A Unit of Information–Based Content Adaptation Method for Improving Web Content Accessibility in the Mobile Internet

Stephen J.H. Yang, Jia Zhang, Rick C.S. Chen, and Norman W.Y. Shao

In the mobile Internet, users generally work with handheld devices with limited computing power and small screens. Their access conditions also change frequently. In this paper, we present a novel method supporting intelligent content adaptation to better suit handheld devices. The underpinning is a unit of information (UOI)–based content adaptation method, which automatically detects semantic relationships among the components of Web contents and then reorganizes page layout to fit handheld devices based on identified UOIs. Experimental results demonstrate that our method enables more sensitive content adaptation.

Keywords: Mobile Internet, content adaptation, Web content, handheld devices.

I. Introduction

In the mobile Internet environment, users often work with handheld devices, such as personal digital assistants (PDAs) and mobile phones, which provide good mobility but limited computational capabilities and display sizes [1]. Since most existing Web content was originally designed for display on desktop computers, direct content delivery without layout adjustment and content adaptation often leads to disorganization of information on handheld screens. Moreover, not every handheld device can play all media types. For example, a non-multimedia mobile phone cannot play continuous video clips. Also, users' access conditions change more frequently in a mobile Internet environment than in a desktop-based Internet environment [2], [3].

Content adaptation refers to a technique of dynamically adjusting content presentation to meet the constraints of different receiving devices for better presentation [3]. The conventional approach to providing Web content to support various types of receiving devices is to prepare the same content in different formats. This approach is straightforward, but it is error-prone and results in tremendous overhead. To support a new device, all previous Web pages have to support a new format. Even worse, any changes in Web content may require consequent changes on every involved format. Obviously, this is neither practical nor feasible for providers of large volumes of Web content.

However, a simple content adaptation solution of changing a multi-column layout to a single-column layout for display on small handheld screens also introduces severe problems.

Manuscript received: Mar. 22, 2007; revised Aug. 25, 2007.

Stephen J.H. Yang (phone: + 886 3 4227151 35308, email: jhyang@csie.ncu.edu.tw) and Rick C.S. Chen (email: chungshiuan@csie.ncu.edu.tw) are with the Department of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan.

Jia Zhang (email: jiazhang@cs.niu.edu) is with the Department of Computer Science, Northern Illinois University, DeKalb, USA.

Norman W.Y. Shao (email: snorman@giga.net.tw) is with the Naval Shipbuilding Development Center, Kaohsiung,Taiwan.

Without retaining semantic coherence and relationships among semantic units, this primitive adaptation may disorganize a Web page and lead to misunderstanding. Tools and mechanisms are urgently needed to provide users opportunities to experience transparent and seamless Web access using either desktop computers or handheld derives.

An item of Web content is typically composed of multimedia objects (such as text, images, audio, and video), which are connected by various relationships. For example, an image can illustrate a section of a text article; a text title can abstract a text article or some images. In other words, these related objects are synergistically integrated to help readers understand what authors intend to express. Improper rearrangement of these objects and their relationships may lead to ambiguous expression or loss of information. Therefore, it is important for a content adaptation mechanism to maintain the original semantic relationship among the objects during an adaptation process.

In this paper, we present a novel method supporting dynamic unit of information (UOI)–based content adaptation for handheld devices. Our goal is to improve Web content accessibility in the mobile Internet, while retaining the semantic coherence of the original content. To achieve this goal, we introduce UOI as an atomic presentation unit of a Web page; all media objects in a UOI have to be presented as a whole. Our algorithm automatically identifies and detects UOIs from Web pages. Experiments show that our UOI detection algorithm can successfully identify 78% of UOI segments in our test bed. Our method also performs well for well-formatted Web pages.

The remainder of this paper is organized as follows. We first introduce background and related work in section II. We present our UOI-based content adaptation method in section III. We present our experimental designs and result analyses in section IV, and finally, we draw conclusions in section V.

II. Background and Related Work

1. Demands for Content Adaptation

Content adaptation techniques are a response to the wide demand to improve Web accessibility in mobile computing environments. In addition to conventional desktop PCs and laptops, advanced computer technologies have empowered various handheld computing devices, such as ultra-mobile PCs (UMPCs), personal digital assistants (PDAs), Pocket PCs, and smart phones.

Mobile users often encounter various presentation problems (such as cut layouts and oversized pictures) when they surf the Internet. Although handheld devices provide good mobility, they generally have lower computational power, smaller display screens, and slower network speed. Direct content delivery without layout adjustment often leads to the disorganization of information previously mentioned. It also requires users to constantly move scroll bars vertically and horizontally before they can perceive a complete piece of information.

Since not every handheld device can play all media types, content providers have to detect the receiving devices, and the original contents may have to be adjusted to ensure proper retrieval on the receiving devices. One solution is to perform transcoding. Media are transformed into lower quality to become playable on the corresponding devices. For example, video clips can be transcoded into static images to be presented on a non-multimedia phone.

Under some circumstances, it may be unnecessary to deliver all rich media information. For example, if a user is driving, it is unnecessary to deliver video clips because drivers are not supposed to watch video for safety reasons. It should be noted that detecting users' environments and providing only necessary content may save significant bandwidth, which is an important factor in the mobile Internet.

Users' access conditions change more frequently in a mobile Internet environment than in a desktop-based Internet environment. For example, assume a user uses a mobile phone to participate in a two-hour multimedia-based meeting. During the first hour, the user is driving, so she can only listen to audio conversations; in the second hour, she is sitting in another conference room so she can only browse video clips occasionally. Thus, for the first hour, only audio information needs to be delivered; for the second hour, only video information needs to be delivered. Consequently, providing mobile Internet with personalized and adaptive content delivery according to the user's environment could offer more user friendly content provisioning and additionally save significant bandwidth.

Content adaptation can also benefit people with disabilities, such as deafness and blindness. People who are deaf have the same service requirements as people who are sitting in a meeting because they cannot listen to audio. People who are blind have the same service requirements as people who are driving because they cannot read content. Those who suffer from weak vision have the same service requirements as people who are in a blurred environment (due to sunny or gloomy weather), so the content should be enlarged or the background color should be brightened. People who suffer from weak hearing have the same service requirements as people who are in a noisy environment (in a marketplace), so the content volume should be turned louder.

2. Related Content Adaptation Methods

Some researchers have focused on content decomposition methods. Chen and others [4] proposed a block-based content decomposition method, DRESS, to quantify content representation. An HTML page is factorized into blocks, and each block is assigned a score denoting its significance. Then, DRESS selects the block with the highest score to represent the content. This method prevents the loss of significant information. It also enables content layout to be adjustable according to the region of interest, attention value, and minimum perceptible size [5]. Ramaswamy and others [6] proposed an efficient fragment generation and caching method based on the detection of three features: shared behavior, lifetime, and personalization characteristic. The smallest adjustable element in these two approaches is a composite of objects (such as text, image, audio, and video). This granularity of decomposition is too large for mobile device screens; therefore, they are not suitable for mobile content adaptation.

Our previous studies in content adaptation [7], [8] focus on multi-column to single-column layout transformation. We have proved that this method can provide a better browsing experience for mobile devices. However, we found that some semantic errors appear when adjacent media objects crosscut, and these errors may confuse users. To overcome this deficiency, we introduce the concept of UOI and present an algorithm to automatically identify semantically coherent presentation units of components that have to be shown together.

