Wrapper-based Personalized Mobile Meta Portal

Yue Shan Chang
Department of Computer Science and Information Engineering,
National Taipei University
151 University Road, Sanhsia, Taipei county, 237, Taiwan R.O.C.
ysc@mail.ntpu.edu.tw

Abstract

In this paper, we present Amor, a novel platform providing reconfigurable and personalized mobile meta portal based on adaptive wrappers that can loosely tie many portals or web pages simultaneously and access desired information. Amor not only provides mobile users with a platform for precisely retrieving personally desired information from multiple portals or web sites using a handheld device, but also a mechanism for dynamically reconfiguring user preferences anytime and anywhere.

Keywords: Mobile Web, Mobile Portal, Personalized, Wrapper, Wrapper Generator, Meta Portal

1. Introduction

It is evident that the use of Internet portals in a) browsing categories of information, b) search engines and c) e-mail boxes or auction services, is very popular. Portals are individually designed for a range of services. Most users are content to select portals or web pages of particular interest and then bookmark them in their browser. So, what kind of portal to visit depends on what personal preferences are being expressed. For example, one user may like to choose Google’s search engine to search the Internet, to visit CNN portal to browse the international and sports news only, and to receive e-mail using their campus’s e-mail box, while another may like to use Yahoo’s search engine, and to browse NASDAQ’s stock information only. Portals generally offer abundant, but of course sometimes redundant, information (e.g., advertisements, pictures and irrelevant information). It is not necessary when accessing desired information to download the whole web page to a mobile device such as an XHTML-enabled cell-phone, a PDA cell phone, or PDA, all of which have small screens and low bandwidths, because downloading the whole page causes inconvenience, increases navigation costs and reduces the attractions of surfing the Web with hand-held devices.

Among the solutions proposed for such devices are the portals offered by Yahoo and Google. The mPERSONA [11] has a personalized portal for wireless users based on multi agents, which successfully tackles some of the problems, as for example when it reduces the click count. Personalized Pocket Directories [4] offer a choice of desired personal catalogues and information. These are examples of efficient web personalization technologies for mobile devices. Anderson [1] proposes a web site personalizer that automatically tailors individual demands to web page contents. Once the mobile user has browsed the content on the desktop, the personalizer observes and tailors it, gradually fitting it to suit the demands.

However, most of them need tying in tightly with specific web sites or portals. They extract the desired information from their web site contents and construct individualized service, but they can only personalize the information that resides in their web site. In addition, existing mobile portals usually offer much unnecessary information and do not cover the variety of demands that a mobile user may have. So, the real need is for mobile meta portals that enable users to access information from multiple portals to suit personal preferences. In this article, we present Amor, a novel platform providing reconfigurable and personalized mobile meta portal based on adaptive wrappers that can loosely tie many portals or web pages simultaneously and access desired information. Amor not only provides mobile users with a platform for precisely retrieving personally desired information from multiple portals or web sites using a handheld device, but also a mechanism for dynamically reconfiguring user preferences anytime and anywhere.

The major difference between Amor and other approaches is that users can dynamically configure personal portals according to preference and retrieve information that fully satisfies their demands no matter where the desired information reside. The prototyping of Amor has shown that users can browse many
web sites in a portal, can set user preference and can accurately, efficiently and cost-effectively retrieve desired contents.

2. Internet Portal

As is known, web portals are amply fit for their purpose, offering access to very large bank of information organized as a collection of items or information units (IU) [10]. Thus the user interest in a web site relates to the information units of particular character rather than to the entire site. A wrapper can be designed to extract particular IUs, discarding any undesired ones.

Observation of an Internet portal shows that there are three categories of service: Query based (QS), Link based (LiS) and Login based (LoS). With QS, users input a query string, via a search engine, to retrieve required information. The list of results may extend to cover multiple pages. Secondly LiS, with which users browse the information directly by clicking hyperlinks relating to particular aspects of information, e.g. news, weather, stocks, or sports and so on and, again, a typical web page here may contain multiple information units of the same kind. Thirdly, LoS, a service such as the Yahoo family or an e-mail box, which requires additional user authentication is thus available only to portal members. The results here may also cover multiple pages of desired IUs.

