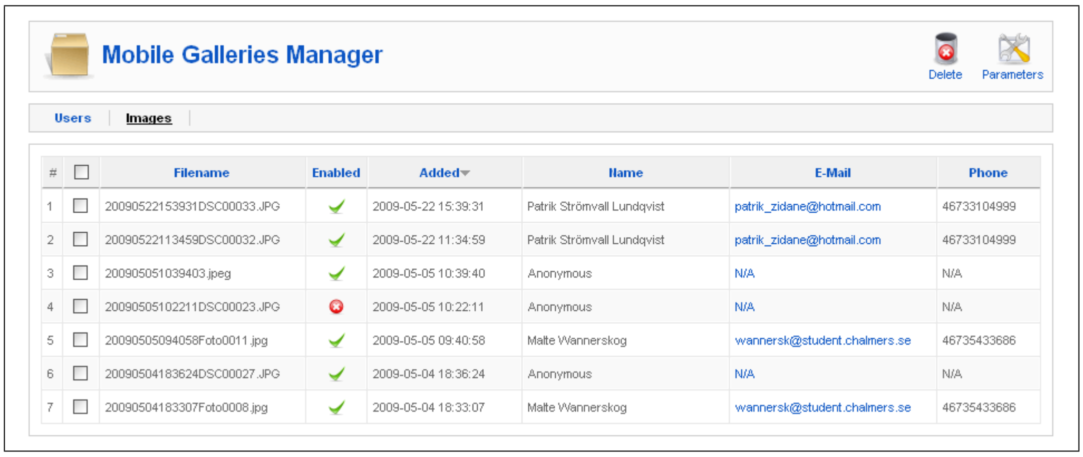
Figure 3.18: Mobile Subscription Back-end - Send Message

3.1.7 Mobile Gallery

The extension *Mobile Gallery* allows a user to send an MMS containing an image to the system which will then be displayed in a module.

Front-End
The front-end is represented by a module displaying images that have been sent in to the system as MMS messages. If the user is registered on the Joomla! site he can choose to register his telephone number, by doing this his name will appear on the images that he sends to the system.

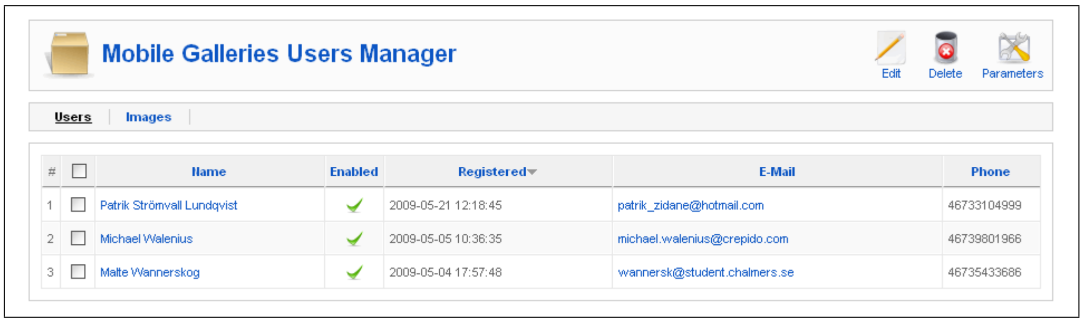
Back-End
In the back-end all uploaded images are shown, if the user who uploaded the image is a registered user, information about that user is also shown. The administrator can enable and disable images; by disabling an image that image won't be showed in the front-end. One or several images can also be deleted; this will completely remove them from the system.