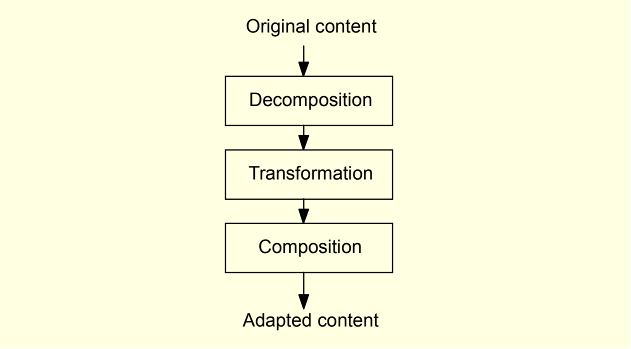


Fig. 1. Three phases of content adaptation.

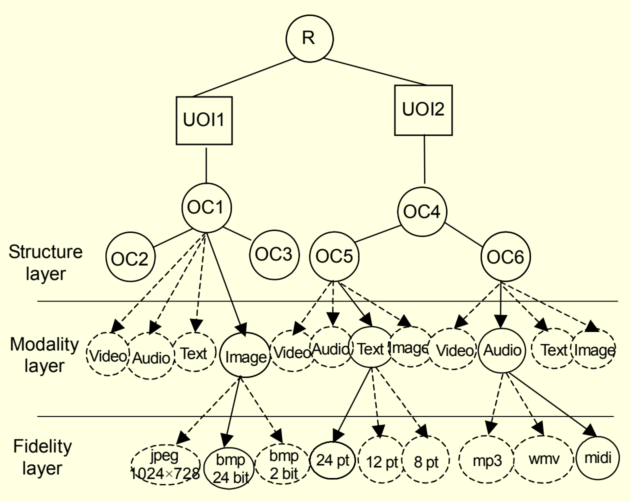


Fig. 2. Content structure model.

III. UOI-Based Content Adaptation

As illustrated in Fig. 1, our content adaptation comprises three main phases: decomposition, transformation, and composition. In the decomposition phase, the original Web page is structurally parsed into components based on a predefined content model [7]. Both the layout and constituent elements (text, image, audio, and video) are extracted separately in this phase. In the transformation phase, transcoding approaches are used to change the fidelity and/or modality of the extracted components for better representation on target devices. In the composition phase, the presentation styles (layouts) and the adapted components are reorganized and recomposed into the final contents to be delivered to the end users.

1. Content Structure Model

A Web page typically contains a set of media objects carrying encapsulated meanings. The semantics among presentation components have to be maintained to deliver correct information. For example, an illustrative figure should be shown close to its detailed text message. When some content is adapted to be displayed on different devices, the semantics of the decomposed portions of the adapted content should remain the same as in the original content. In other words, adapted objects should be grouped on the basis of semantic consistency. As a result, determining the object grouping is the most critical step.

We formalize this object grouping requirement into an isomorphism problem. The relationships among objects and formed groups before and after adaptation should be able to be expressed by an isomorphic graph. To solve this problem, we utilize a layered content structure model [7] to organize objects with possible presentation versions of a given Web page. As shown in Fig. 2, a content structure model maintains available adaptation rules and possibilities for individual presentation objects. According to the content structure model, Web content is organized in three layers, namely, a structure layer, a modality layer, and a fidelity layer. The structural layer comprises the objects contained in the content. The modality layer comprises possible presentation types for each object. The fidelity layer further specifies possible presentation formats for each presentation type. For example, object OC6, shown in Fig. 2, may be presented in four presentation types: video, audio, text, and image. Its audio presentation type can be provided in three formats: mp3, wmv, and midi. If the end user is using an mp3 player while driving, OC6 should be provided in audio using an mp3 format.

We then extend the content structure model by incorporating object relations into its structure layer. The goal is to maintain semantic inherences among objects in the layout re-arrangement to enable more sensitive content adaptation under various circumstances and contexts.

2. Unit of Information and Segment Tree

As shown in Fig. 2, we define an atomic information unit, or unit of information (UOI), as a semantic unit comprising a set of segments and media objects that have to be presented together on the same screen. In our research, the UOI is considered the basic presentation unit of Web content. In the content structure model, the composition of UOIs is expressed in the structure layer. The UOIs have to be identified in the decomposition phase. The subsequent transformation and composition phases have to retain the UOIs unbroken.

A UOI contains two types of elements: segments and object clusters. To design Web page content in a markup language (such as HTML), authors typically use various partition elements (HTML tags, such as <frameset>, <table>, and <div>) to arrange the layout of information objects. These partition elements contain no substantial information; rather, they include layout arrangements and containing relationships. Each of these partition elements is called a segment. Thus, a Web page can be decomposed into a set of segments organized in a hierarchical structure, as shown in Fig. 3. This structure is called a segment tree.

Segments can be further classified into two types: arranging segments (ASs) and containing segments (CSs). Figure 4 is a segment tree which illustrates the concepts and relationships between AS, CS, and OC. The purpose of constructing a segment tree is to detect UOIs in a Web page. An AS refers to a partition element which contains no concrete media objects as direct children. It is used to define the layout of a specific portion of a Web page. In contrast, a CS refers to a partition element which contains at least one concrete media object as a child.

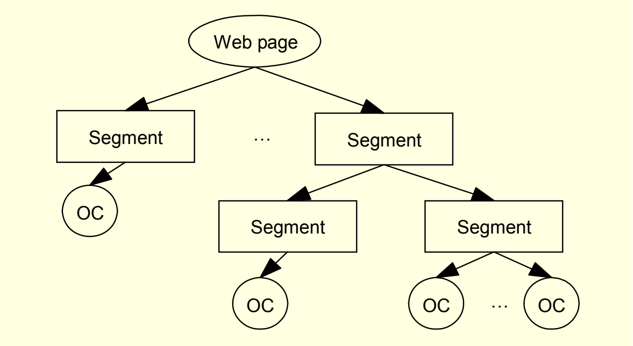


Fig. 3. Relationship among Web page, segments, and object clusters (OCs).

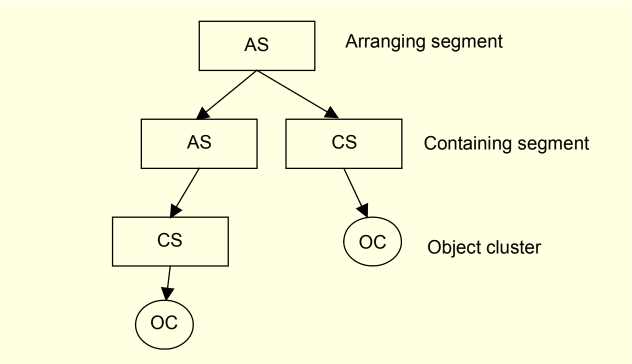


Fig. 4. Segment tree containing arranging segment (AS), containing segment (CS), and object cluster (OC).

All media objects of a Web page are further classified into different object clusters based on their types. Without losing generality, in this research, we consider four types: text, image, audio, and video. After a parsing process, the presentation components are identified as objects associated with presentation attributes. The objects with the same attributes (that is, modality) may have the same semantic hierarchies. An object cluster is thus defined as a collection of media objects that possess the same modality inside of the same containing segment (CS). Six types of object clusters are identified:

- text cluster (TC) (text objects)
- still image cluster (SIC) (jpg, bmp, tiff, and gif objects)
- video cluster (VC) (avi, wmv, and mpg objects)
- dynamic image cluster (DIC) (png and gif objects)
- Flash cluster (FC) (swf objects)
- audio cluster (AC) (mp3 and wav objects)

3. Algorithm to Construct a Segment Tree

Figure 5 shows the pseudo-code of the segment-tree construction algorithm. The procedure includes HTML clean up, tag parsing, object cluster annotation, and segment annotation.

HTML provides great flexibility to integrate a variety of multimedia types; however, the fact that it allows free style writing makes it hard to identify and determine various types of objects in an HTML document. To overcome this problem, our first step is to transform the content into a well-formed format using the open-source package “Tidy” [9]. Then, the well-formed HTML page is parsed into a tree-like structure, in which each node represents a tag in the page.