2.1 Information unit extraction

A wrapper is a frequently used web component for retrieving information, a specialised program routine that can automatically extract data from Internet web sites and convert them into structured formats [12]. Extracting information from an IU by a wrapper needs extraction rules, schemas of content, and query templates [3]. For example, using the Google search engine to search web sites for the string “java”, the query template will be the string "http://www.google.com.tw/search?hl=zh-TW&q=java&meta="", the schema of content will be URL, TITLE and DESCRIPTION, and the extraction rule will be the HTML tag, e.g. <a></a><table><tr><td></td></tr></table>, enveloping the information desired by the user. The results generated by the wrapper can be packed into a structured format, such as XML. Obviously, inherent characteristics make it difficult to extract a variety of information units, e.g. QS, LiS, and LoS, by the same wrapper.


Developing many wrappers systematically and manually is difficult. Fortunately, given the level of present technology, dozens of commercial and non-commercial wrapper development tools [10] are to hand to greatly ease wrapper generation [6]. The system administrator generates multiple wrappers to suit the characteristics of the IUs. There is, though, an important issue. Kushmerick [9] has estimated that over 44% of web pages change their page layout or tag during six months. If a wrapper fails to extract desired information because of a changed page layout or tag format, then there is a need for an adaptive wrapper (AW) with the power of self recovery and automatic repair [2, 5] in the dynamic environment. For this reason, we implement a technology for generating different kinds of wrappers (QS, LiS, and LoS) in the prototype. The wrapper implementation for different kind of services has the distinct differences described as follows:

- **Extraction rule and schema.** Each wrapper has its own extraction rule and schema for retrieving specific IU.
- **Wrapper parameter:** The need for a wrapper parameter in the query template varies. The LiS service doesn’t need extra one because it can access targeted information units gradually via hyperlinks, whereas with the QS service the user must key at least one parameter to query backend search engines. For the LoS service, a user account and password are needed as the input parameter for logging into backend servers.
- **Session and security handling:** In general, while wrappers are not needed for keeping the session and security with the server in QS and LiS services, this not the case for LoS. Most of LoS services only offer to their members and use secure HTTP protocol to connect to backend server. The wrapper also needs to maintain the secure HTTP connection.
- **Recovery and repair:** The technology of recovery and repair [5] in QoS and LiS is different from LoS because it has explicit domain ontology.

3.1 Information relationships.

The web pages are hierarchically organized and linked using hyperlink.
Accessing desired IU in the target page usually means browsing more than one page. For example, when browsing say for a specific page of social news, users must navigate multiple pages through a variety of news category pages to reach the target of interest. The IUs therefore come into being in relationship to the information. As mentioned above, a wrapper only retrieves a specific IU. Navigating multiple pages call for multiple wrappers to retrieve targets of interest sequentially. For this purpose, we adapt a wrapper linkage mechanism to maintain the relationship between the wrappers and the information being targeted.

### 3.2 Adaptive Wrapper Architecture

Fig. 1 shows the architecture of our proposed adaptable wrapper. The wrapper consists of three modules described as follows:

1. **Input/Out Encapsulation module (IOEM):** the module is responsible for translating user query into query string of backend NIS and retrieving desired information form NIS response. All retrieved results are packed as XML format for future processing.

2. **Detection and Recovery Module (DRM):** the module is mainly parsing and validating the change of web source. When the change detected, the module will raise the recovery phase and re-deduction the extraction rule for extractor.

3. **NIS module (NM):** the module is responsible for connecting to backend NIS.

Here will only describe the component of DRM and its operation. As mentioned above, the DRM will raise recovery phase while web page change and retrieve new extraction rule for extractor. It consists of four components that are depicted as follows:

1. **Recovery Manager (RM):** RM is a major component of DRM and is used to control the recovery phase. When the RM receives a recovery event from Parser/Validator, it initial recovery operations that include new extraction rule discovery and update the rule for extractor.

