2012

DESIGN AND IMPLEMENTATION OF MOBILE NEWS SERVICES: SUPPORTING SOCIAL NETWORKING FEATURES

Christos K. Georgiadis
University of Macedonia, geor@uom.gr

Follow this and additional works at: http://aisel.aisnet.org/icmb2012

DESIGN AND IMPLEMENTATION OF MOBILE NEWS SERVICES: SUPPORTING SOCIAL NETWORKING FEATURES

Georgiadis, Christos K., University of Macedonia, Egnatia Street 156, 54006 Thessaloniki, Greece, geor@uom.gr

Abstract

This paper discusses and implements features both for a mobile Web and for a mobile native application, capable to provide news services enriched with indicative social networking elements. The objective was to facilitate the involvement of mobile users to produce and upload new content on the News Website and to facilitate their participation in the promotion and evaluation of the existing content. A significant addition was the implementation of an application for the automatic storing of articles derived from external sources, such as RSS/Atom feeds and emails. The results of our work are primarily related with the investigation and understanding of serious technical challenges, as well as the eventual development of proper solutions regarding the support of the 'participative relation' of mobile users with news-oriented Web content.

Keywords: mobile news services, social networks, syndication, mobile applications, mobile Web.

1 Introduction

Participative Web (or Web 2.0) applications encourage users to contribute to the production of richer and more dynamic content. The end-result for the users-authors are that they are allowed to add content of various forms as well as categorize and rate content. Moreover, the Web has fostered the emergence of 'news infomediaries', which is applications/sites that work in publishing material already published by a third party, acting as specialists in information brokering (Goyette-Côté et al., 2012). Characteristic examples are the portals of popular Internet service providers (such as MSN–the Microsoft Network) and the news aggregator services (such as the Google News service). In the latter, advanced features for searching articles and sorting/grouping the results are present along with provision of hyperlinks to news messages from different content websites together with the first two or three lines of the concerned message (van Loon, 2012).

Our approach mainly focuses on providing users the ability to share content and to support user generated content production. An important additional aspect of the provided functionality is the capability to pull news content from different sites and present it in a mobile environment. Thus, it provides news infomediary services to a certain extent.

Website creation was based on the popular (and easy-to-use) Joomla Content Management System (plus the mobile Joomla component). Some modifications, adaptations and additions were applied in order to enrich it with functions necessary for the needs of the participative Web. Two discrete ways for accessing the content through mobile devices are described. The first concerns the adaptation of content (depending on the device specifications) that is performed by the Website, whereas the second concerns the development of a suitable customized mobile application that allows users to access the content of the News Website. For this reason, an application for the Android platform was developed in the NetBeans environment.

Finally, after using the News Website under real conditions, an evaluation of its usability was performed, both by selected users and by random users who were forwarded to the Website through popular social networks or who discovered the Website themselves through search engine results. This

evaluation was of vital importance, as the feedback received allowed correction of bugs and implementation of additional features. The results of our work are primarily related with the understanding of technical challenges and the development of proper solutions regarding the support of the "participative relation" of mobile users with the News Website. In addition, an important aspect of this work (very close to our initial objectives) is the resulting conclusions regarding the provided mobile news services that could be used as an important decision making tool by news and media professionals.

2   Social Networking: Background

Web 2.0 (Murugesan, 2007) is a collection of technologies, business strategies and social trends. Moreover, Web 2.0 defines an improved user interaction and advanced management of the Web environment and its sources, something that ensures that user applications will be more flexible (Omar et al., 2007). User interaction in applications has spread widely over the past few years, mainly owing to applications such as wikis, blogs and social networks. Users can participate in the authoring of a global encyclopaedia or become authors and publishers of important news. Finally, users can share a series of services, applications and news feeds to virtual social groups.

Social networking sites allow users to monitor existing interpersonal relationships and to develop new ones. Social networks also have apparently a significant role in e-commerce (Prescott, 2007). Many social network sites have found pioneer ways to project their services through mobile devices. Mobile users can send instant messages, check their emails and post online. The Dodge ball application, introduced by Google, allows mobile users to track the location of their friends and to reveal their own location. At the same time, it allows sending of multiple text messages to friends that are in proximity to the users so that their friends can find each other. A similar micro blogging service (as users publish only small amounts of text in each post) is offered by Twitter. Twitter allows users to send messages to a select number of friends and to receive texts and information from people whom the user follows. What is really important is that twitter offers API services that allows twitter functionality to be incorporated into other applications.

3   News Services: Requirement Analysis

Analysing typical user requirements on news-oriented Websites, we may distinguish the following user categories: visitor, member, author and administrator.