In theory, any XML parser could be used to parse the HTML content. The generated segment tree structure is traversed to search for object clusters. We use file extensions to identify the

```
page = cleanup(page);
tree = parsePage(page);

//Annotate object clusters (OC)
public void markOC(Node n) {
   if (n.children != null) {
         for (int i = 0; i <= n.children.length(); i++)
            markOC(n.children[i]);
   }
   if ((n.isTC() || n.isSIC() || n.isVC() || n.isDIC() || n.isFC() || n.isAC())
         n.type = "OC";

}

//Annotate arranging segment (AS) & containing segment (CS)
public void markASCS(Node n) {
   if (n.children != null) {
      for (int i = 0; i <= n.children.length(); i++)
         markASCS(n.children[i]);
   }
   if ((n.type!="OC")&&(n.numOfOCChildren()>=1))
      n.type = "CS";
   else n.type = "AS";
}
```

Fig. 5. Algorithm to construct a segment tree.

six types of object clusters. Take the following tag as an example:

<img src=http://news/peace.jpg width="50">

The tag node is considered as a still image cluster (SIC) due to its file extension "jpg." In general, any tree traversal algorithm is applicable here. We adopted a recursive post-order traversal algorithm, where each node is visited after all of its child nodes are visited.

After all object clusters are annotated, we traverse the segment tree once again to identify containing segments and arranging segments. Take the following example:

```
<li id=" 82"> <a href= 27>
<img src="http://news/peace.jpg" width="" height="21">
   Holiday wreath sparks controversy
</a></li>
```

Recall that the "img" segment has been annotated as an object cluster. Its enclosing segment "a href" contains an object cluster; therefore, it is marked as a containing segment. Since the outmost segment "li" only has one containing segment as a direct child, it is marked as an arranging segment.

The result of this construction algorithm for a Yahoo Web page (Fig. 6(a)) is a segment tree, as shown in Fig. 6(b), in which each node is annotated as one of three categories, OC, CS, or AS. The annotation numbers of decomposed segments shown in Fig. 6(b) are marked in Fig. 6(a). For example, one top-level segment (#1) represents the tool bar including the Yahoo logo. The next step is to identify and detect UOIs in the segment tree.

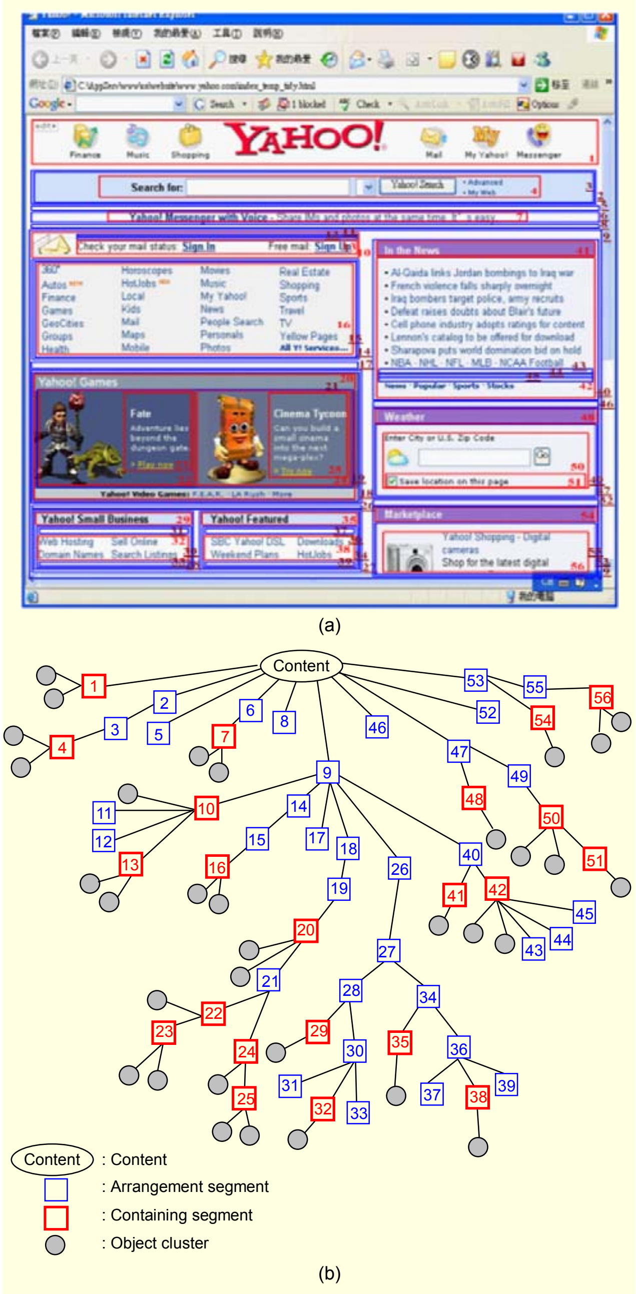


Fig. 6. (a) Original Yahoo Web page and (b) its constructed segment tree.

4. Identification and Detection of UOIs

Figure 7 shows our UOI detection algorithm which is designed on the basis of a segment tree. Figure 8 illustrates the detailed rules for annotating and merging various segment nodes to identify UOIs. Our algorithm goes through a two-phase process: the first phase traverses the initial segment tree and annotates an initial set of UOIs (step 1); the second phase traverses the resulting segment tree from phase 1 to further identify all possible UOIs (steps 2 to 4).

In step 1, the initial segment tree is recursively traversed in post-order to identify all UOIs. As shown in Fig. 8(1), a segment node is annotated as a UOI if it meets all three conditions: its type is AS, it has been annotated with a color attribute, and it contains at least two OC children.

In step 2, UOI candidates and groups are identified in a segment tree. As shown in Fig. 8(2.1), a segment is marked as a UOI candidate if it meets two conditions: the segment type is CS and the segment contains at least two OC children. As shown in Fig. 8(2.2), a segment is marked as a group if it meets two conditions: the segment type is CS, and the segment contains only one OC child.

Step 3 deduces more UOIs by merging UOI candidates and groups in the resulting segment tree in four ways. In step 3.1, as shown in Fig. 8, if a group contains a UOI candidate as a child, it merges with its UOI candidate child to form a new UOI candidate. In step 3.2, as shown in Fig. 8, if a group contains no UOI candidate children but has an adjacent UOI candidate sibling, it merges with the UOI candidate sibling to form a new UOI candidate. If the newly formed UOI candidate has no siblings, it is further merged with its parent to form a new UOI candidate. This process may be recursively repeated toward the root of the tree. In step 3.3, as shown in Fig. 8, if a group has neither UOI candidate children nor siblings but has an adjacent group sibling, it merges with its adjacent group sibling to form a new UOI candidate. If the newly formed UOI candidate has no siblings, it is further merged with its parent to form another UOI candidate. Again this process may be recursively repeated toward the root of the tree. In step 3.4, as shown in Fig. 8, if a group does not have any child or sibling, it is merged with its parent to form a new group, and the process goes back to step 3.1.

