Applying Semantic Segment Detection to Enhance Web Page Presentation on the Mobile Internet[1]

STEPHEN J.H. YANG*, JIA ZHANG**, STELLA T.C. TSAI*** AND JEFF J.S. HUANG***
*,***Dept. of Computer Science & Information Engineering, National Central University, Taiwan
**Dept. of Computer Science, Northern Illinois University, USA

To facilitate wide adoption of handheld devices for mobile Internet access, it is important to enable the "develop once and use everywhere" concept. This means that a Web page can be developed in HTML for presentation on desktop machines, and then be automatically adapted and displayed on any devices including handheld devices. In this paper, we present a mediator Web service that automatically adapts a Web page into suitable formats based on receiving devices. Our underpinning is a hierarchical semantic meta model associated with a semantic ontology. They guide in partitioning a Web page into semantically coherent segments, which refer to atomic units encapsulating semantically coherent elements in a Web page. Our experimental results demonstrate that our semantic segment detection algorithm helps in transforming Web pages into semantic-retained formats suitable to be displayed on handheld devices.
Keywords: Mobile Internet, semantic segment, Web page presentation, page adaptation

1. INTRODUCTION

As people are increasingly using handheld devices (e.g., PDAs and cell phones) to access mobile Internet [1], many content publishing tools appear to provide content adaptation facilities that transform Web pages into proper formats before delivering them to different receiving devices. This is because handheld devices have smaller screens, slower network connections, and less computing power [2]. Typical Web page adaptation examples include Oracle application server wireless [3], Sun Java system portal server mobile access [4], and WebSphere transcoding publisher [5].

However, these publishing tools require that the adaptable Web pages be developed using the same tools and platforms from the beginning. Requiring that all Web pages be developed in certain vendor-specific formats is neither feasible nor even desirable. Meanwhile, many Web pages exist, and may continue to appear, in an HTML format targeting on desktops. Requiring that all these HTML pages be regenerated and reformatted before supporting multiple channels is even more impractical. Thus, how to make these large-screen-oriented HTML pages automatically and transparently adaptable and accessible to mobile users is highly necessary but challenging. Since many Web pages are generally HTML documents on the Internet, the terms Web page and HTML document are used interchangeably throughout this paper.

In contrast with existing Web page adaptation techniques that focus on transforming

[1] This paper is a significant extension to our conference version presented in the IEEE International Conference on Services Computing (SCC 2008), Jul. 8-11, 2008, Honolulu, HI, USA, pp. 210-217.

*Corresponding author. Tel.: +886 3 422 7151x57102 e-mail: jhyang@csie.ncu.edu.tw (Stephen J.H. Yang).

stored raw data (typically in an XML format) into various formats according to the types of receiving devices [6, 7], our approach is analog to the concept of reverse engineering that starts from parsing an existing HTML page, re-generates the original page design knowledge, and then transforms it into appropriate formats based on receiving devices.

This paper reports our continuous efforts that explore an automatic semantic segment detection method for HTML documents. Our key contribution is a practical semantic segment identification procedure supported by a hierarchical semantic model. The procedure comprises four consecutive steps:

(1) one that deals with the coding page problem at HTML's text level;

(2) one that parses an HTML document into an annotated structure at HTML's tag level and results in an HTML tag tree;

(3) one that constructs a formal structural model at HTML's structure level and results in an HTML structural tree based on structural meanings and visual effects through original structure and style-sheet analysis, and

(4) one that identifies semantic segments at HTML's semantic level by aggregating units that retain highly related semantic meanings.

The resulted semantic segment tree for the entire HTML document can be used in the re-composition process of Web page adaptation, as well as in a variety of other applications such as content caching, content sharing, content annotation, page understanding, and information retrieval.

The remainder of this paper is organized as follows. In Section 2, we review related work. In Section 3, we propose a semantic ontology. In Section 4, we present our semantic segment detection method and discuss how its results can be used to facilitate Web page adaptation. In Section 5, we discuss our experiments and evaluations. In Section 6, we make conclusions.

2. RELATED WORK

Berhe, Brunie, & Pierson [8] present a service-based adaptation framework. An adaptation operator is introduced as an abstraction of various transformation operations. However, the actual implementation is still in a primary phase. How to map constraints to adaptation operators remains unsolved.

Some researchers focus on Web page decomposition methods. Chen et al. [9] propose a block-based Web page decomposition method. Ramaswamy et al. [10] propose a fragment generation method based on detection of three features: shared behavior, lifetime, and personalization characteristic. However, the smallest adjustable element in these two approaches is a composite of presentation objects (e.g., text, image, and audio). Their decomposition granularity is too large for handheld device screens; therefore, they are not suitable for mobile Internet page adaptation.

Yu et al. [11] propose a Vision-based Page Segmentation (VIPS) algorithm to detect semantic structures in an HTML page based on visual cues. In contrast with their work, our approach takes into consideration of Cascading Style Sheet (CSS) [12] that is usually associated with HTML pages to separate presentation layout from Web page.