Unregistered users (visitors) read access rights mainly. Because our focus is to support social networking issues, we may allow them to share news content (articles) that they like via popular social networking sites (such as Facebook) and to rate news content.

Visitors may become members either by registering using our registration service or by certifying their identity via a social network. Members have extra access rights: they may post comments and receive newsletters by e-mail.

On the other hand, authors have content management rights: they may upload/withdraw/publish/delete news, organize their articles in categories, parameterize the presentation of their articles and manage the comments on their articles. Finally, they may upload multimedia for their articles from a variety of resources including popular social networks (such as YouTube and Picassa).

Syndication is currently a major and important part of the Internet. There are two main forms of Syndication: RSS 2.0 (Really Simple Syndication) and Atom 1.0 (Atom Syndication Format). Although they accomplish the same objectives there are some differences. RSS 2.0 is a standard copyrighted by Harvard, while Atom 1.0 has been defined in RFC4287. Atom was invented because the RSS 2.0 prototype was considered a prototype with limited expansive opportunities and capabilities (Wittenbrink, 2005). Websites that provide RSS and ATOM feeds can maintain an «open

connection» with their users. That means that users do not have to periodically visit a Website as they can simply use the aforementioned feeds to stay informed. Therefore, our approach for providing news services supports authors to import articles using RSS/Atom feeds. In this way, all users (members and visitors) may have access to content provided by syndication feeds.

4   Mobile News Services

Increased sophistication of mobile technology makes itself an ideal channel for offering valuable services to mobile users (Georgiadis, 2010). The following factors should be taken into consideration:
- A vast variety of mobile devices exists, and these devices differ (from a developer's perspective) in at least two major parameters: the screen size and the reading format.
- Internet access is required generally and not necessarily in a constant online state: mobile offline applications are increasingly gaining users' attention (Malykhina, 2011). However, certain alternatives of accessing Internet content may be used: browser-based approaches that allow users to access existing websites content (the mobile Web) and native application implementations.

4.1   Web browser–Web technology

Most mobile devices have a preinstalled Web browser that allows users to access Websites. But there are restrictions applied by the screen size and the limited capability of typing text through a touch-screen, which imposes selective modification of appearance of content depending on the specifications of the mobile appliance that is being served by the Website. Such Websites are either developed separately from their associated desktop-oriented version or both the versions are created simultaneously by employing methodologies that support multi-platform context-aware Websites requiring an extensive engineering effort (van Woelsen, 2011).

The device recognition can be achieved by reading the HEADERS variables of the SERVER. The assignment of the proper values (concerning the device's specifications for suitable content presentation) can be handled in four discrete ways:

1. Addressing all devices in the same way. Obviously, this is the most cursory and ineffective treatment.

2. Attaching predetermined values (suitable for a list of well-known mobile devices).

3. Redirecting and uploading proper templates by locating suitable domains based on abstract mobile devices categories.

4. Recognition and extraction of the actual device characteristics and assignment of all required ones to specific important parameters that define the way the Website is presented.

For the first three cases, the Website administrator may also develop predetermined templates for a list of known mobiles, based on their characteristics, so that after the recognition of the device the appropriate template can be loaded.

For the last case, the TERA Wireless Universal Resource File class (http://www.tera-wurfl.com/wiki/index.php/Main_Page) can be used. Tera WRUFL is a mechanism for recognizing the specifications of mobile devices, and it can support over 800 known mobile devices that are equipped with a wireless antenna. After recognizing the mobile device, it automatically changes the way the Website is presented so that the content can be easily accessed by mobile users. Context-aware adaptation for web browsing on handheld devices would be one of the major new functions of smart handheld devices in the near future (Zhang and Lai, 2011).

4.2 Mobile native application–The customized approach

An additional application is developed, tailored for a specific mobile platform, aiming to cover complete user requirements by exploiting the advanced characteristics of the current mobile device. This approach, apart from the obvious disadvantages (building a different application for each platform is very expensive if written in each native language), it has a notable benefit: it is widely acceptable in the mobile research community that the intention of mobile users to adopt basic Internet services largely depends on the importance of using similar services in the mobile domain as on the fixed Internet (De Reuver et al., 2011). The Web technology stack has not yet achieved the level of performance we can attain with native code (Charland and Leroux, 2011), thus mobile users will continue to seek native experiences in order to enjoy services of similar quality with the fixed Internet.

5 Mobile News Website

5.1 Mobile Joomla

Mobile Joomla is a powerful open source component regarding mobile device-oriented adaptations for Joomla Websites. It is available in the BackEnd Joomla environment, and it allows Website administrators to choose their preferred way of assigning templates (using standardized templates, customizing the predetermined templates or supporting TERA WURFL mechanisms). Some well-known predefined templates are smartphone, iPhone, WAP and iMobile templates. Mobile Joomla allows adding or removing menus, modules or other user interface components. Moreover, it favours the selective placement of components: it contains modules capable of modifying the export of the Website content and place it in suitable template containers. For example, it is possible to only present the introduction of an article and hide information such as the name of the author or the date on which the article was published.