Finally, step 4 cleans up the resulting segment tree. If no group exists in the segment tree, all UOI candidates are marked as UOIs.

```
//Step 1. Annotate UOIs (post-order)
public void markUOI(Node n) {
  if (n.children != null) {
    for (int i = 0; i <= n.children.length(); i++)
      markUOI(n.children[i]);
  }
  if ((n.type ==”as”) && (n.color != null) && (n.numOfOCChildren() >=2))
    n.type = “uoi”;
}

// Step 2. Identify UOI candidates and groups
public void markUOICandidateGroup(Node n) {
  if (n.type == “uoi”) return;      //uoi already
  if (n.children != null) {
    for (int i = 0; i <= n.children.length(); i++)
      markUOICandidateGroup(n.children[i]);
  }
  if ((n.type == “cs”) && (n.numOfOCChildren() >= 2))
    n.type = “uoic”;
  if ((n.type==“cs”) && (n.numOfOCChildren()==1))
    n.type = “group”;
 }

// Step 3. UOI determination
public int determineUOI (Node n) {
  if (n.children != null)
    for (int i = 0; i <= n.children.length(); i++)
      determineUOI (n.children[i]);

  //3.0 if all children are UOI candidates
  if (n.children != null) {
    boolean flag = true;
    for (int i = 0; i <= n.child.length(); i++)
    if (n.child[i].type != “uoic”) {flag = false; break;}
    if (flag) n.type = “uoic”;
    return 0;
  }

  if (n.type != “group”) return 0;

  //3.1 merge group with UOI candidate child
  if (n.contain_UOIC_child())   n.type = “uoic”;

  //3.2 merge group with adjacent UOI candidate
  if ((!contain_uoic_child()) && (n.has_uoic_sibling()))
    n.type = “uoic”;

  //3.3 merge group with adjacent group
  if ((!contain_uoic_child()) && (!n.has_uoic_sibling())
    && (!n.has_group_sibling())) {
      n.type = “uoic”;
      //assign “uoic” to adjacent group
  }

  //3.4 merge group upward
   if ((n.child == null) && (!n.has_sibling())) {
    n.parent = “group”;
    return “-1”;
  }
}

// Step 4. UOI Remark
public void remarkUOI(Node n) {
  if (!groupExist(n)) {
    if (n.children != null)
      for (int i = 0; i <= n.children.length(); i++)
        markUOICandidateGroup(n.children[i]);
    if (n.type == “uoic”) n.type = “uoi”;
  }
}
```

Fig. 7. UOI detection algorithm.

5. Content Adaptation through UOI-Based Segment Tree

Our proposed algorithm helps to automatically detect all UOIs of a Web page. Through this process, a segment tree is constructed and annotated by UOIs, which can be used to generate the final adapted content (such as HTML format). Figure 9 illustrates the relationships between the original content, the UOI-based segment tree, and the final adapted content.

Figure 9(a) shows the original content designed for PC or notebook, which contains 12 information objects (OC1 to OC12). To browse the same content via a PDA, however, its size is far larger than a PDA screen. As shown in Fig. 9(b), a

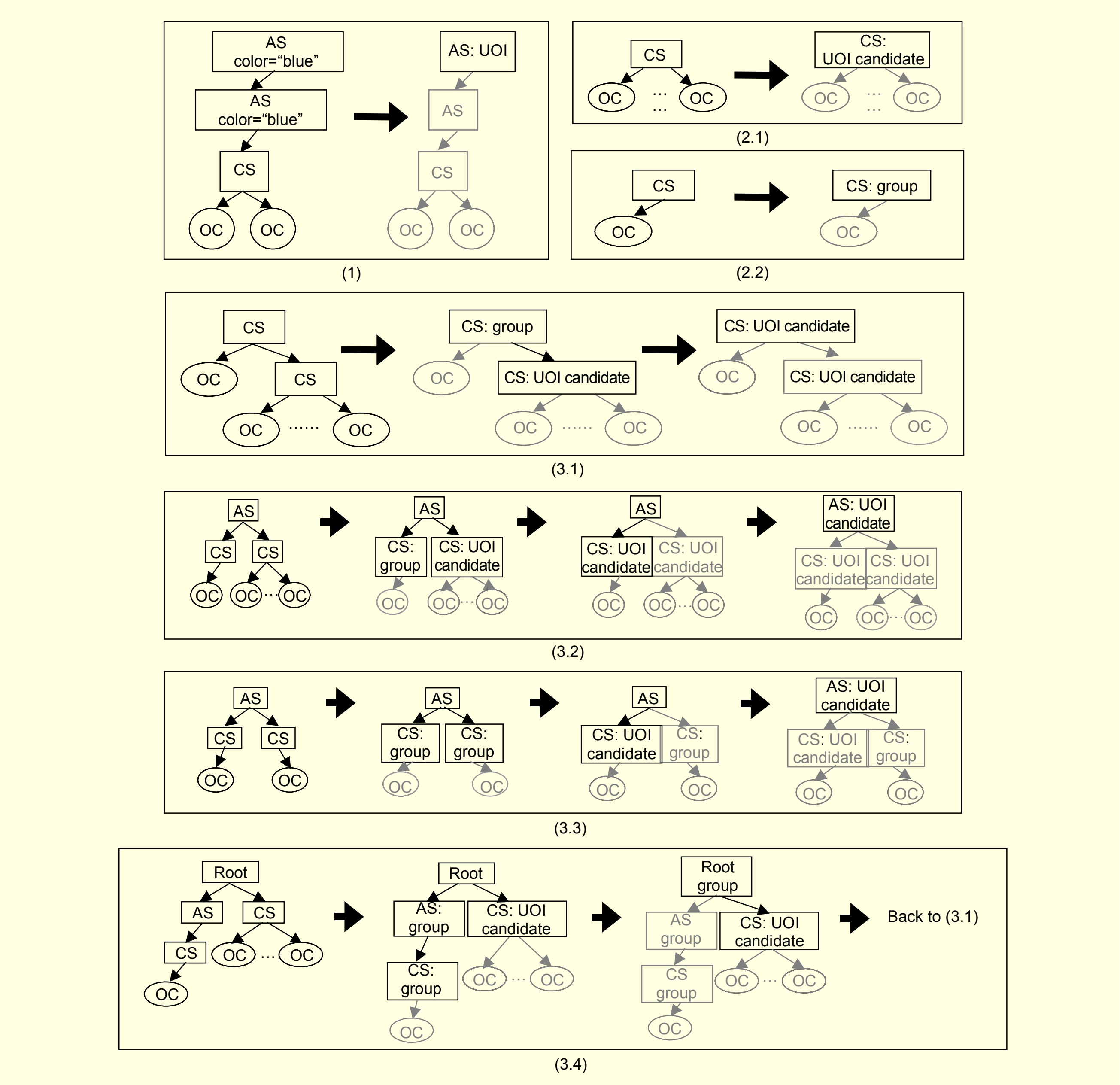


Fig. 8. UOI identification and detection process.

PDA may only present 3 full units (OC3, OC8, and OC10) and part of one unit (OC9). A user has to move the scroll bar vertically and horizontally to view the entire content. Therefore, the original content (in HTML) is transformed into a segment tree, as shown at the top of Fig. 9, by extracting UOIs containing content objects and segments.

After UOI detection and evaluation, the nodes in the segment tree are reorganized under corresponding UOI nodes. As shown in Fig. 9, four UOIs are identified and include all presentation units. For example, UOI1 contains the three units in the upper section of the original design shown in Fig. 9(a), OC1, OC2, and OC3. Then, if each UOI fits on the PDA screen, their locations are rearranged in columns, as in Fig. 9(c). The original multi-column layout is changed into a single-column layout.