Chen et al. [13] propose a pattern recognition-based method for displaying a large Web page on a small-screen device. Their page analysis algorithm partitions the page into

smaller and logically related Web page blocks, by analyzing various scaffolds (e.g., header, footer, side bars, and body). In contrast with their work considering semantic meanings of some HTML tags, our method systematically studies semantic meanings of HTML tags and yields a tag-oriented ontology. Our ability of detecting semantic segments outperforms their method (will be shown in experiments).

Chen et al. [14] present a Function-based Object Model (FOM) for website adaptation. In contrast to their approach focusing on identifying functional properties of presentation objects, our approach focuses on identify semantically coherent presentation units.

Baluja [15] casts the Web page segmentation problem into a machine learning framework and applies computer vision techniques (e.g., the lens of entropy reduction and decision tree learning) to identify coherent regions. In contrast to their algorithm based on structural analysis of HTML pages, our approach provides a more comprehensive analysis over HTML pages including both structural analysis and semantic analysis.

Buyukkokten et al. [16] propose an HTTP proxy that automatically fetches Web pages and dynamically generates summary views for clients. In contrast to their work focusing on analyzing hyperlinks between interconnected Web pages, our work focuses on analyzing semantic relationships between presentation objects on the same Web page.

Hori et al. [17] propose embedding annotations into Web documents to provide hints for content adaptation to various receiving devices. Xiao et al. [18] propose to apply the VIPS technique to deliver an HTML page as a converted thumbnail-based view to mobile users. In contrast to their works providing methods for page designers to provide page adaptation recommendations, our work focuses on automatic page adaptation over generic HTML pages.

Kao et al. [19] propose a system to allow mobile users to specify personal preferences so that irrelevant information can be removed. In contrast to their work focusing on finding the relationships between content and user preferences, our research focuses on finding semantic relationships between presentation blocks.

Colak et al. [20] propose a rule-based inference approach to determine a user's knowledge status, which can be used to guide personalized content adaptation. Heraclitus [21] is a framework for Semantic Web adaptation where user needs and requirements are used to support Web site ontology and topology evolution. Chua et al. [22] study how to support shared viewpoints and personal viewpoints among multiple mobile users. In contrast to their works focusing on ensuring the same view between multiple users, our work focusing on adapting HTML-based content to fit in a single mobile screen.

Our previous work reports an approach to analyze HTML tags to detect atomic presentation units [6], which is the foundation to our systematic page segmentation efforts. We also developed algorithms to find associated constraints (rules) and specifications for presentation object scaling, resizing, and adaptation [23, 24]. Compared with our previous works, the work reported in this paper makes significant enhancements in several aspects. First, we conduct a comprehensive examination of HTML tags and establish an ontology to categorize all HTML tags, to enable automatic tag functionality analysis. Second, we take into consideration CSS association, which can be used to help more effective segmentation. Further, CSS information helps appropriate context-aware Web page adaptation. Third, we establish a methodology to systematically conduct incremental page segmentation. Our experiments prove that our enhancement work significantly increases the effectiveness of page segmentation, while being able to cover more types of HTML pages.

3. SEMANTIC ONTOLOGY

To improve Web page presentation on handheld devices, we propose a method that automatically detects semantic segments in HTML documents. Our underpinning is an ontology for categorizing HTML tags into a hierarchy based on their semantic meanings, which does not exist in the current W3C-recommended Web Ontology Language (OWL) [25].

HTML grammar provides a rich set of tags to define Web page presentation structures. We notice that these tags are associated with implicit semantic meanings. In other words, behind HTML syntax is an implicit semantic language. For example, a tag <p> represents a paragraph; , <ul>, and <li> present a list of items in an HTML document. Thus, analyzing HTML tags and structures may help in extracting and eliciting semantic meanings embedded by the original page developers.

We further found that these semantic units have implicit relationships between them. For example, a <BODY> tag can be constructed using objects (<TABLE>) as below:

```
<HTML><HEAD><TITLE>Example page</TITLE>
   <BODY>
      <TABLE width="95%" border=0>       </TABLE>
   </BODY>
</HTML>
```