The main Joomla tables (related to the news-oriented Web content) are the Content table, the Categories table and the Sections table. Content (or article) is the basic entity that belongs to a category which in turn belongs to a section. One section may contain many categories and one category may contain many articles.

5.2 Publishing content through RSS/Atom feeds

RSS feeds for a News Website could be used for indexing in order to create news articles from various sites and sources clustered together in one Website. The first major modification/improvement on Joomla's default functionality was storing feeds in Website's database. Subsequently, the stored articles might be filtered by keywords or even other feeds' metadata (such as date or origin). Joomla has a function named Jfactory::getXMLparser which accepts an RSS or ATOM feed as input and returns a PHP-Based RSS and Atom Feed Framework object, namely the SimplePic. An indicative approach for storing RSS feeds in the Joomla content table is by using its fields as follows:

```
$contenttable->title=$title;
$contenttable->permalink=$rssurl->section_id;
$contenttable->catid=$rssurl->cat_id;
$text=$item->get_content();
$contenttable->created=$item->get_date('Y-m-d H:i:s');
$contenttable->publish_up=$item->get_date('Y-m-d H:i:s');
```

We may preserve "Content" Joomla table using a secondary table (feedtable) in which information regarding the author, the hyperlink of the source and other information will be stored.

```
$premalink=$item->get_permalink();
```

```
$feedtable->sourcelink=$premalink;
```

The capability to have multiple feeds, which can be enhanced with filters, could work as a mechanism for article indexing from selected sources. On the other hand, indexing could also be achieved through social network applications such as the 'Yahoo pipes' mashup. Fig. 1 below depicts a Yahoo pipe. It receives feeds from many well-known news sites, which in turn are filtered by both keywords and publication date. After that, feeds are sorted with reverse chronological order.

An author may adjust a mashup in Yahoo pipes for his own needs and interests and then send those feeds to the news Website in an RSS or JSON format. In any case, if the advantages of permanent storage are required, these data have to be stored in the site's database.

5.3   Publishing content via email

The second indicative functionality that we have investigated and developed was related to emails. Users with rights to publish articles (both members and authors) may in this way send an article via email, and that article will be automatically published in the appropriate section. All that is needed is the author's approval.
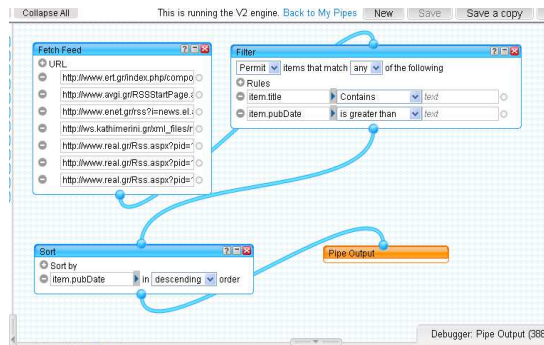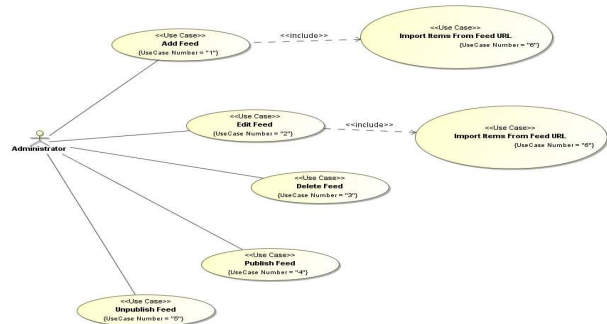

Figure 1–Yahoo pipes   *Figure 2–User cases for managing URL feeds*

The usual case of uploading an article via email without the automated method would be as follows:
- An email is send by a certain member or author. The email includes the article and an attached image and may have any fonts or text format.
- Author has to save the image locally, upload it through ftp to the Website, create the article, copy the text and add the image as an object in the article.

In the automated method, the whole process of the second step is simplified, as follows:
- The automated system creates an article based on the delivered email.
- Author approves (or disapproves) the publication of the article.

In this case, the necessary library for the management of the emails is provided directly by PHP. The library is called IMAP and it provides the functions that allow the handling of the email structure. An email parser is actually developed; it is capable to split the email into discrete pieces (title, main body and attachments). These pieces may then be used for the creation of the article just like in the case of the RSS feeds. Consequently, this integrated approach of handling both RSS/Atoms and emails leads to the formalization of certain user cases for managing URL feeds, as Fig. 2 illustrates.