If the largest UOI in the content cannot fit onto a small screen, the scales and positions of the objects in the UOI should be further adjusted for a suitable presentation. Note that the objects comprising the same UOI may require some layout relationships (such as parallel and serial). As shown in Fig. 9(c), OC3 should be presented to the right of OC2 (parallel), and OC2 should be presented after OC1 (serial). Layout adaptation

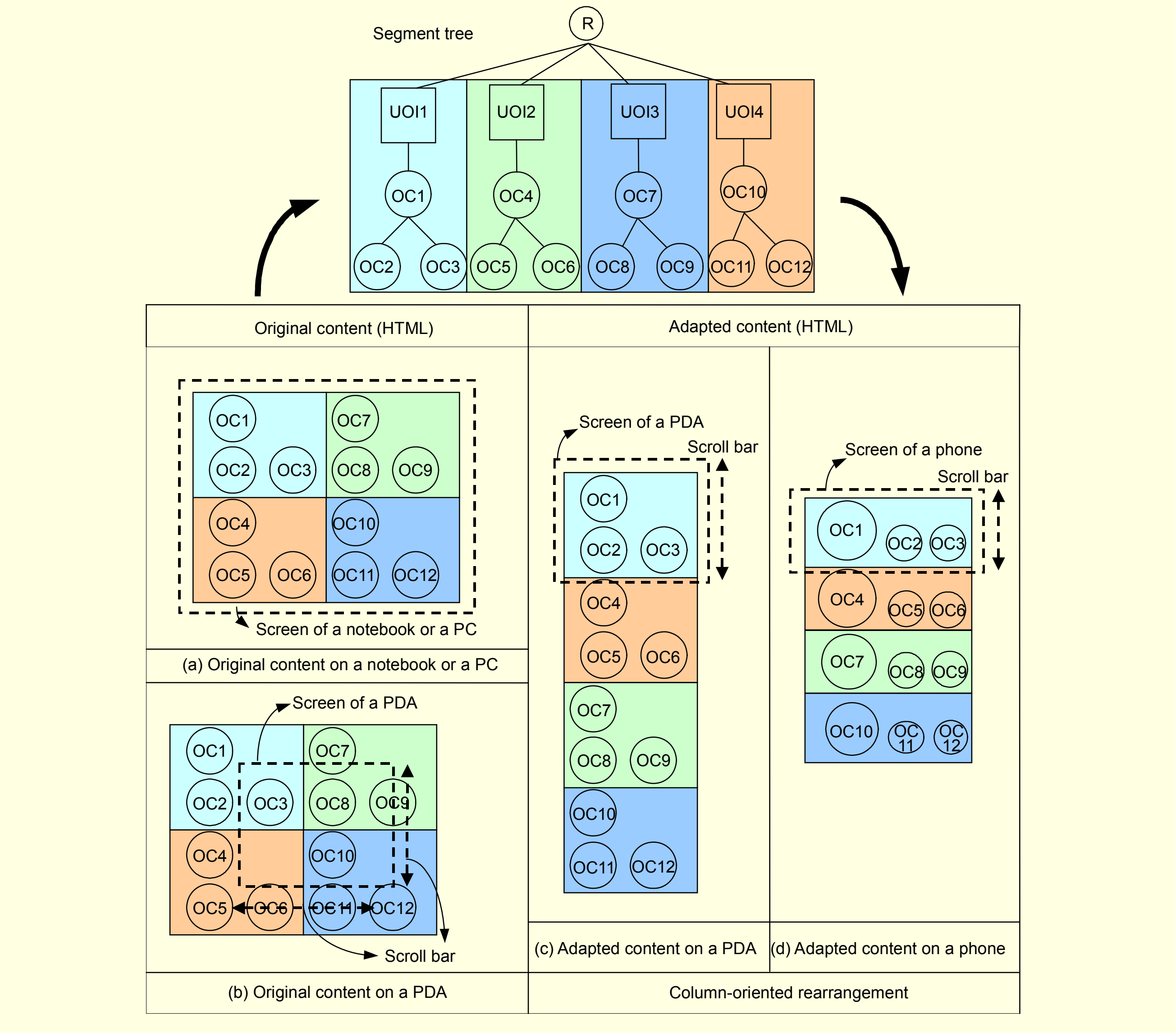

Segment tree
R
UOI1
UOI2
UOI3
UOI4
OC1
OC2
OC3
OC4
OC5
OC6
OC7
OC8
OC9
OC10
OC11
OC12
Original content (HTML)
Adapted content (HTML)
Screen of a notebook or a PC
(a) Original content on a notebook or a PC
Screen of a PDA
Scroll bar
(b) Original content on a PDA
Screen of a phone
(c) Adapted content on a PDA
(d) Adapted content on a phone
Column-oriented rearrangement

Fig. 9. Content adaptation process with UOI-based segment tree.

should maintain these implicit relationships. As shown in Fig. 9(d), to keep an entire UOI on one screen, the layout positions between containing objects are adjusted. Both OC2 and OC3 are changed to be parallel with OC1 with reduced sizes. Thus, their implicit inter-relationships are maintained. This study focuses on UOI-based fragment detection; therefore, the algorithm for adjusting layout positions will not be discussed. For details, see our previous works, [7] and [8].

IV. Experiments and Discussion

To evaluate our proposed UOI detection algorithm for Web page decomposition, we designed and conducted a set of experiments to measure the correctness rate of UOI detection. We conducted experiments for both visual performance and quantitative analysis. The visual performance experiment was designed to evaluate the performance of our UOI detection method on Web page adaptation and transformation. We randomly selected a set of Web sites, some are text-oriented (e.g., Yahoo site) while some are image-oriented (e.g., B'z site). For each Web site, we used a PDA screen to visualize three results: the Web page without any adaptation, the Web page with primitive content adaptation algorithm applied (directly transforming multi-column to single-column layout), and the Web page with UOI detection-based content adaptation applied. The visualization results for each Web site under the three strategies were captured as screen shots for comparison.

The experiment for quantitative analysis was designed to quantitatively examine whether our proposed UOI detection method could successfully identify all UOIs of a Web page. We designed a four-stage validation procedure to measure the correctness rate of UOI detection: Stage 1 builds a target baseline; Stage 2 executes our algorithm; Stage 3 evaluates the result against the baseline; Stage 4 further assesses whether our algorithm could facilitate context-aware service provisioning in a mobile computing environment.

The goal of stage 1 is to establish a comparison baseline for later stages. A Web page is manually browsed to identify all UOIs. The information objects which have the same or similar semantic meanings are grouped into a UOI.

In stage 2, our proposed UOI detection algorithm is executed over the same Web page to identify UOIs. The identification process is automatically executed and monitored.

The goal of stage 3 is to evaluate the correctness rate of UOI detection and identify the misdetermined or lost information objects using our proposed UOI detection algorithm. The results from stage 1 and stage 2 are compared to calculate the correctness rate. The erroneous fragments are scrutinized for future improvement.

The goal of stage 4 is to visually validate whether our UOI detection algorithm could maintain the semantic meanings of the original content. Our method is to adapt the Web page based on UOIs identified in stage 2 and show the results on a PDA screen. Considering the limited size of a PDA screen, the presentation sequence of the page is re-arranged by forming a single-column layout based on identified UOIs. The transformation of the original Web page presentation into a single-column layout is just one simple yet efficient way to evaluate our algorithm. The visual effect is examined to validate whether the original semantic meanings are maintained.

1. Performance Analysis of UOI Detection Method

We constructed a test bed with 35 Web sites grouped in four categories: 5 from academia, 2 from news stations, 11 from business corporations, and 17 from general Web portals. Without losing generality, we randomly selected Web sites as testing samples. For each selected Web site, we performed the designed four-stage validation procedure. The results were monitored and accumulated for analysis.

The comprehensive test results for each of the selected 35 testing samples are summarized in Table 1. For each testing Web site, we measured the results for the following five factors: the number of manually identified UOIs, errors occurring in the decomposition phase, errors occurring in the composition phase, incorrectly identified UOIs, and correctly identified UOIs. The detected errors caused by the UOI detection algorithm in stage 3 are counted and analyzed in the decomposition phase; the detected errors caused by the presentation rearrangement in stage 4 are counted and analyzed in the composition phase. Errors caused by decomposition are further divided into two categories: data loss in the pre-process and errors caused by the UOI detection algorithm. Errors caused by composition are further divided into two categories: errors cause by misarrangement and errors caused by composition.

