Dimension Technology Solutions Team 2

eMESA Web Service Extension and iPhone Interface

"…6 weeks, 3 phases, 2 products, 1 client, design, implement…"

- **Presentation Date: Thursday June 18**
-

Authors: Mark Barkmeier and Matthew Mock

Client: Dimension Technology Solutions (DTS) - Daniel Lynn

Colorado School of Mines

Field Session 2009

Contents

I. Abstract

DTS specializes in software for maintenance professionals with their eMESA Live application. They would like to add capabilities to interface features of the eMESA website with the iPhone. In essence, the services they would like to provide would allow managers, such as those from a mining company, to schedule work orders on the eMESA Live website and have the details of the order sent to a worker, live, directly accessible via an iPhone Application.

The scope of work accomplished for the summer field session is the development of the framework for the iPhone application. The focus will be on a web administrator interface, the authentication to the eMESA server by the iPhone, and basic data exchange via web services. A large feature set for the iPhone application itself was not planned for the scope of the summer work.

To fulfill the scope of the project, three development phases were accomplished with two viewable products in the end. In phase one changes were made to the eMESA framework to allow management of eMESA Mobile users. Phase two dealt with the creation of the web service which handles all communication between the eMESA framework and the iPhone application. Phase three is the groundwork for the iPhone application which authenticates with the web service and displays some basic data.

An iPhone users management screen was implemented into the eMESA Live website to be accessible on the internet by company managers, allowing them to add and remove login permissions for employees using iPhones.

The groundwork for the eMESA iPhone Application was implemented with authentication methods and some basic data reporting. For the iPhone to communicate with eMESA, a web service was installed on the eMESA framework to handle information transfer and authentication. When an iPhone user account is created via the web interface (for an employee to access eMESA on the iPhone) the user will be able to authenticate into the eMESA system via their iPhone or other mobile device.

The iPhone application communicates through the web service with eMESA and stores the necessary information to authenticate with eMESA for future connections. The iPhone application itself forces authentication with the web service as the first step before a user can access the other features.

II. Introduction

Dimension Technology Solutions (DTS) writes software that is designed for maintenance professionals to be used through a web interface. Their primary product, eMESA, is used to

track equipment, and create work orders or requisitions for them. eMESA also does the scheduling of the maintenance personnel for professional maintenance companies. Currently the eMESA software is accessed through a web browser, so users are not able to easily use it on a mobile device in the field. DTS wants an iPhone application for that allows portable access to the eMESA system and thus, usable by workers in the field. The project assigned was to build a platform that could be used to develop the iPhone mobile versions of eMESA.

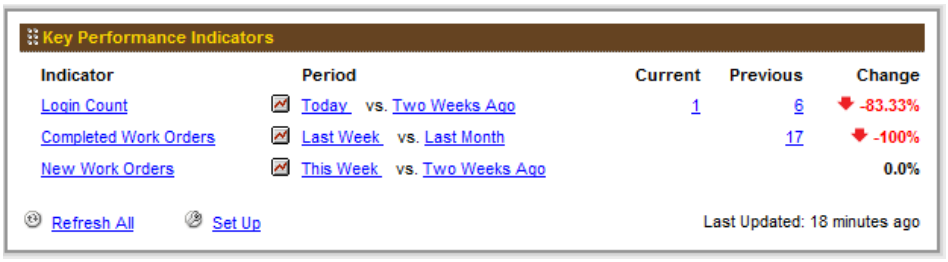
III. Requirements

A. Functional Requirements

Administrators in the eMESA application need the capability to manage which users are able to connect to eMESA via the iPhone. The capabilities of the management interface include searching for a user by name, adding iPhone access permissions to a user, removing a user's iPhone access permissions, and reporting basic information about a user (dates, access information, etc).

The eMESA iPhone application must have the capability of authenticating to an instance of eMESA. iPhone eMESA users are able to do a few things in addition to authenticating to an instance of eMESA. Users of the iPhone version of eMESA are able to view their profile information. They can also view and refresh their main Key Performance Indicators (KPIs)[1].

KPIs are statistics that are used to compare company performance between different time frames. They are recalculated regularly by the eMESA system and at request by the user. They are readily available for comparisons between several date ranges (including today, yesterday, this week, last year, etc). Some examples of the data compared with KPIs include work orders created and work orders closed. KPIs, as illustrated in figure 1 below, are determined and supplied in the system on a per-user basis.