The storage of the URL of a certain feed or of a certain email triggers (triggerEvent) the processes of importing RSS items (SimpleRssFeedImport) or emails (emailRssFeedImport) in the 'Content' table and simultaneously the update in the secondary table (feedtable) of the additional fields that cannot be saved in the basic table (Fig. 3).

Fig. 4 depicts the basic steps of the function saveEmail ($mbox, $mid, $rssurl) towards updating content and feedtable tables. Function's inputs are the mailbox, the message id and the 'rsstoa' object which contains parameters (stored in the corresponding xml file) concerning important settings such as the category in which the mail should be imported, the reference of the source, the date of publication, if email should be published on the front page of the Website (Fig.5).
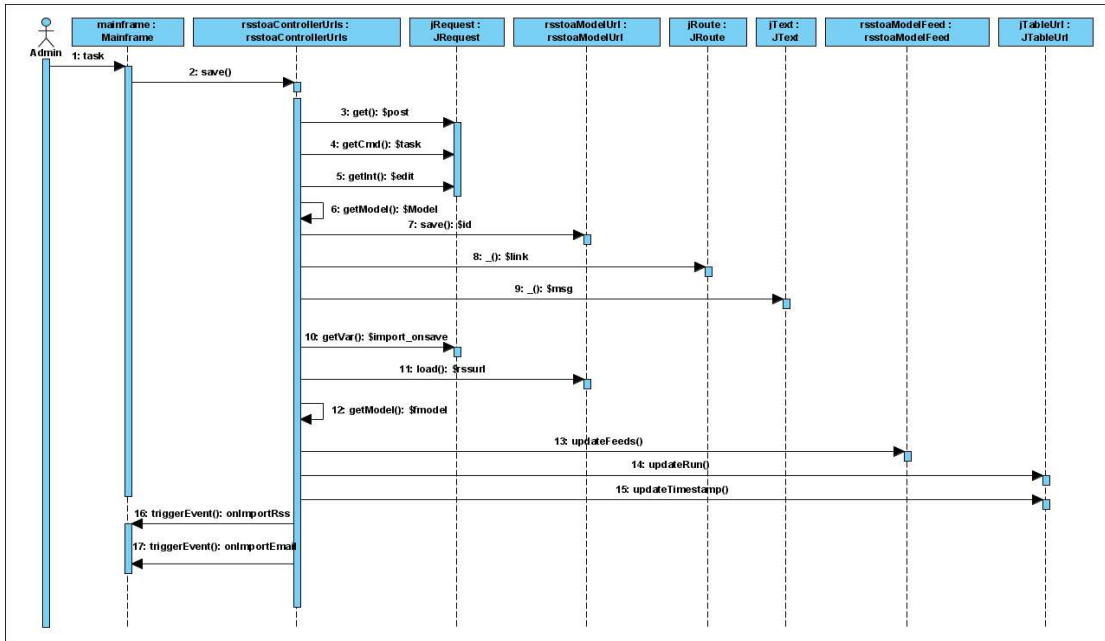

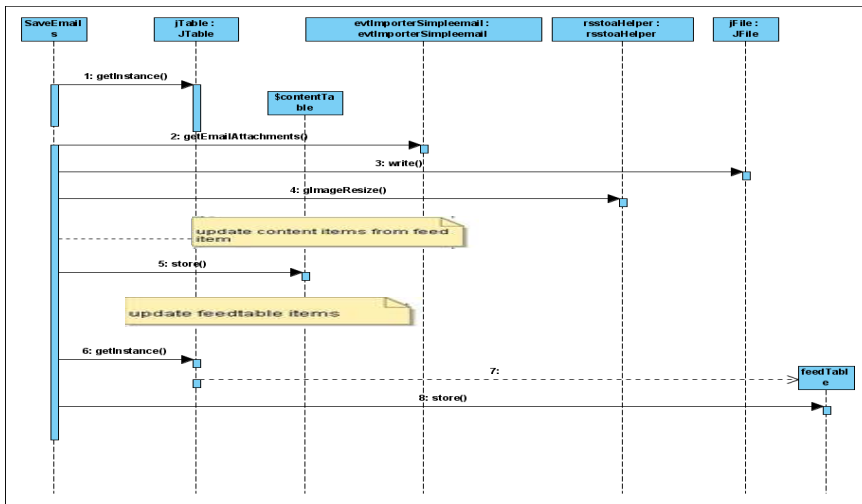*Figure 3–Sequential diagram: action Save/Edit Feed (edit rss/email FeedUrl)*

Figure 4–Sequential diagram: function saveEmail (updating Content and feedtable tables)

6 Mobile (Native) Application

The customized application is written in Java and is developed in NetBeans 7.0 IDE using Android Software Development Kit. It was developed to support all required news services (the handling of the news articles published on Website) through mobile devices. Indicative functions are as follows:

- The retrieval and presentation of articles' categories.
- The retrieval and presentation of the ten most recent articles.
- The retrieval and presentation of the ten most recent articles per category.
- The publishing and deletion of an article.