The data from Table 1 indicates that our proposed UOI detection algorithm successfully detected an average of 78.49% of UOIs from the Web content in the test bed. Comparing the adaptation results from the four categories of Web pages, we found that the academic Web sites have the highest correctness rate of UOI detection (88.06%). The business Web sites have the second highest correctness rate, followed by general portals with a 72.98% correctness rate. News station Web sites have the lowest correctness rate (48.89%).

The data shows that our algorithm has a promising high correctness rate of UOI detection when the Web sites provide well-formatted Web pages. Academic and business Web sites are typically developed by professional Web developers and do not undergo frequent changes, and this may be the reason our UOI detection algorithm shows a constant high correctness rate for Web sites in the two categories. In particular, it showed a 100% correctness rate of UOI detection for 5 business Web sites (about half of the Web sites we tested in this experiment). By examining and monitoring one of the business Web sites (IBM) with a low correctness rate of UOI detection, we found that the Web site undergoes frequent changes. As a result, inconsistent HTML formats are prone to reduce the correctness rate of UOI detection.

Web sites belonging to the category of general portals are developed by people with varying levels of Web development skills. Therefore, our UOI detection algorithm shows high correctness rates for some sites (such as 100% for the Yahoo Web site) and low correctness rates for some other Web sites (such as 27.27% for the Swirve Web site).

It is difficult to successfully detect UOIs on news Web sites. As shown in Table 1, only 24 out of 103 UOIs were identified in the CNN Web site. After careful examination of corresponding Web sites, we found that they include an enormous number of “table” fragments and a variety of multimedia information. This causes our UOI detection algorithm to produce many false segment nodes and leads to incorrect merging in step 3. We also found that news Web sites undergo constant changes, and this leads directly to inconsistent or untidy HTML content here and there. This factor also greatly affects the correctness rate of our UOI detection algorithm.

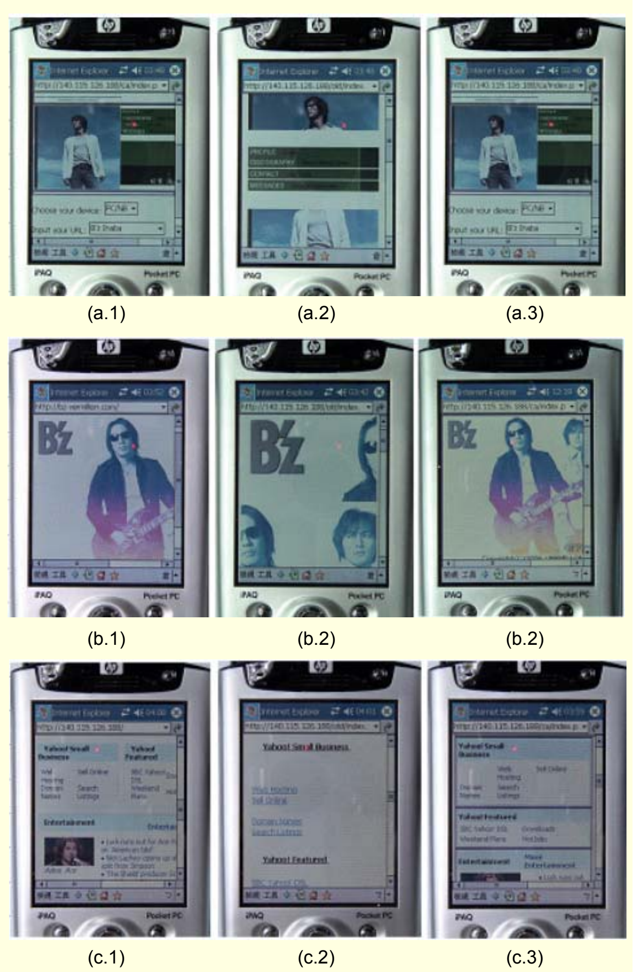
Table 1. Experimental results of the UOI detection from 35 Web sites.

Index	Category	Websites	Detected UOI no.	Decomposition		Composition		Error UOI		Correct UOI	
				Data loss in the pre-process	Errors from stage 3	Errors caused by mis-arrangement	Errors caused by composition	No.	Percentage	No.	Percentage
1	School	Harvard	12		3			3	25.00%	9	75.00%
2		Berkeley	16		2			2	12.50%	14	87.50%
3		MIT	4					0	0.00%	4	100.00%
4		NCU	9		2			2	22.22%	7	77.78%
5		NKFUST	3					0	0.00%	3	100.00%
6	News	BBC	47		2	7	3	12	25.53%	35	74.47%
7		CNN	103		72	1	6	79	76.70%	24	23.30%
8	Business	Scientific American	32		9			9	28.13%	23	71.88%
9		Apple	7				1	1	14.29%	6	85.71%
10		ASUS	6	1				1	16.67%	5	83.33%
11		NTT DoCoMo	10					0	0.00%	10	100.00%
12		Socket	10					0	0.00%	10	100.00%
13		B'z	1					0	0.00%	1	100.00%
14		Inaba	3					0	0.00%	3	100.00%
15		CASINO	12					1	8.33%	11	91.67%
16		IBM	13				4	4	30.77%	9	69.23%
17		francetelecom	14					0	0.00%	14	100.00%
18		NotiEmail	6	2				2	33.33%	4	66.67%
19	Portal	Yahoo	19					0	0.00%	19	100.00%
20		rediff	25		4		2	6	24.00%	19	76.00%
21		Love To Know	72		2			2	2.78%	70	97.22%
22		Best Spider	5					0	0.00%	5	100.00%
23		Flavorpill	16	1			2	3	18.75%	13	81.25%
24		Intermix	8	2				2	25.00%	6	75.00%
25		Web 100	3					0	0.00%	3	100.00%
26		Top 10 Links	4	1				1	25.00%	3	75.00%
27		iWon	23		2		1	3	13.04%	20	86.96%
28		My Way	7	4			1	5	71.43%	2	28.57%
29		Sify	28	1	14			15	53.57%	13	46.43%
30		BONZI	4				1	1	25.00%	3	75.00%
31		Swirve	11		8			8	72.73%	3	27.27%
32		Irish Abroad	12	1			4	5	41.67%	7	58.33%
33		Thing Find	23					0	0.00%	23	100.00%
34		DIMUSEUM	11				5	5	45.45%	6	54.55%
35		Elite	22		8	1		9	40.91%	13	59.09%
Total			601	13	128	9	30	181	21.51%	420	**78.49%**

2. Visual Effect Analysis

Figure 10 shows the visualized results on PDAs applying different strategies on three randomly selected Web sites (Inaba, B'z, and Yahoo) either focusing on images or on texts. The experimental results for each Web site occupy one row, which comprises three screen shots: original content, adapted content without UOI detection, and adapted content based on UOI detection. As shown in Fig. 10, content adaptation based on our UOI detection algorithm effectively reorganizes and adjusts the original content on a PDA screen.

Some Web sites emphasize image-oriented content, such as Inaba and B'z as shown in Figs. 10(a.1) and (b.1), respectively. Using the primitive column-wise approach, each object is treated independently. Thus, the adaptation process may scale up one image object to the entire screen size. Moreover, some unrelated information objects may be integrated into one screen, as shown in Figs. 10(a.2) and (b.2). This simple object-based adaptation may cause confusing representation. In contrast, by applying our UOI detection algorithm, the original semantic meanings associated with objects are preserved in the process of content adaptation. As shown in Figs. 10(a.3) and (b.3), original large images are scaled down to fit into the PDA screen, associated with related information.


(a.1) (a.2) (a.3)

(b.1) (b.2) (b.2)

(c.1) (c.2) (c.3)

Fig. 10. Comparison of visualization effect from (a) Inaba, (b) B'z, and (c) Yahoo: (1) without adaptation, (2) primitive column-wise adaptation, and (3) UOI-based adaptation.