#	☐	Filename	Enabled	Added▾	Name	E-Mail	Phone
1	☐	20090522153931DSC00033.JPG	✓	2009-05-22 15:39:31	Patrik Strömvall Lundqvist	patrik_zidane@hotmail.com	46733104999
2	☐	20090522113459DSC00032.JPG	✓	2009-05-22 11:34:59	Patrik Strömvall Lundqvist	patrik_zidane@hotmail.com	46733104999
3	☐	200905051039403.jpeg	✓	2009-05-05 10:39:40	Anonymous	N/A	N/A
4	☐	20090505102211DSC00023.JPG	✗	2009-05-05 10:22:11	Anonymous	N/A	N/A
5	☐	20090505094058Foto0011.jpg	✓	2009-05-05 09:40:58	Malte Wannerskog	wannersk@student.chalmers.se	46735433686
6	☐	20090504183624DSC00027.JPG	✓	2009-05-04 18:36:24	Anonymous	N/A	N/A
7	☐	20090504183307Foto0008.jpg	✓	2009-05-04 18:33:07	Malte Wannerskog	wannersk@student.chalmers.se	46735433686

Figure 3.19: Mobile Gallery Back-end - Images

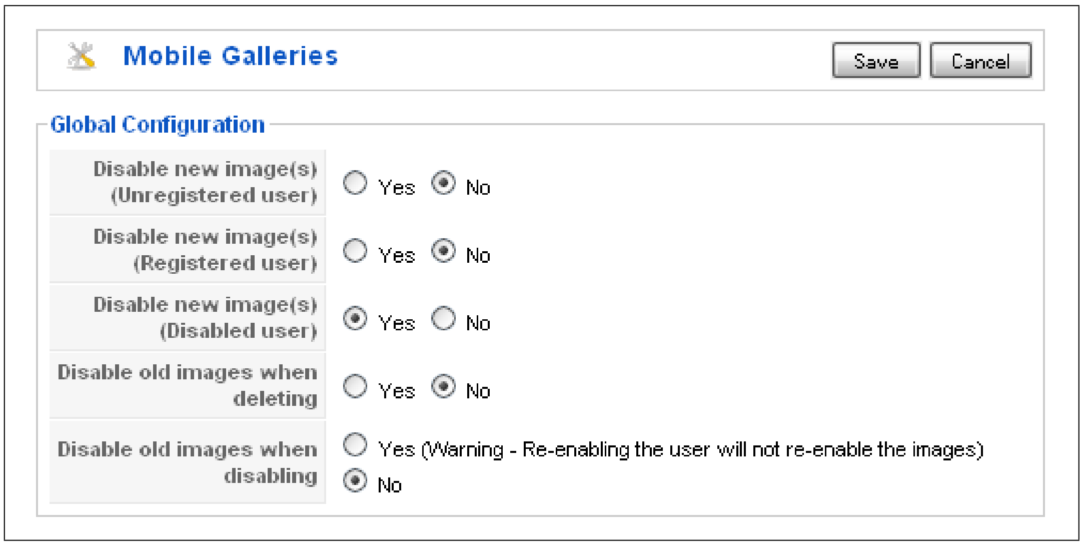
All users who have registered a phone number are shown. An administrator has the ability to enable and disable a user, as well as edit the information about that specific user. He can also delete a user completely.



#	☐	Name	Enabled	Registered▾	E-Mail	Phone
1	☐	Patrik Strömvall Lundqvist	✓	2009-05-21 12:18:45	patrik_zidane@hotmail.com	46733104999
2	☐	Michael Walenius	✓	2009-05-05 10:36:35	michael.walenius@crepido.com	46739801966
3	☐	Malte Wannerskog	✓	2009-05-04 17:57:48	wannersk@student.chalmers.se	46735433686

Figure 3.20: Mobile Gallery Back-end - Users

In the parameters section several options exist to control the behavior of the gallery. By editing the settings here an administrator can chose whether newly added images will be enabled or disabled when they are uploaded. If they are disabled they will not be shown unless an administrator manually enables them, this gives the administrator the ability to review the content before it is published in the system. He may also choose whether all images uploaded by a user should be disabled if that user is deleted or disabled.