(**Figure 1**: Key Performance Indicators in eMESA, for a specific user)

B. Non Functional Requirements

Some requirements for the programmers are the use and integration of the existing eMESA framework, the use of Objective-C (Cocoa), the iPhone SDK, Mac OS X (the operating system required to produce the iPhone application), and use of a web service developed in Visual Studio with C#. The eMESA framework must be utilized in order to retrieve and update data. The use of Cocoa and the iPhone SDK are both necessary to create an iPhone application. And finally, the use of Mac OS X is required to use the iPhone SDK and therefore is required to make an iPhone application as well.

C. Scope

The scope of this project was laying the foundation for a full iPhone application for eMESA. The application authenticates with the eMESA servers and reports data to the users from the database. The Agile programming methodology was utilized with three iterations (or cycles) of two weeks in length spanning the 6 week programming session.

D. Use Cases

- Admin assigns activation key to a user via website
- Admin deactivates activation key for an iPhone user
- A user logs into the iPhone application and is able to view their profile information and key performance indicators

- An admin logs into the eMESA web interface and searches for an existing eMESA user by typing their name in a search box. Existing usernames show up matching the search query.
- An admin can access a list of mobile registrations owned by the user's account

- Admin can generate a mobile access key for a user with a corresponding expiration date. This will allow the user to access the eMESA system with their iPhone, using the key for authentication.
- Admin can change the expiration date for a user and set it infinite.
- An iPhone user with a valid mobile key that is expired will not be permitted access to the server
- A user mobile eMESA user with valid mobile access key can log into the iPhone application, be authenticated by the web service, and then gain access into the applications features.

- An eMESA user that has been authenticated can view a page in the iPhone application containing detailed information from the user's profile, such as contact information.

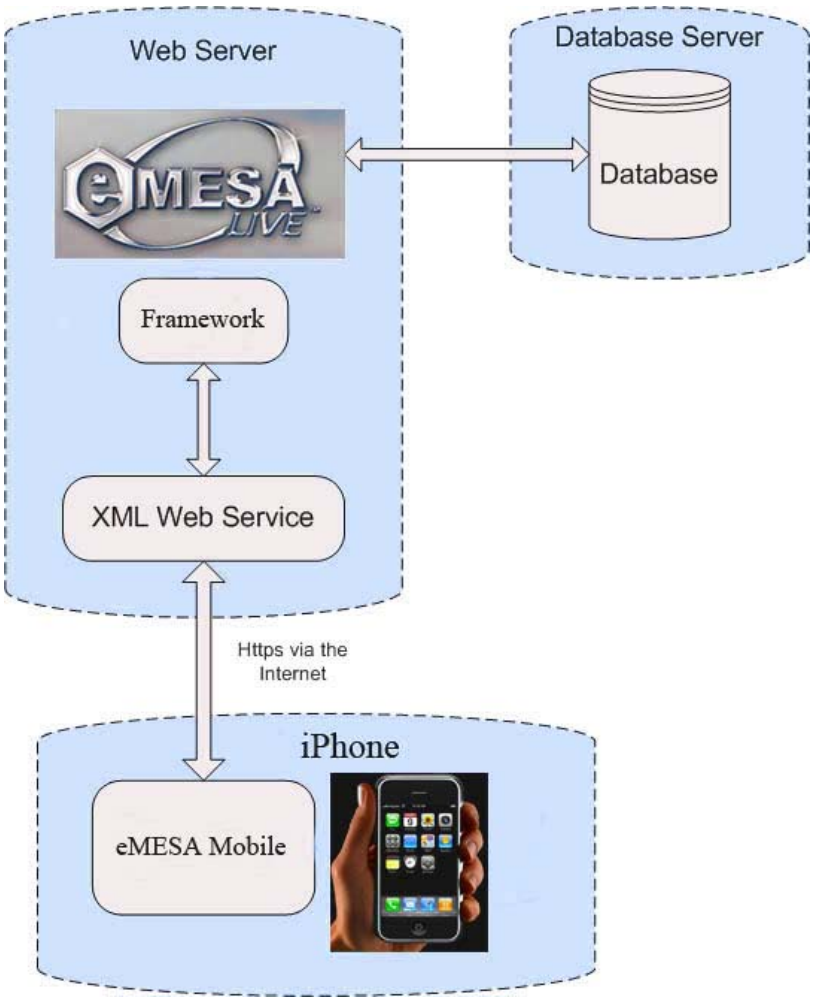
- Authenticated users can view a chart with statistics containing the user's key performance indicator data on the iPhone.