Some Web sites emphasize text-oriented content, such as Yahoo as seen in Fig. 10(c.1). Using the primitive column-wise approach, each information object (mostly text-oriented object) is adapted based on delivery context. However, the presentation sequence of the adapted objects may become ambiguous, as shown in Fig. 10(c.2). By using our UOI detection algorithm, the related information objects are grouped as an integral presenting unit as shown in Fig. 10(c.3). The semantic relationships between information objects are preserved.

3. Analysis on Detection Errors

We further analyzed the possible causes leading to UOI identification failures. Based on the statistical information summarized in Table 1, we identified four categories of causes which are shown in Fig. 11.

Information loss in the decomposition phase results in 4.97% of the total number of errors. Errors which occur in step 3 of UOI detection algorithm contribute to 70.72% of the total number of errors. Information loss in the composition phase leads to 17.13% of the total number of errors; and misarrangement in the composition phase results in 7.18% of the total number of errors.

We utilize HTML Tidy [9] to transform the original Web pages to well-formatted contents before generating a segment tree, but due to the free-writing style of HTML, some information objects may be lost during the transformation process. For example, Fig. 12(a) shows two information objects, one Flash object marked by F(1) and an image object marked by I(2). Through visual analysis, they should form one

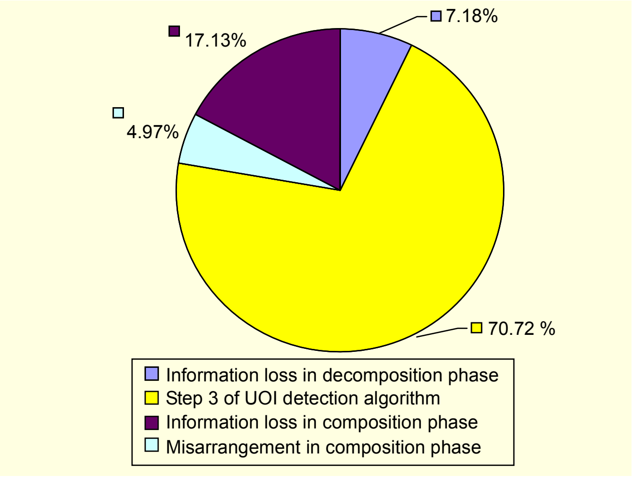


Fig. 11. Distribution of detection error causes.

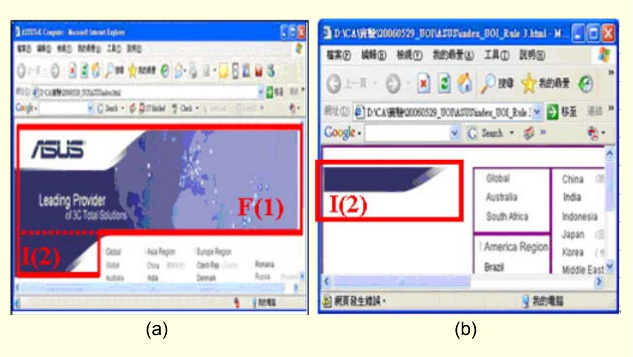


Fig. 12. (a) Correct UOI detection through a manual process and (b) erroneous UOI detection due to information loss in and the decomposition phase.

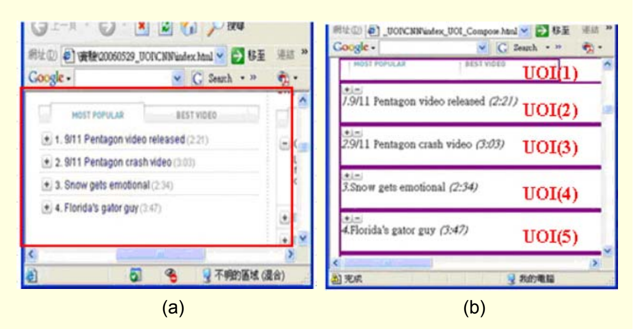


Fig. 13. (a) Correct UOI detection through a manual process and (b) erroneous UOI detection caused by step 3 in the UOI detection algorithm.

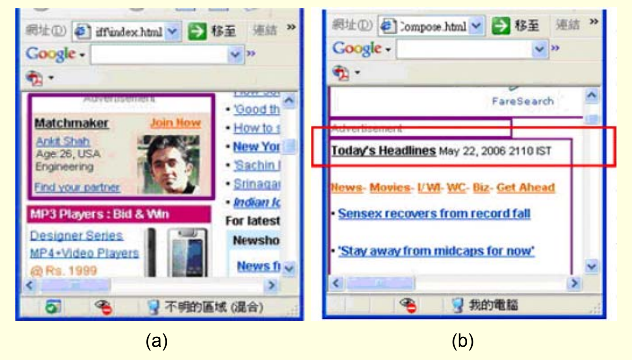


Fig. 14. (a) Correct UOI detection through a manual process and (b) erroneous UOI detection due to information loss in the composition phase.

UOI enclosed by a red box. Nevertheless, after the Tidy transformation process, only the image object is left as shown in Fig. 12(b). The Flash object F(1) is lost, which results in a detection error.

In step 3 in our UOI detection algorithm, a standalone group should be merged with its adjacent segment nodes. However, if corresponding HTML scripts do not follow the exact formats, the merge attempt may fail. Figure 13 shows such a UOI detection error due to an inadequate merge. In Fig. 13(a), a manually identified UOI contains a set of adjacent segment nodes. In Fig. 13(b), the detection results of step 3 split the UOI into five UOIs, which is apparently a UOI detection error.

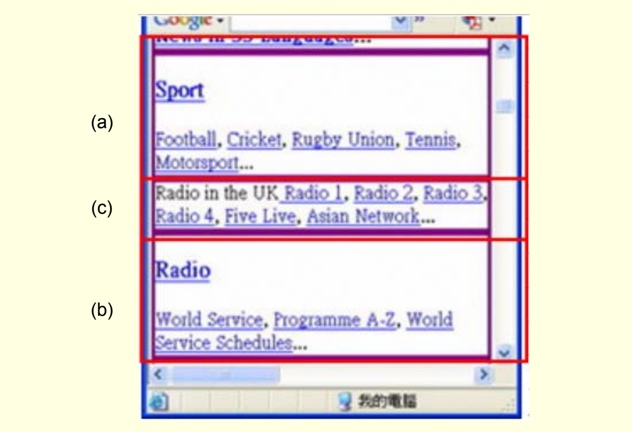

(a)
(c)
(b)
Sport
Football, Cricket, Rugby Union, Tennis, Motorsport...
Radio in the UK Radio 1, Radio 2, Radio 3, Radio 4, Five Live, Asian Network...
Radio
World Service, Programme A-Z, World Service Schedules...

Fig. 15. Erroneous UOI detection due to miss-arrangement in the composition phase.

In the composition phase, some information may be lost due to the limited screen size of wireless devices. To transform a multi-column layout into a single-column one, composition rules may lead to errors. In the composition process, if comprising UOIs involve content adaptation, sometimes information in some identified UOIs may be lost. For example, in Fig. 14(a), the UOI enclosed by a purple box consists of several information objects. However, after adaptation and re-composition, some information was lost as shown in Fig. 14(b).

In the composition phase, misarrangement may also lead to errors, if the relationships among the UOIs are incorrectly interpreted. For example, in Fig. 15, the presenting sequence of UOIs should be shown as (a), (b), and then (c). However, after composition, this presentation sequence is incorrectly arranged as (a) and (c) followed by (b).

4. Further Discussions

Our experiments show that our UOI detection algorithm works well with well-formatted HTML Web pages. Regarding ill-formatted Web pages, it seems that more cleanup work is necessary in addition to that which can be done by the Tidy package we have adopted. We are working on this topic in our current research work.