It exploits the server-side XML-RPC methods and uses the client-side XML-RPC library for the Android platform.

6.1   XML-RPC

XML-RPC (remote procedure call) protocol is a specification and a series of software implementations independent of the operational system capable of being executed in different environments to call remote service methods via the Internet. It uses XML coding at the transport layer. Joomla is equipped with the XML-RPC library that runs as an API Service. In addition, the MetaBlogging XML-RPC plugin (Fig. 6) provides powerful capabilities to handle these methods from any client device, including mobile devices (Fig. 7).
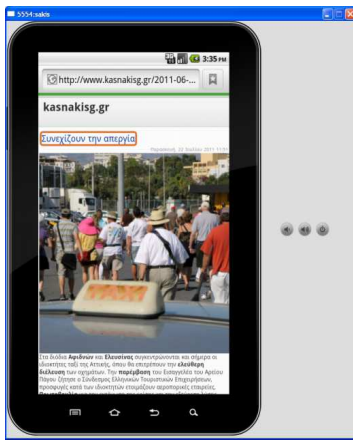

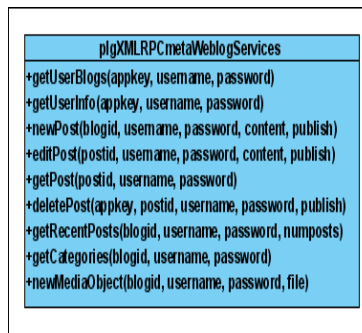*Figure 5–Website's Frontpage via mini browser (WebView) Mobile Web*

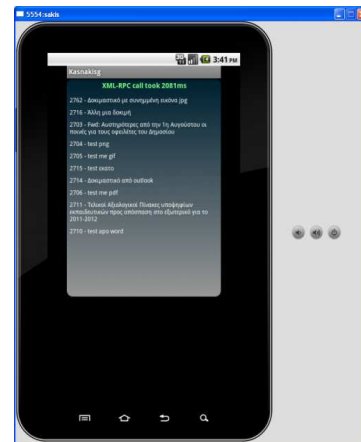*Figure 6–MetaBlogging XML-RPC methods*

Figure 7–List of articles per category (XML-RPC) Mobile Native Application

A potential alternative of the XML-RPC approach is the use of REST technology. The REST protocol uses HTTP GET and HTTP PUT methods for the communication between the client and the server. The reference to the API is done through a URL. Every REST service has its own URL, making it possible to use cache copies and bookmarks unlike with XML-RPC. On the other hand, XML-RPC remotely calls procedures that are server-based. Some examples are the getUser(), addUser(), removeUser(), updateUser(), updateLocation() and listUsers() procedures. The clients create XML requests and are capable of translating every XML response.

Although more effort is required in XML-RPC to convert data structures to XML and vice versa, remote procedure call technology has a big advantage in mobile settings: the client does not have to be a Web browser. It can be any application, with the most obvious choice in our work to be a mobile application.

6.2   Mobile news services–The activities of the Android application

The main services that the customized application may provide are listed below:

```
<activity android:name=".ViewArticle"></activity>
<activity android:name=".ListArticles"></activity>
<activity android:name=".NewArticle"></activity>
```

```
<activity android:name=".DeleteArticle"></activity>
<activity android:name=".ListCategories"></activity>
<activity android:name=".LatestArticles"></activity>
```

Let us provide at this point a few comments on indicative code, which uses XML-RPC methods for retrieving a list of the most recent articles (Latest Articles activity):

```
public void onCreate(Bundle savedInstanceState)
  {   super.onCreate(savedInstanceState);
```

1. The first step is to create a client for the XML-RPC service that runs on the Joomla server:

```
uri = URI.create("http://www.kasnakisg.gr/xmlrpc/index.php");
        client = new XMLRPCClient(uri);
```

2. The second step is to create the appropriate user interface (UI):

```
setContentView(R.layout.main);
katigoria=this.getIntent().getExtras().getString("kat");
int kat=Integer.parseInt(katigoria);
errorDrawable = getResources().getDrawable(R.drawable.error);
errorDrawable.setBounds(0, 0, errorDrawable.getIntrinsicWidth(), errorDrawable.getIntrinsicHeight());
status = (TextView) findViewById(R.id.status);
tests = (ListView) findViewById(R.id.tests);
final ArrayAdapter<String> adapter = new TestAdapter(this, R.layout.test, R.id.title);
```