IV. Design

In the current eMESA system, administrators assign permissions to users to enable access to different functionality. Additional permissions were added to the system to enable user's access to eMESA via the iPhone. This extension required new "admin" screens as well as additions to the database to identify and provide authentication data for users permitted to access eMESA via iPhone.

Once users have been authorized to access eMESA via iPhone, they have the ability to view their profile and their KPIs. To provide this capability, a new web service was added to eMESA that interacts between the iPhone and the database using the eMESA framework.

The framework was designed to abstract the database queries out, making the code seem as if there is no database present. This is handled using a framework called NHibernate. The flow of information from the database through the web service and ultimately to the iPhone is illustrated in figure 2.



(**Figure 2:** eMESA Mobile Communication Framework Diagram)

A. High Level Design

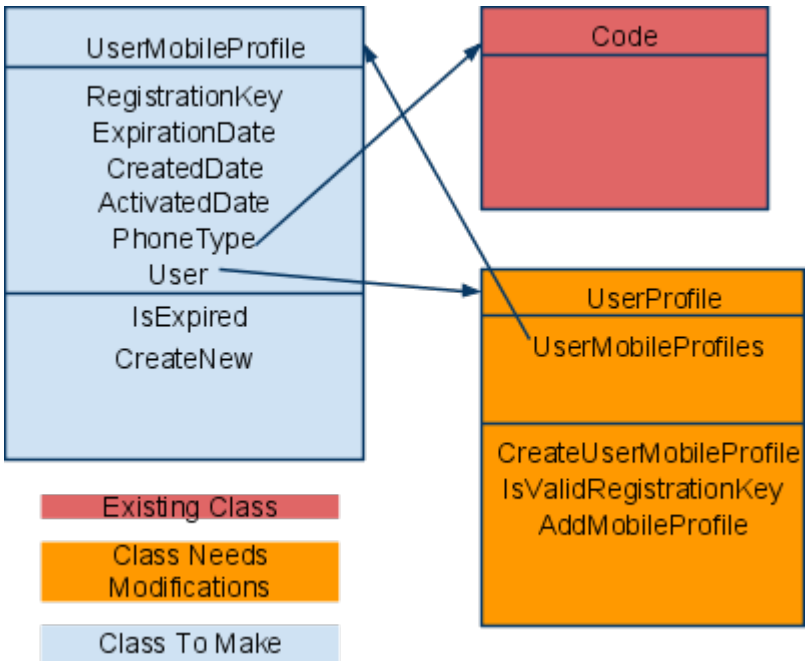
The existing eMESA application is meant to be used solely as a web based program and is therefore not mobile friendly. The pages are too large and have a reliance on JavaScript that doesn't work well on mobile web browsers. The current system was extended to create administration of mobile users and a separate web service was written as the basis for a mobile platform for eMESA. This web service communicates with the eMESA system and serves as a basis for a full blown iPhone application that will interface with eMESA.

B. Detail Design

i. Modifications to existing system (Phase 1)

Modifications to the existing system were required in a few classes. The first set of changes necessary were in the eMESA framework. The eMESA Framework consists of all of the classes that represent the different data types and structures in eMESA. The existing UserProfile class had the data type "UserMobileProfile" added to it. The Code class is an existing class which is used to store a single generic piece of data. In most contexts, it is storing a type. In this specific situation, it contains the type of the phone (the framework was designed so any mobile phones can have an application written for them to also connect).

The UserMobileProfile class contains all the data for a specific mobile registration. Below is the UML for the modifications to the eMESA system (the full UML is too large to easily fit in a single document).



(**Figure 3**: UML of modifications to eMESA Framework)

In addition to the new UserMobileProfile class that was added to the UserProfile class, methods were added to UserProfile for verifying security and adding mobile profiles. The IsExpired

function in UserMobileProfile returns true if the registration has expired, and CreateNew is a static function which creates a new UserMobileProfile with fields initialized properly (constructors initializing members cause problems with database mappings).

The eMESA system adheres to the Model View Controller (MVC) pattern for processing and displaying information. A new view was added to display the Mobile User's information. An existing controller, AdminController, was modified to incorporate the mobile profiles into the framework. Changes were made to the AdminController which designates the views and functionality available to an administrator.