Based on our working experiences in the software industry, the reason many existing HTML pages are irregular and ill-formatted is not because their creators intentionally confuse the HTML expression of the pages in order to make difficulties for algorithms such as the one proposed in this paper. Rather, it is mainly due to two reasons. First, many Web page designs involve non-technical developers (such as visual designers), who are not familiar with HTML or any programming languages. They typically exploit some HTML page design tools such as DreamWeaver. While these tools provide a fantastic "what you see is what you get" feature, their generated HTML code is not well-formatted, let alone after multiple rounds of editing (add and remove). Second, an ongoing Web page, such as a CNN page, typically goes through many versions, which may involve different developers with various coding habits and preferences. Even worse, it is a common practice in Web page development that a new page is built by modifying on top of an existing page, especially when the time constraints are tight. For example, CNN has to generate many pages on a daily basis. As a result, it is impractical to require page designers (or content generation systems that build the pages) to always provide regulated formats. It is obviously challenging for researchers to explore approaches to fill the gap. While there are still many challenging issues remaining unsolved, our reported work in this paper establishes a technical foundation and framework for building such a content adaptation engine to automatically transform existing HTML pages into appropriate formats to be shown on mobile devices.

Even with well-formatted Web pages, there is still room to further improve content analysis. In the real world, it is common for a Web page to undergo multiple changes involving different developers. It is possible that some implicit semantic relationships and dependencies exist in a Web page. To adapt such a Web page according to its original semantic coherence, it is necessary to analyze and identify these implicit semantic segments, and this is our ongoing research topic.

We also found several shortcomings in our current adaptation techniques, such as the lack of capability to process script languages (such as JavaScript and VBScript), and the lack of a session and message processing mechanism (such as login session). We are planning to address these shortcomings in our future research.

We found one particularly interesting phenomenon. Many Web pages have similar layout structures, even though their contents are significantly different. These similar but rarely changed portions, such as header fragments and navigation fragments, occupy significant storage space and consume many computing resources. To speed up the decomposition process and reduce required storage space, we plan to continue to work on an intelligent fragment detection method to examine similar fragments among Web pages.

Moreover, we realize that our content adaptation algorithm consumes some processing time. Based on our current test bed, the delay is acceptable (within 11 seconds on average). However, we plan to design a dedicated set of experiments to systematically measure and evaluate the impact of the content adaptation process on the performance of Web browsing of various types of content pages. We plan to identify extreme situations in which the impact is too significant to ensure a reasonable response time in a mobile device. We also plan to examine and compare the processing delay caused by individual steps of our content adaptation to improve performance.

V. Conclusion

In this paper, we presented a UOI-based dynamic content adaptation approach. We presented algorithms that automatically detect semantic relationships among components in a Web page and then reorganize page layout to suit handheld devices based on identified UOIs. Our experiments demonstrated that our UOI detection algorithm effectively preserves semantic meanings and the coherence of information objects in a Web page and can greatly facilitate the adaptation of Web pages to mobile devices, and that it works especially well with well-formatted Web pages.

We are continuing our research in several directions. First, we are investigating how to detect and elicit semantic segments from original HTML content by identifying implicit semantic dependencies and relationships in addition to HTML tag relationships. Also, we are exploring how to add the capability of processing script languages to our content adaptation mechanism. We are also examining page layout patterns to improve the performance of content decomposition process. We are designing test cases to examine the overall performance impact, as well as the impact of the composing steps our content adaptation algorithm on Web browsing. Finally, we are studying Web 2.0 technology to further enhance our content adaptation technique.

References

[1] M.K. Kim and K.Y. Jee, "Characteristics of Individuals Influencing Adoption Intentions for Portable Internet Service," *ETRI Journal*, vol. 28, no. 1, Feb. 2006, pp. 67-76.

[2] M. Roman, N. Islam, and S. Shoaib, "A Wireless Web for Creating and Sharing Personal Content through Handsets," *IEEE Pervasive Computing*, Jan.-Mar., 2005, vol. 4, no. 2, pp. 67-73.

[3] A. Pashtan, S. Kollipara, and M. Pearce, "Adapting Content for Wireless Web Service," *IEEE Internet Computing*, 2003, vol. 7, no. 5, pp. 79-85.

[4] L.Q. Chen, X. Xie, W.Y. Ma, H.J. Zhang, H.Q. Zhou, and H.Q. Feng, *DRESS A Slicing Tree Based Web Representation for Various Display Sizes*, Technical Report MSR-TR-2002-126, Microsoft Research, 2002.

[5] L.Q. Chen, X. Xie, X. Fan, W.Y. Ma, H.J. Zhang, and H.Q. Zhou, *A Visual Attention Model for Adapting Images on Small Displays*, Technical Report MSR-TR-2002-125, Microsoft Research, 2002.

[6] L. Ramaswamy, A. Iyengar, L. Liu, and F. Douglis, "Automatic Fragment Detection in Dynamic Web Pages and Its Impact on Caching," *IEEE Trans. Knowledge and Data Engineering*, vol. 17, no. 6, 2005, pp. 859-874.

[7] S.J.H. Yang and N.W.Y. Shao, "An Ontology Based Content Model for Intelligent Web Content Access Services," *Int'l Journal of Web Service Research (JWSR)*, vol. 3, no. 2, Apr.-June 2006, pp. 59-78.

[8] S.J.H. Yang and N.W.Y. Shao, "Enhancing Pervasive Web Accessibility with Rule-Based Adaptation Strategy," *Expert System with Applications*, vol. 32, no. 4, May 2007, pp. 1154-1167.

[9] HTML Tidy Library Project, http://tidy.sourceforge.net/.

Stephen J.H. Yang received his PhD degree in computer science from the University of Illinois at Chicago in 1995. He is now a professor of the Department of Computer Science & Information Engineering and the Associate Dean of Academic Affairs, National Central University, Taiwan. He is the co-founder and the CEO of T5 Corp, a company providing XML-based Web services. Dr. Yang has published 2 books and over 140 journal articles and conference papers. He is currently on the advisory board of the Advances in Web Services Research Book Series, IGI Global. He is a member of the editorial boards of the International Journal of Web Services Research and the International Journal of Knowledge and Learning. He severed as the Program Co-Chair of IEEE MSE 2003, IEEE CAUL 2006, and IEEE W2ME 2007. His research interests include Web services, Web 2.0, software engineering, knowledge engineering, semantic Web, and context aware ubiquitous computing. He is a member of IEEE.

Jia Zhang received her PhD in computer science from University of Illinois at Chicago in 2000. She is now an assistant professor of the Department of Computer Science at Northern Illinois University. hang has published 1 book titled "Service Computing" and over 70 refereed journal articles, book chapters, and conference papers. She is an associate editor of the International Journal of Web Services Research (JWSR) and an associate editor of the Advances in Web Services Research (AWSR) Book Series, IGI Global. Zhang serves as Program Vice Chair of IEEE International Conference on Web Services (ICWS 2008 & 2007 & 2006). Her current research interests center around services computing. She is a member of the IEEE and ACM.

Rick C.S. Chen received the BS and MS degrees from the Department of Computer and Communication Engineering, National Kaohsiung First University of Science and Technology in June 2000 and June 2002, respectively. He is currently a PhD student in the Department of Computer Science and Information Engineering, National Central University, Taiwan. His research interests include content adaptation, expert systems, and artificial intelligence.

Norman W.Y. Shao received his PhD degree in Information Engineering from National Kaohsiung First University of Science and Technology, Taiwan in 2006. Dr. Shao is a research engineer with the Naval Shipbuilding Development Center. His research interests include content management system, content adaptation, and personalized knowledge design.