Global Configuration

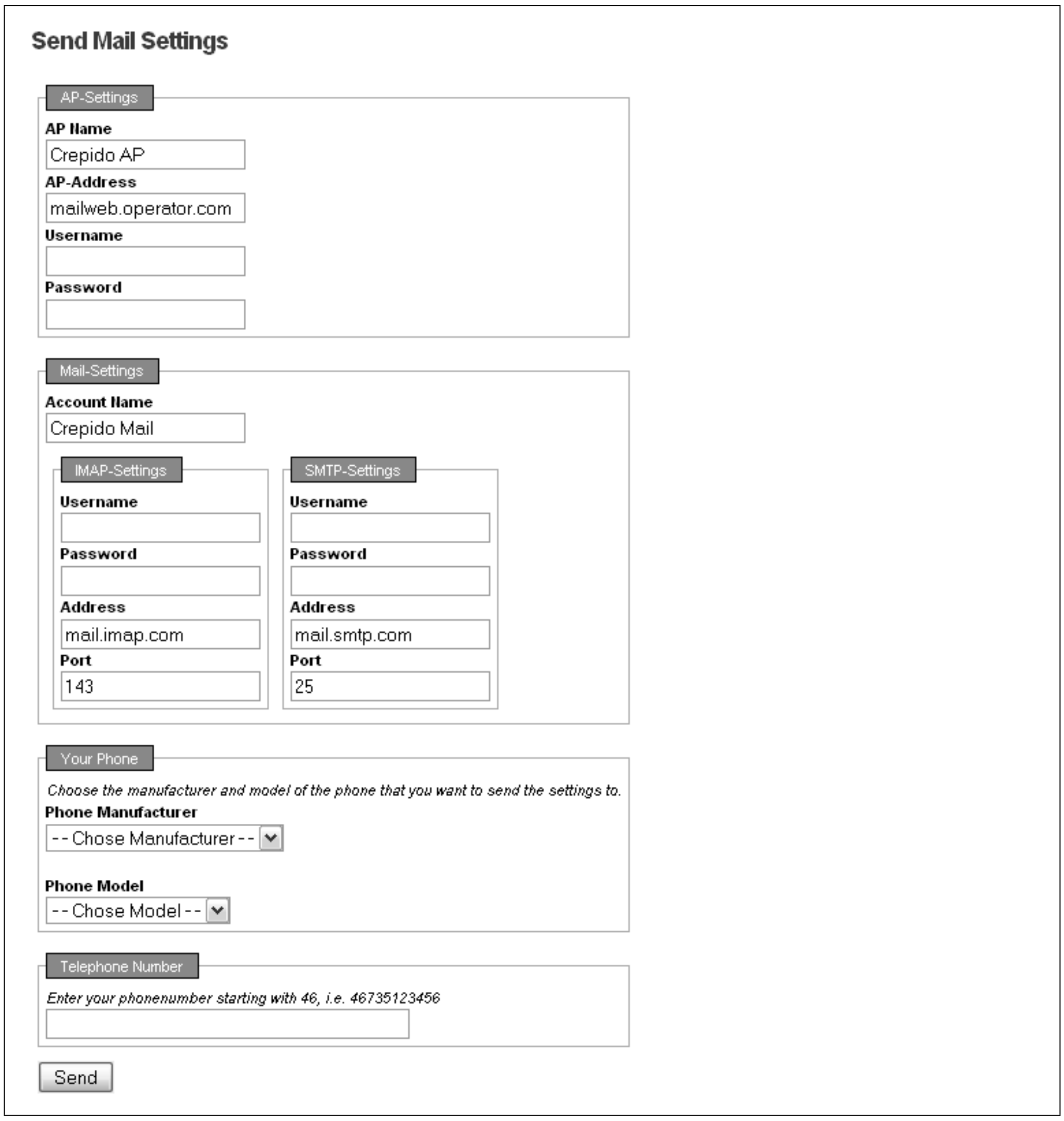
Disable new image(s) (Unregistered user)	Yes [✓ No]
Disable new image(s) (Registered user)	Yes [✓ No]
Disable new image(s) (Disabled user)	[✓ Yes] No
Disable old images when deleting	Yes [✓ No]
Disable old images when disabling	Yes (Warning - Re-enabling the user will not re-enable the images) [✓ No]

Figure 3.21: Mobile Gallery Back-end - Parameters

3.1.8 Send E-mail and Browser Settings

The component allows the user to send IMAP, SMTP, Access Point, Internet and browser configurations from the system to mobile phones Over-The-Air via SMS.