3. Remotely calling the getRecentPosts:

```
XMLRPCMethod method = new XMLRPCMethod("metaWeblog. getRecentPosts",
    new XMLRPCMethodCallback() {
              public void callFinished(Object result) {
```

4. At this step, we receive as response an array of objects. The number of rows is examined, and in case it exceeds ten (10), the array is limited to the first ten rows only. If the number of rows is null, the user receives an appropriate message:

```
Object[] arr=(Object[]) result;
int i;
if  (arr.length>=10) numart1=10;
    else  numart1=arr.length;
```

5. The mobile application then decodes the structure into pairs of fields in the form of "Fields name"-"Fields price":

```
for(i=0;i<numart1;i++) {
          Map<String, Object> map = (Map<String, Object>) arr[i];
```

6. Every row of the array is transformed into a new object for the UI list structure, the 'adapter'. At the same time, all the necessary fields are saved to ensure that the onClick procedure works flawlessly:

```
adapter.add(map.get("postid").toString()+" - "+map.get("title").toString());
desc.add(map.get("description").toString());   }  }  });
```

7. The parameters have to be placed into an array:

```
Object[] params = { "",user, pass,10, kat,   };
```

8. The call is achieved through the method.call procedure of XML-RPC:

```
method.call(params);
```

9. The list object 'adapter' fills the tests list container, as well as the testListener functionality.

```
tests.setAdapter(adapter);
```

```
tests.setOnItemClickListener(testListener);  }
```

10. When onClick is activated, a specific activity class is called with the appropriate parameter input.

```
OnItemClickListener testListener = new OnItemClickListener() {
   public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
      Bundle bundle = new Bundle();
```

11. The parameter test receives a string value that is the header for the html coding and the article's short description.

```
Bundle.putString("test", header+desc.get(position).toString());
```

12. The function ViewArticle is then called; it retrieves the content of the article.

```
startActivity(new Intent(view.getContext(), ViewArticle.class).putExtras(bundle));  }  };
```

13. As shown in the code below (part of the onCreate method of ViewArticle activity), the content of the article is being projected through a mini browser (WebView type).

```
...
browser=(WebView)findViewById(R.id.webkit);
strUrl=this.getIntent().getExtras().getString("test");
browser.loadData(strUrl,"text/html", "utf-8");    }
```

6.3   On the server side...

On the server side, a request to the data base is executed and results are being received:

```
function getRecentPosts($blogid, $username, $password, $numposts) {
global $xmlrpcerruser, $xmlrpcI4, $xmlrpcInt, $xmlrpcBoolean, $xmlrpcDouble, $xmlrpcString, $xmlrpcDateTime,
$xmlrpcBase64, $xmlrpcArray, $xmlrpcStruct, $xmlrpcValue;
```

The following code is for identifying the user (with the use of a certificate):

```
if (!plgXMLRPCmetaWeblogHelper::authenticateUser($username, $password))
                  return new xmlrpcresp(0, $xmlrpcerruser+1, "Login Failed");
$user =& JUser::getInstance($username);
$aid  = plgXMLRPCmetaWeblogHelper::getUserAid($user->id);
$plugin =& JPluginHelper::getPlugin('xmlrpc','metaweblog');
$params = new JParameter( $plugin->params );
```

A data base object is being created:

```
$db =& JFactory::getDBO();
```

A query is being executed:

```
$query = 'SELECT c.id, c.title, c.alias, c.created_by, c.introtext, c.created, c.state'
        .' FROM #__content AS c'
        .' INNER JOIN #__sections AS s ON c.sectionid = s.id'
        .' INNER JOIN #__categories AS cc ON c.catid = cc.id'
        .' WHERE s.published = 1 AND cc.published = 1'
        .' AND s.access <= '.$aid .' AND cc.access <= '.$aid.' AND c.access <= '.$aid .' AND c.state >= 0'
        .' ORDER BY c.created DESC';
$db->setQuery($query, 0, $numposts);
$items = $db->loadObjectList();
```

A check for results is being performed and if results are found they undergo a proper serialization before transfer.

```
if (!$items) return new xmlrpcresp(0, $xmlrpcerruser+1, 'No posts available, or an error has occurred.' );
require_once (JPATH_SITE.DS.'components'.DS.'com_content'.DS.'helpers'.DS.'route.php');
$structArray = array();
foreach ($items as $item)    {
 $dateCreated=& new JDate($item->created);
 $articleLink= JURI::root() .(ContentHelperRoute::getArticleRoute($item->id, $item->catid, $item->sectionid));
 $structArray[] = new xmlrpcval(array(
      //'dateCreated'           => new xmlrpcval($dateCreated->toISO8601(), 'dateTime.iso8601'),
         'title'                => new xmlrpcval($item->title),
         'description'          => new xmlrpcval($item->introtext),
         'userid'               => new xmlrpcval($item->created_by),
         'postid'               => new xmlrpcval($item->id),
         'link'                 => new xmlrpcval($articleLink),
         'permaLink'            => new xmlrpcval($articleLink)
         ), $xmlrpcStruct);  }
return new xmlrpcresp(new xmlrpcval( $structArray , $xmlrpcArray));            }
```

The return is an array of articles having selected fields. All activities of the Android application follow the model described above.