2. New Rule Discovery component: This component is to deduce new extraction rule from new document using our new rule discovery algorithm.

3. **Validating-Recovering Data Repository (for short VRDR):** This component stores all recovery necessary data including query string (QS), extraction rule, web page that fetched at wrapper initialized, and schema of target NIS.

4. **Parser/Validator:** Web page retrieved from NIS will be parsed to check and verify whether validation or not. If the web page changed, the component will notify the recovery manager to enforce recovery phase; otherwise it will filter most redundant part of web page and send it to IOE module.

Fig.2 shows adaptive wrapper execution flow. First adaptive wrapper translate user query string and encapsulate associated parameter into NIS acceptable query string by a query template that in the IOEM. The translated query string is sent to DRM. DRM first save the query string into VRDR and pass to backend NIS by the NM. Then the wrapper waits for the response from backend NIS and gets the result. The Parser/Validator will verify the return page and filter out most redundant part. If the wrapper is valid, then extracts all of desired information unit and packs the result as XML format for future use and return to users. If the wrapper is invalid, the enter recovery phase to infer new extraction rule.

---

**Adaptive Wrapper execution flow**

```
Begin
  translate users query string;
  save query string;
  send the transformed query to information source;
  wait for the return and get the result;
  if (some error occurred)
    enter recovery phase;
  else
    packing result;
    return result;
  endif
End
```

---

### 3.2.1 Adaptive Wrapper Generation

The adaptive wrapper can be generated using wrapper generator. The wrapper generation is shown in Fig. 3.

The first step of wrapper generation is to select the target NIS and input its schema of desired information, as step 1 and 2 of Fig. 3. In our example the desired information is Title,
Description, and URL respectively. Schema and content in step 2 is stored in a temporary file. Step 3, 4 and 5 will parse the web page, generate extraction rule of the page like as NRD algorithm, and store the rule into a temporary file. Finally, adaptive wrapper generator generates a wrapper using previous created data and component repository as step 6 and 7.

Fig. 5: Adaptive test

Adaptive test of an adaptive wrapper is made using two analogous search engines, Yahoo and Lycos. In this test we generate two adaptive wrappers for Yahoo and Lycos respectively, because these two search engines, according to our observation, have similar result but different format that result in the extraction rule is different.

In the test we first redirect Lycos web page to Yahoo’s wrapper, as shown in Fig. 5. That obviously raises the wrapper invalid. The Yahoo’s wrapper immediately initiate NRD algorithm for learning Lycos’s extraction rule and save extraction rule into VRDR. The wrapper finally output Lycos’s result. It shows that the wrapper is adaptive for different web page.

4. Our solution: Amor

Anne Kaikkonen [7] addressed usability problems for mobile internet portals, including the technology in use, such as portal structures, navigation aids, page and path lengths, feedback for users and images. The Amor platform of our design approaches these issues to make them as user-friendly as possible. For example, we adapt XHTML for page presentation and provide following features as far as possible: informative page titles and page naming, clear page layouts, consistent navigation, double selection avoidance, click count reduction, and an Internet portal-like download process.

Fig. 3: Adaptive Wrapper Generation Flow

Fig. 4 shows a simple UI for generating adaptive wrapper. In this prototype, user input the schema and copy desired content of web page into the UI. And finally press the generator button to generate an adaptive wrapper.

Fig. 4: A Simple UI for Generating Adaptive Wrapper

3.3 Web Page Adaptive Test

Fig. 6: Mobile Internet meta portal

As mentioned above, Amor introduces adaptive wrappers that retrieve desired information while accommodating web page changes. As Fig. 6 shows, it is divided into two parts. The first is a meta-portal platform that maintains reconfiguration and personalization functions and the second consists of a large number of AWs generated by an AW generator [2] (AWG) for accurately retrieving desired information via related
portals and web pages.