Front-End
The front-end provides two interfaces where the user can enter his desired account information, phone model and phone number. When the user presses send an attempt to send the entered configuration is made and an appropriate message is displayed to the users showing whether the settings were successfully sent or not.


Send Mail Settings

AP-Settings

AP Name
Crepido AP

AP-Address
mailweb.operator.com

Username

Password

Mail-Settings

Account Name
Crepido Mail

IMAP-Settings

Username

Password

Address
mail.imap.com

Port
143

SMTP-Settings

Username

Password

Address
mail.smtp.com

Port
25

Your Phone

Choose the manufacturer and model of the phone that you want to send the settings to.

Phone Manufacturer
-- Chose Manufacturer --

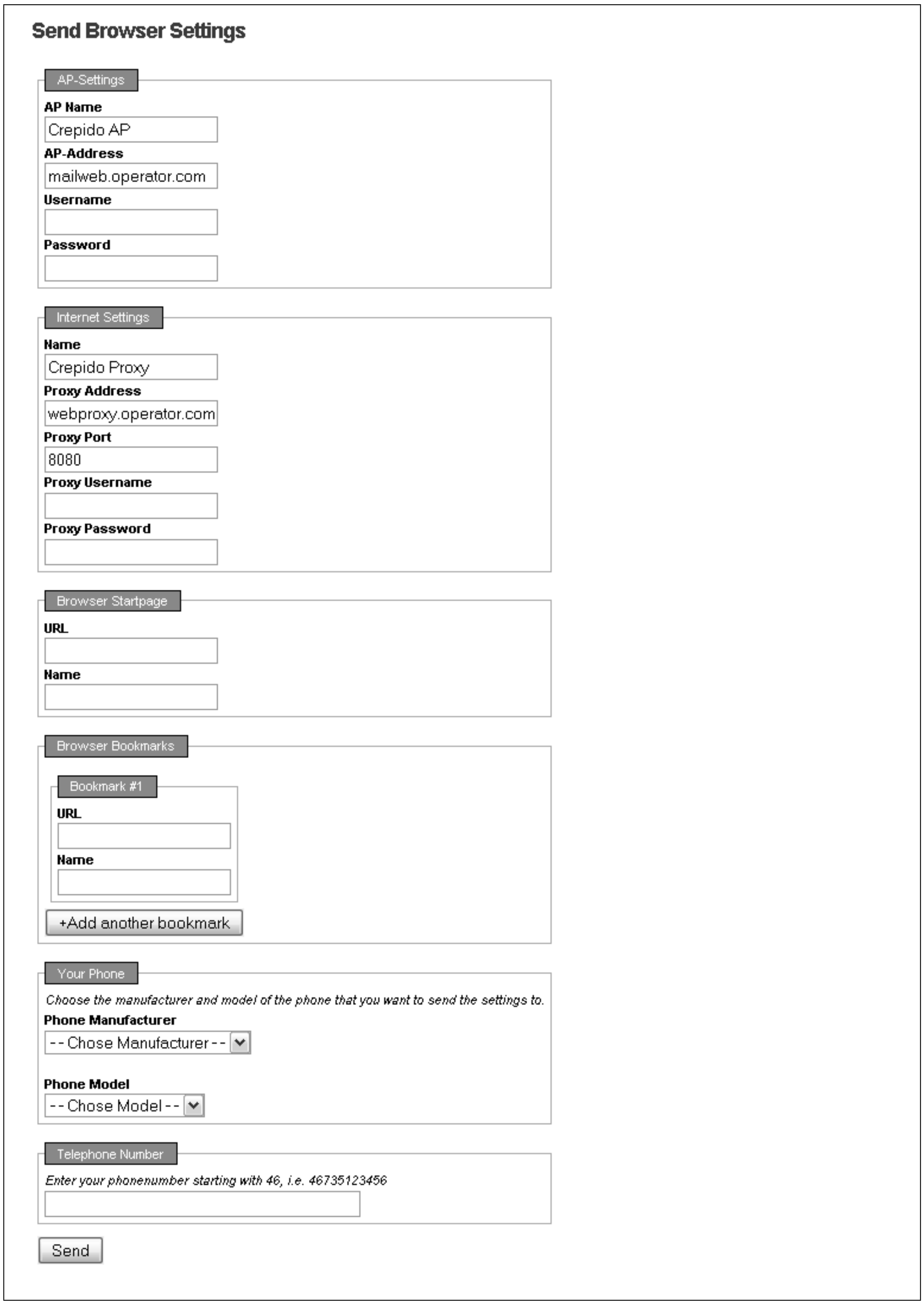
Phone Model
-- Chose Model --

Telephone Number

Enter your phonenumber starting with 46, i.e. 46735123456

Send

Figure 3.22: Send E-mail Settings Front-end.



Send Browser Settings

AP-Settings

AP Name: Crepido AP

AP-Address: mailweb.operator.com

Username:

Password:

Internet Settings

Name: Crepido Proxy

Proxy Address: webproxy.operator.com

Proxy Port: 8080

Proxy Username:

Proxy Password:

Browser Startpage

URL:

Name:

Browser Bookmarks

Bookmark #1

URL:

Name:

+Add another bookmark

Your Phone

Choose the manufacturer and model of the phone that you want to send the settings to.