6.4 Use of RSS feeds–AndroidXML

Since the Website can provide feeds for articles, we can take advantage of this fact so that the content will be presented to the mobile user more concisely. User can choose from a list the category that she is interested in, and only titles of articles that belong in that category will be presented. After selecting a title, the whole article can be projected to the user (Fig. 8).
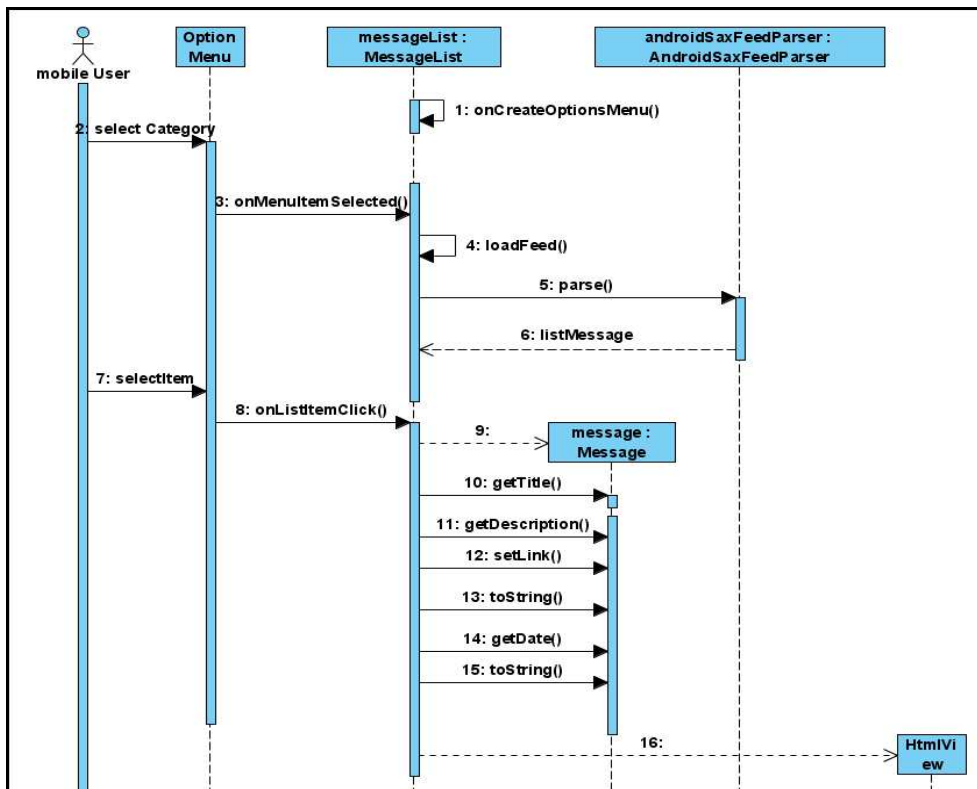

Figure 8–Sequential diagram: AndroidXML

7   Trials and Conclusions

A series of trials and tests were performed to ensure that the website operates flawlessly.
- Articles were published using RSS feeds and regular emails.
- We allowed selected users to comment on the articles, to avoid future problems.
- Articles were shared in well-known social networking sites.
- Articles were handled through the mobile application.

In order to complete the aforementioned tests we created accounts in well-known social networks such as Facebook, Twitter and Google. In addition, to take advantage of their social features, monitoring services such as Google Analytics and Google WebMasters were activated.

7.1   Conclusions–Discussion on business aspects

A single tweet of an article in Twitter resulted in the instant rise of people visiting the website by 10-16 visitors. Especially when the subject of the tweet was about important news the aforementioned rise of visitors lasted longer and attracted followers. The fact that the news were taken from other websites is the reason why visitors didn't stay longer on the website.

On the other hand, on Facebook, the article sharing was done in a more confined group of people so it was more difficult for the news feed to spread wider. Facebook's live stream is an interesting option for the direct commenting on topics by various users that can be of major importance for advertising since user's comments are visible for all his Facebook friends.

In order to allow a user to comment on an article his identity must be certified through a social network. Users found it easier to participate and use every feature of our website with that way of certification.