As part of the User Moble Administration view, JavaScript scripts were created to handle the data submitted from managers. These functions ensure that the data submitted by the user is formatted correctly before submission and then makes Ajax requests to the Admin Controller, which accesses the specific functions built for manipulating the data in the UserMobileProfile. Such functions include creating a new account, updating the expiration date for an account, and deactivating an account.

ii. New systems to build

a. Phase 2 (web service)

The web service for communicating with eMESA uses the framework already created for all of the objects in eMESA. This includes the automatic handling of database communications. All of the programming and functionality of eMESA interacts directly with the objects and classes in the framework; the service itself does the interaction with the database. The web service was phase 2 because it is the layer between eMESA and the iPhone application, and Phase 3 (the iPhone Application stage) requires the web service to function.

As mentioned above, the features of the web service will include KPI's that the eMESA system has associated with the user's account. Basic user information that is reported include a user's full name, contact information (email and phones), Title, Employee Number, Work Group, Primary Role[2], and Site[3].

Communication between the web service and the iPhone is done using XML[4]. The web service sends data to the iPhone in an XML encoded document and also receives information from the iPhone in an XML encoded document. The iPhone application extracts the necessary information from the XML and parses it as needed for displaying to the iPhone user.

The web service was written using Microsoft's Windows Communication Foundation (WCF)[5] framework, which handles all transformations from data to and from the XML. WCF also requires minimal configuration. The web service does communication using SOAP[6] which allows for a header and a body portion of the message.

Each request must include the following in the SOAP Header:

```
<o:UsernameToken u:Id="uuid-a1b557d3-6b53-4b6b-b9d8-3eb15069721b-1">
   <o:Username>[UserId]</o:Username>
   <o:Password o:Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">[MobileKey]</o:Password>
</o:UsernameToken>
```

The use of the authentication information in the header serves as an alternative to session variables. This simplifies the system, and makes eMESA more flexible in its deployment options. For references on the format of data in the SOAP requests, please see Appendix 1.

b. Phase 3 (iPhone application)

The iPhone application has mobile users classes recreated because it has no direct access to the eMESA framework. The iPhone will not store any information from these classes, but they are used just to keep an Object Oriented approach to the system (rather than use a lot of variables for the data in the framework objects).

Only a few of the classes from the eMESA Framework exist on the iPhone as the ones there are meant only to be the foundation for a more robust iPhone application to be continued by DTS in the future. The two main classes are Users Profiles and KPIs.

The User profile has fields including a user's full name, phone number, email address, title, employee number, work group, primary role, and site.

The iPhone application has three different screens--an initial login screen with a button that prompts authentication with the server, a user information tab that displays user profile information received from the eMESA system in a table, and a KPIs tab that displays statistical data from eMESA for a user in a table.

Each class that is reimplemented from the eMESA framework into the iPhone has field names that are the same as the wrapper tags in the XML received from the web service. This makes parsing the XML and handling the data simpler. When parsing the XML, the iPhone SDK is able to change the field of an object using a variable name (meaning if you have the variable as "Name", you can change the Name field using that variable).

V. Implementation

A. Web Service

WCF was chosen for the type of web service due to its extensibility. Without any changes in the code, or recompilation it can be changed from a web service to a Windows service. The code can also able be used within the eMESA framework without a web service call (eliminating an extra set of authentication codes). This makes future changes in eMESA simpler. DTS hopes to place all the logic from the eMESA site into the web services, thus having the logic in only one location.

All changes to the eMESA framework were placed under unit testing as to verify the framework does not have bugs. The web services also had tests written for them in order to verify they work without the need to use the iPhone application. If there were problems with the iPhone application it would have been harder to identify whether the issue was from the iPhone application or the web service. The unit tests performed are listed in Appendix 2.

B. iPhone Application

The 3.0 version of the iPhone SDK was chosen so to keep eMESA at the forefront of technology. The SDK was released shortly before work on the project began, and version 3.0 of the iPhone operating system was released during work on the project. This will ensure that eMESA will be able to work with the latest technology.

VI. Conclusions and Future Directions

The project was designed as a building block for a full eMESA iPhone application. The employees at DTS will be able to build off of the work completed and implement more features of eMESA into the iPhone application. It will have the same capabilities as the eMESA web site, including but not limited to: work orders (creation, modification, viewing and closure), and requisition (creation and viewing).