4.1 Components of Amor

Here we present in Fig. 7 the important components of the Amor:

The **Taxonomy Manager** is responsible for managing the domain of user-desired information and support system for providing accurate information retrieval. The AW generated for a specific IU can be classified according to the taxonomy of a general portal. The administrator creates a certain amount of domain-dependent information. The information is stored in a declarative knowledge base. The management tasks include the declarative classification of an IU, such as naming, description and field information, and the wrapper related information, such as name, WUID (Wrapper Universal ID) and version.

The **User Model Manager** maintains user models, interacting with users, generating user interfaces, and updating and selecting user services. The manager accesses the user associated database, such as user information, device profiles, and user preferences, and plans and organizes the personal web page.

The **Session Manager** manages individual user sessions from entering to exiting as well as multiple user sessions, and activates user models and personal preference modules.

The **Authentication Manager** is used only to confirm user identity and to return user IDs, so that the session manager profits from the user ID to activate the user model manager and plan user models and services.

The **Preference Manager** examines user preference information from the context database to pack the request and launch the information retrieval process. After retrieval from the mediator and associated wrappers, the manager calls up the content aggregation and adaptation modules to generate results to suit the predefined profiles of the personal mobile device and returns them to the user.

The **Mediator** is responsible for coordinating, looking for proper wrappers, delivering requests, and collecting and merging responses from the wrappers. The Mediator first checks the cache for desired information. If the cache-miss, it looks for a proper wrapper via **Wrapper Selector**, launches the wrapper using associated WUIDs and delivers user requests to the wrapper. Finally the Mediator collects and merges multiple responses into unified structured formats, such as XML.

The **Wrapper Selector** is devoted to looking for proper wrappers to suit the user requests passed from the Preference Manager. It then queries the Taxonomy Manager to retrieve the WUID of the wrapper, and passes it back to the Mediator.

The **Caching and Filtering** module maintains and filters personally retrieved content to increase efficiency.

The **Context DB** is a main database that keeps all kinds of information such as User Information, Device Profile, User Preference Information, Information Unit Taxonomic Information, Wrapper Metadata, Personalized Cache, and System Services.

5. System prototyping and evaluation

Amor is a modularized and distributed implementation using java technologies. It offers the administrator an AWG to generate AWs for loosely tied backend information units. The generated AW can be a Java Bean or a Web Service. The current Amor prototype represents a cost-effective approach to developing large-scale meta portals. This approach increases flexibility and scalability by allowing the new Amor components to be replaced dynamically, and by easily distributing AWs over networked servers. Mobile users can browse desired information using any XHTML-enabled cell phone browser.

5.1 Operation scenarios.

Fig. 8 shows an example of browsing the meta portal. Most of the contents nowadays are retrieved from web sites in traditional Chinese script, such as Yahoo Taiwan, Yam, TVBS, Google, Sina, and the announcement and e-mail box of our campus. The technologies and results would be the same as for those retrieving content from English web sites, except for the encoding of data. The first
The step in the operation is to login and configure the homepage of the personal portal, as shown in Fig. 8(a). The mobile device can be used to reconfigure user preference at will. Fig. 8(b) shows the personal homepage of the meta portal. Fig. 8(c) displays the search results from the meta search engines because the user can query more than one search engine simultaneously. Amor aggregates and filters the results from the search engines. It is a query-based service. Fig. 8(d) lists stock information associated with a certain stock number that is a personal setting. Fig. 8(e) displays items of social news from two news web sites. Similarly, Amor aggregates and filters the responses. Then, Fig. 8(f) shows one social news item in detail. That is a link-based service. Finally, Fig. 8(g) shows the content of the student e-mail account for our campus. This is a login-based service.

5.2 Evaluation

There are three commonly used methods for assessing the performance of the system for correctness, efficiency, and cost. By the first, we assess the correctness of the AWs generated by AWGs, the results for which are shown in Table 1. Recall and Precision are the two factors for assessing correctness in information retrieval [8]. Here, \( E_c \) is the number of correct data items extracted by the wrappers, \( N_t \) is the number of correct data items extracted in a request, and \( E_t \) the number of data items extracted by the wrappers. Recall is defined as \( \frac{E_c}{N_t} \) and Precision is \( \frac{E_c}{E_t} \). The results shows that average Recall and Precision are all over 99%, thus indicating that the AWs accurately extracts most of the items from the 11 IUs.