Phone Manufacturer: -- Chose Manufacturer --

Phone Model: -- Chose Model --

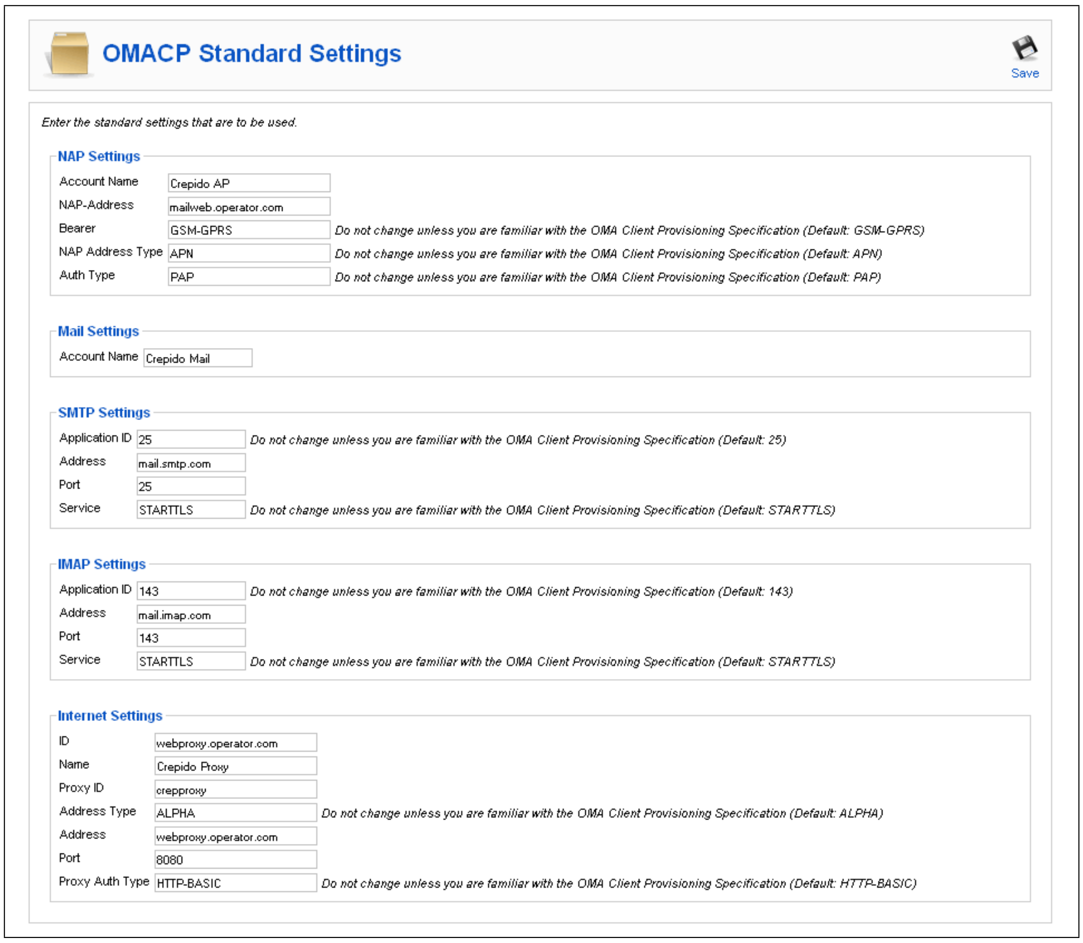
Telephone Number

Enter your phonenumber starting with 46, i.e. 46735123456

Send

Figure 3.23: Send Browser Settings Front-end.

Back-End
In the back-end an administrator can set the default values of the fields displayed in the front-end. He can also set more advanced settings that require knowledge about the OMA Client Provisioning Specification [5]; these settings are not displayed in the front-end.



Enter the standard settings that are to be used.

NAP Settings

Account Name	Crepido AP	
NAP-Address	mailweb.operator.com	
Bearer	GSM-GPRS	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: GSM-GPRS)*
NAP Address Type	APN	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: APN)*
Auth Type	PAP	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: PAP)*

Mail Settings

Account Name	Crepido Mail

SMTP Settings

Application ID	25	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: 25)*
Address	mail.smtp.com	
Port	25	
Service	STARTTLS	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: STARTTLS)*

IMAP Settings

Application ID	143	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: 143)*
Address	mail.imap.com	
Port	143	
Service	STARTTLS	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: STARTTLS)*

Internet Settings

ID	webproxy.operator.com	
Name	Crepido Proxy	
Proxy ID	crepproxy	
Address Type	ALPHA	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: ALPHA)*
Address	webproxy.operator.com	
Port	8080	
Proxy Auth Type	HTTP-BASIC	*Do not change unless you are familiar with the OMA Client Provisioning Specification (Default: HTTP-BASIC)*

Figure 3.24: Send E-mail Settings Back-end

4 Discussion

During our work we have encountered, for us, new areas in which we had no prior knowledge. For example working with Joomla! was a new experience for both of us. We found it to be relatively simple to both learn how to use it as well as learning how to develop our own extensions. We find that using installable extensions makes Joomla! very flexible. As developers we think that this is very good since the components we have developed can be installed on all systems running Joomla!.