The existence of RSS and other feeds allows users to be informed on various subjects faster and more completely. Nevertheless, the republishing simply of articles that were firstly published in other websites, decreases the website's value as a source of information. Consequently, to increase the business perspectives of news intermediary services more attention has to be given on organizing the selection, organization, hierarchical ranking and distribution of the news published by others. This new 'industry of access' to content produced externally may raise despite questions concerning the pluralism of news. Current research indeed argues that, while it appears that the Internet has theoretically favoured the emergence of countless 'new' news channels, it seems to not have fostered as many new sources, so much as new gateways to access the same contents produced by a few companies (Goyette-Côté et al., 2012).

It is worth mentioning at this point that significant business-related pitfall that a news infomediary service developer has to avoid: given the well-known heavy criticism for copyright infringement by the Google News service (Laurent, 2011), our approach favours portal-oriented procedures: agreements/deals with news providers, similar to the ones a press company agrees with a news agency–a subscription to a newswire with an agreement to use a certain number of news items each day.

Some final notes: People working in the media were interested in the site's capability to publish articles directly from emails. They believe that this is a compelling feature that automates and simplifies a procedure that they use every day. Finally, there was a great interest for the Android application and especially for its capability to publish articles through a mobile device. Many visitors asked for a presentation of the application's full capabilities, while many asked if a port in the IOS platform was in plans.

7.2   Future plans

- The development of a mobile application that would enable more features like the geographical identification of the source of news feed, the sharing of news based on geographical criteria and the use of REST services that are offered in many social networking sites.
- The optimization of the application for publishing articles through emails and or relative services.
- The development of an innovative Web Service for a Greek service. Key considerations: a) it is of great importance to publish a news feed fast; b) there are many sources of information, but many of them probably have the same source; c) the amount of readers is related with the quality of a news source, but occasionally a source may not be easily accessible to the wider audience; d) people have the power to express their thoughts through a plethora of channels, but only few opt to do it even in important subjects and e) advertisement is often a factor that undermines news feeds.

References

Goyette-Côté, M.-O., Carbasse, R., George, E. (2012). Converging Journalism, Journalism Studies, iFirst Article, 1-10.

van Loon, S. (2012). The Power of Google: First Mover Advantage or Abuse of a Dominant Position? In A. Lopez-Tarruella (ed.), Google and the Law, Information Technology and Law Series 22, pp. 9-36, T.M.C. Asser Press, The Hague, The Netherlands.

Murugesan, S. (2007). Understanding Web 2.0. IT Professional, July/August, 34-41.

Omar, W.M., Abbas, A.D.K., Taleb-Bendiab (2007). SOAW2 for Managing the Web 2.0 Framework. IT Professional, May/June, 30-35

Prescott, L. (2007). Hitwise US Consumer Generated Media Report. White paper.

Wittenbrink, H. (2005). RSS and ATOM: Understanding and Implementing Content Feeds and Syndication. Puckt Publishing.

Georgiadis, C.K. (2010). Developing Personalized Information Services for Mobile Commerce Location-Aware Applications. Int. J. on Advances in Internet Technology, 3 (3&4), 274-283.

Malykhina, E. (2011). Surviving the Unwired Wild: 6 Mobile Offline Apps Make a Smart Phone an Essential Part of a Camper's Tool Kit. Scientific American, August 11, http://www.scientificamerican.com/article.cfm?id=mobile-apps-camping

van Woelsen, W., Casteleyn, S., de Troyer, O. (2011). A Generic Approach for On-The-Fly Adding of Context-aware Features to Existing Websites. In Proceedings of the 22nd ACM Conference on Hypertext and hypermedia (HT '11), pp. 143-152, ACM, USA, New York.

Zhang, D., Lai, J. (2011). Can Convenience and Effectiveness Converge in Mobile Web? A Critique of the State-of-the-Art Adaptation Techniques for Web Navigation on Mobile Handheld Devices. Int. J. of Human-Computer Interaction, 27:12, 1133-1160.

de Reuver, M., Ongena, G., Bowman, H. (2011). Should mobile Internet services be an extension of the fixed Internet? Context-of-use, fixed-mobile reinforcement and personal innovativeness. In Proceedings of the 10th International Conference on Mobile Business, pp. 16-25, IEEE CS, Italy, Como.

Charland, A., Leroux, B. (2011). Mobile application development: Web vs. Native. Commun. ACM 54, 5 (May 2011), 49-53.

Laurent, P. (2011). Copiepresse SCRL & alii v. Google Inc. - In its decision of 5 May 2011, the Brussels Court of Appeal confirms the prohibitory injunction order banning Google News and Google's "in cache" function. Computer Law & Security Review 27, 542-545.