The second, efficiency assessment, measures the time consumed by each of the components associated with the whole systemic performance. The results in Table 2 show that the execution time of all components is less than one second, which is a reasonable and acceptable overhead. The major overhead is extraction from the information unit.

<table>
<thead>
<tr>
<th>Yahoo SE</th>
<th>Sina SE</th>
<th>Open SE</th>
<th>Yam SE</th>
<th>Msn SE</th>
<th>Google SE</th>
<th>UDN News</th>
<th>CT News</th>
<th>TVBS News</th>
<th>Yahoo Stock</th>
<th>Yahoo Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nt</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>Ec</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>59</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>Et</td>
<td>62</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>61</td>
<td>60</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>Recall (%)</td>
<td>96.77</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>98.36</td>
<td>98.33</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Precision (%)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>98.33</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>
Table 2. Components’ performance evaluation (ms)

<table>
<thead>
<tr>
<th>Session Manager</th>
<th>Preference Manager</th>
<th>Mediator</th>
<th>Context DB</th>
<th>Caching &amp; filtering</th>
<th>Wrapper Selector</th>
<th>Taxonomy Manager</th>
<th>Information unit extraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>search</td>
<td>388</td>
<td>282</td>
<td>565</td>
<td>521</td>
<td>723</td>
<td>242</td>
<td>242</td>
</tr>
<tr>
<td>news</td>
<td>417</td>
<td>310</td>
<td>508</td>
<td>437</td>
<td>672</td>
<td>213</td>
<td>430</td>
</tr>
<tr>
<td>stock</td>
<td>312</td>
<td>195</td>
<td>417</td>
<td>440</td>
<td>329</td>
<td>280</td>
<td>309</td>
</tr>
<tr>
<td>weather</td>
<td>203</td>
<td>148</td>
<td>442</td>
<td>365</td>
<td>378</td>
<td>199</td>
<td>220</td>
</tr>
</tbody>
</table>

The third method evaluates the navigation costs, comparing the meta portal with the general one. Here, we measure the size rather than the cost of the downloaded packet because size can be translated into cost when using a telecommunication system, such as the GPRS or 3G, or the bandwidth in Wireless LAN, such as WiFi or WiMax. The results in Table 3 show that the packet size is far smaller using the meta portal rather than the general one because the AW extracts the desired information only and not the whole web page. The evaluation makes clear that the downloaded packet size can be reduced up to 30 to 60 times, or on average 45 times.

Table 3. Network traffic (Bytes)

<table>
<thead>
<tr>
<th>Search Engine</th>
<th>News</th>
<th>Stock</th>
<th>Weather</th>
<th>e-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amor</td>
<td>5540</td>
<td>7296</td>
<td>7559</td>
<td>8321</td>
</tr>
<tr>
<td>Internet Portal</td>
<td>216583</td>
<td>443597</td>
<td>372709</td>
<td>256590</td>
</tr>
</tbody>
</table>

6. Conclusions

Amor is a mobile, reconfigurable, and personalized meta portal based on adaptive wrappers. It ties loosely multiple portals. Mobile handheld devices with XHTML-enabled browsers, such as certain types of cell phones, PDA, and PDA cell-phones, can easily retrieve user-desired information from many web portals or web sites and dynamically reconfigure user preference. The information can be from query-based, link-based, or login-based services. Amor allows mobile users to browse many web sites, to set preferences and to accurately and efficiently retrieve cost-effective, desired content.

We intend at a future stage to improve AWGs to generate AWs for user-defined IUs. Mobile users are able to define the granularity of IUs as required and this in turn helps provide cost-effective information even more precisely and efficiently.

Reference:

11. Panayiotou C., Samaras G., mPERSONA: personalized portals for the wireless user: An agent approach, Mobile Networks and Applications 9, 6 (December 2004), 663-677.