Integrating mobile services into Joomla! wasn't very difficult, however, if developing extensions on regular basis that require mobile services we feel that a tool *Mobile Base* that simplifies this process is needed. A developer that is new to mobile services will save a lot of time by using the library in *Mobile Base*. Another advantage is that if a Joomla! site uses a lot of extensions where all provide mobile services, a lot of redundancy is avoided by gathering this functionality into a library.

In addition to the development of *Mobile Base* we have developed a set of extensions that all provide mobile services through the use of *Mobile Base*. While some of these extensions may seam ludicrous the purpose of them are to demonstrate different services that can be implemented by the use of *Mobile Base*. For example the component *Mobile Quote* demonstrate the behaviour where a user sends an SMS which is processed by the system and receives an SMS as a response. This can be compared to a ticket buying system.

At the moment only two mobile gateways are supported, however the original problem description only stated the support for Mobilstart. The reason why MO-SMS was added was to see how easily support for several gateways could be implemented. In future implementations not much time would have to be spent if support for more gateways is needed.

5 Conclusion

We feel that our work has successfully resulted in a tool that will aid developers interested in developing extensions for Joomla! that are to provide mobile services. By installing and using *Mobile Base* developers will save a lot of time, however we feel that some improvements can be made. The scenario is most likely that not all of the functionality provided by *Mobile Base* is needed for all web sites. For example, a developer might only be interested in using the functionality to send messages provided by *Mobile Base*. By installing *Mobile Base* he will now also get interfaces for receiving messages and handling payments which for him might be unnecessary. Therefore if we were to continue the development of *Mobile Base* our first step would be to divide it into smaller packages. This would result in that a developer can chose the package or packages that suit his needs.