The web service installed allows for use of eMESA on other mobile platforms as well by the same calls to the web services as used by the iPhone application. This allows for the extension to even more mobile devices, since the field “MobileType” is logged by the web service as a means of showing its versatility.

Glossary

[1]**KPI**: Key Performance Indicators; Data metrics (such as work orders created, work orders closed, etc) that are collected and used to compare performance levels between different time periods
[2]**Role**: Users are assigned roles that define what permissions they have in the eMESA system
[3]**Site**: location that work is done (taken from client backend systems)
[4]**XML**: A general purpose specification for a set of data used in markup languages
[5]**WCF:** Windows Communication Foundation, a framework made by Microsoft for deploying web and Windows services
[6]**SOAP:** Simple Object Access Protocol, an XML design for communication between computer systems (most often used in web services)

Appendix 1

SOAP XML Schemas

Authentication Request:

```
<MobileLogin xmlns="http://tempuri.org/">
  <request xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web services"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <a:MobileKey>Guid</b:MobileKey>
    <a:MobileType>String</b:MobileType>
    <a:MobileVersion>String</b:MobileVersion>
  </request>
</MobileLogin>
```

Authentication Response:

```
<MobileLoginResponse xmlns="http://tempuri.org/">
  <MobileLoginResult xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web
services" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <a:SiteId>Long</b:SiteId>
    <a:UserName>String</b:UserName>
    <a:Valid>Bool</b:Valid>
  </MobileLoginResult>
</MobileLoginResponse>
```

KPI List Request:

No Data or:

```
<List xmlns="http://tempuri.org/">
  <request xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web services"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <a:PageNumber>Int(0 is default)</b:PageNumber>
  </request>
</List>
```

KPI List Response:

```
<ListResponse xmlns="http://tempuri.org/">
  <ListResult xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web services"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <a:LastUpdated>String</b:LastUpdated>
  <a:kpis>
    <a:KPI>
      <a:CurrentPeriod>String</b:CurrentPeriod>
```

```
        <a:CurrentValue>Int</b:CurrentValue>
        <a:Name>String</b:Name>
        <a:PreviousPeriod>String</b:PreviousPeriod>
        <a:PreviousValue>Int</b:PreviousValue>
      </b:KPI>
   </b:kpis>
   </ListResult>
</ListResponse>
```

KPI Update Request:
No Data

KPI Update Response:
No Data

User ViewProfile Request:
No Data or:

```
<ViewProfile xmlns="http://tempuri.org/">
   <request xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web services"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
      <a:SiteId>Int(0 is default)</b:SiteId>
   </request>
</ViewProfile>
```

User ViewProfile Response:

```
<ViewProfileResponse xmlns="http://tempuri.org/">
   <ViewProfileResult xmlns:b="http://schemas.datacontract.org/2004/07/eMESA.web
services" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
      <a:Email>String</b:Email>
      <a:EmployeeNumber>String</b:EmployeeNumber>
      <a:FirstName>String</b:FirstName>
      <a:LastName>String</b:LastName>
      <a:MobilePhone>String</b:MobilePhone>
      <a:OfficePhone>String</b:OfficePhone>
      <a:PrimaryRole>String</b:PrimaryRole>
      <a:Site>String</b:Site>
      <a:Title>String</b:Title>
      <a:WorkGroup>String</b:WorkGroup>
   </ViewProfileResult>
</ViewProfileResponse>
```

Appendix 2

Unit Tests

eMESA Framework:

UserProfile:

MobileProfileAdd: Verifying that adding mobile profiles works (multiple ones dont overwrite each other)
MobileProfileAddOwner: Verifying that a new mobile profiles owner is set properly

UserMobileProfile:

Equals: Verify that self reflection equals was overridden properly
EqaualsOtherObject: verify that equals object of other type returns false
EqualsOtherMobileProfile: verify that two different MobileUserProfileObjects are not equal
NaturalKey: verify that the natural key (for database purposes) is calculated by the framework is calculated properly
Expired: Verifying that the IsExpired function works properly

Web Service:

TestInvalidCredentialsThrowsException: verify that invalid credentials dont authenticate
TestNoCredentailsThrowsException: verify that no credentials dont authenticate
TestValidCredentials: verify that valid credentials authenticates
TestServiceResponses: verify that services send a response
(All web services ran through a packet sniffer to check data, as it was dependent on info in the database)