6 References

[1] Hagen Graf, *Building Websites with Joomla! 1.5*. Packt Publishing, Birmingham, United Kingdom, 2008. ISBN 978-1-847195-30-2.

[2] Joseph LeBlanc, *Learning Jooma! 1.5 Extension Development Ű Creating Modules, Components, and Plug-Ins with PHP*. Packt Publishing, Birmingham, United Kingdom, 2007. ISBN 978-1-847191-30-4.

[3] Open Mobile Alliance, Homepage. [Fetched: May 13, 2009]. http://www.openmobilealliance.org/

[4] LookupTables, ASCII Table. [Fetched: May 26, 2009]. http://www.asciitable.com/

[5] Mobilstart, Telenor Mobilstart [Fetched: March 4, 2009]. http://www.mobilstart.se

[6] Joomla!, What is Joomla? [Fetched: March 7, 2009]. http://www.joomla.org/about-joomla.html

[7] Joomla!, Developing a Model-View-Controller Component [Fetched: Juni 18, 2009]. http://docs.joomla.org/Developing_a_Model-View-Controller_Component_-_Part_1

[8] PHP, PHP: cURL - Manual [Fetched: Juni 17, 2009]. http://se.php.net/curl

[9] Open Sync, libwbxml [Fetched: Juni 26, 2009]. https://libwbxml.opensync.org/

[10] Sony Ericsson Mobile Communications AB, *OMA Client Provisioning and Device Management for Sony Ericsson phones with DM client v.1 Ű 3*. January 2008. Publication Number: EN/LZT 108 8469 R14A.

[11] PayPal, *Website Payments Standard Integration Guide*. April 2009. Document Number: 100000.en_US-200904.

[12] PayPal, *Instant Payment Notification Guide*. March 2008. Document Number: 10087.en_US-200903.

[13] PayPal, *Express Checkout Integration Guide*. April 2008. Document Number: 100010.en_US-200904.

[14] Open Mobile Alliance, *Provisioning Content.* 26 February 2008. Document Number: OMA-WAP-TS-ProvCont-v1_1-20080226-C.

[15] Open Mobile Alliance, *Binary XML Content Format Specification.* 25 July 2001. Document Number: WAP-192-WBXML-20010725-a.

[16] Open Mobile Alliance, *Provisioning Bootstrap.* 21 Apr 2009. Document Number: OMA-WAP-TS-ProvBoot-V1_1-20090421-C.

[17] Bozoka.com AB, *Bozoka SMS Send API 2.1.* 13 February 2009.

[18] Bozoka.com AB, *Bozoka MMS Send API 1.1.* 13 February 2009.

[19] Bozoka.com AB, *Bozoka mobilForward manual.* 8 June 2009.

7 Glossary

CMS - Content Management System (A system for managing the content of an application).
WCMS - Web Content Management System (A system for managing the content of a web application).
Joomla! - Open-Source WCMS.
PHP - PHP Hypertext Preprocessor (A web server based script language used to develop dynamic websites).
MySQL - My Structured Query Language (A relational database management system).
SMS - Short Message Service (A communication service for exchanging short messages via mobile communication systems).
MMS - Multimedia Messaging Service (A communication service for exchanging multimedia objects (images, audio, video, text) via mobile communication systems).
IMAP - Internet Message Access Protocol (An internet standard protocol for e-mail retrieval).
SMTP - Simple Mail Transfer Protocol (An internet standard protocol for e-mail transmission).
OMA - Open Mobile Alliance (A standards organization which develops open standards for the mobile phone industry).
CP - Client Provisioning.
OTA - Over-The-Air.
PayPal - PayPal is an online business that allows users to transfer money

and do payments over the internet.
IPN - Instant Payment Notification (PayPal's interface for handling real-time purchase confirmation and server-to-server communications).
DTD - Document Type Definition (A DTD outlines the rules of a particular document structure. It lists the elements, attributes, and entities in a document and it defines the relationships between the different elements and attributes).
XML - Extensible Markup Language (A general-purpose specification for creating custom markup languages).
WBXML - WAP Binary XML (A binary representation of XML).