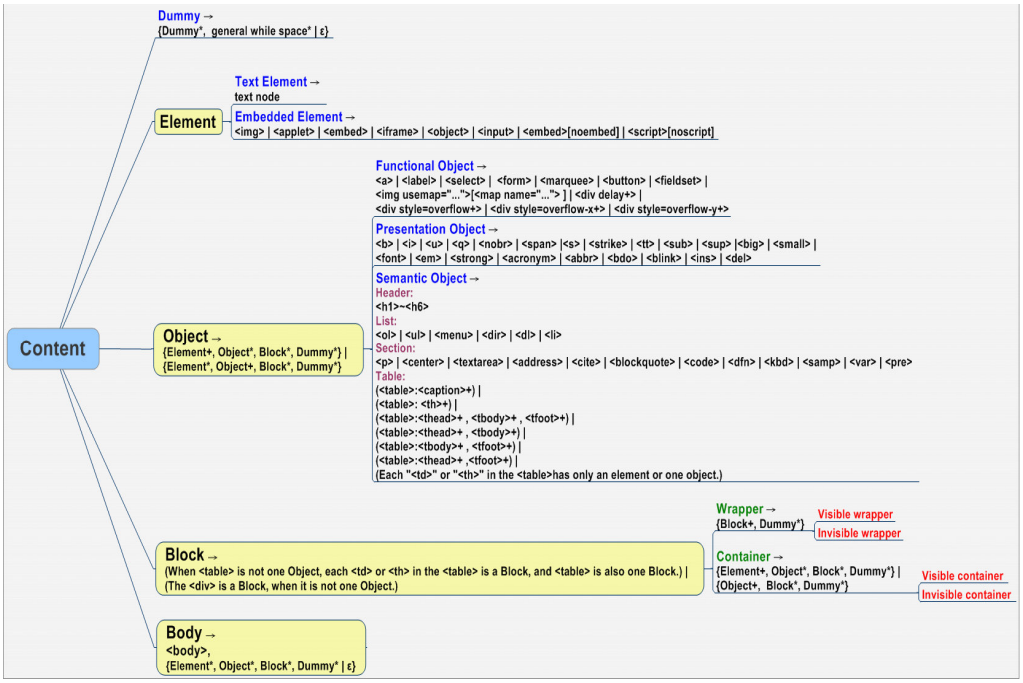

Fig. 1. Ontology for HTML semantics.

Our ontology for HTML semantic units is summarized in Fig. 1. We classify various HTML tags (syntax units) into a set of categories (semantic units), based on their implicit semantic meanings. An HTML document may contain five fundamental classes of semantic units: dummy, element, object, block, and body. Element is defined as a basic seman-

tic member of an HTML document. An element in turn is divided into text element and embedded element, the former referring to textual data and the latter referring to multimedia elements inserted in a webpage. An Object refers to a container encompassing elements or other objects that have the same function, purpose, or presentation style. An object is in turn divided into functional object, presentation object, and semantic object. They refer to sections encapsulating functions (e.g., anchoring), presentation styles, and structural relationships, respectively. Dummy does not represent any actual presentation data in a Web page (e.g., whitespaces). A Block delimits and arranges components in a Web page. Blocks can be further classified into two types: wrapper and container, depending on whether they contain actual display materials. Body of an HTML document comprises the overall information of the document. Finally, content represents the entrance (root) of an entire HTML document.

We use BNF form [26] to formally define these semantic units, so that formal verification mechanisms can be applied. As an example shown below, text element refers to a text node in an HTML document. Composed of characters, a text node may refer to a word, a symbol, a sentence, or a section contained in an HTML tag.

text_element := text_node
text_node := char+ | <html_tag> char+ </html_tag> | <html_tag> char+

4. SEMANTIC SEGMENT DETECTION METHOD

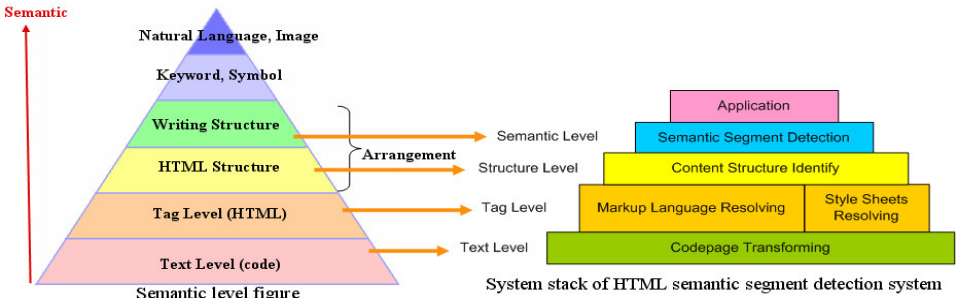

Fig. 2. Semantic segment detection method.

Based on our formalization of HTML semantic units, we establish a semantic model as shown in the left side of Fig. 2. From bottom up, six semantic levels are identified in an HTML page: (1) text level: Page processing is only possible for correct text writings without wording errors. (2) tag level: HTML is a semi-structured programming language, so that its code structure is delimited by a set of predefined HTML tags. (3) HTML structure level: Tags have inherent semantic meanings; so that they can depict the presentation and semantic structure of an HTML page. (4) writing structure level: Semantically related HTML sections may share some common words. For example, an article introducing a scientist typically has his/her name shown in both the body of the article and the caption of his/her picture. (5) keyword and symbol level: A Web page is usually organized by predefined keywords and symbols. For example, the homepage of a company is organized by the various aspects of the company. (6) natural language and image level. Words may expose some semantic intention of the page developers. For example,

synonyms may help in detecting semantic relationships between sections. In this paper, we will focus on the first four semantic levels. Leveraging the top two levels to further enhance semantic segment detection will be our future research topic.

Based on the semantic model, we establish a stepwise procedure to incrementally identify and detect semantic segments in an HTML document. The procedure comprises four steps as shown in the right side of Fig. 2: codepage transforming at HTML's text level, CSS localizing and HTML parsing at HTML's tag level, Web page structure identification at HTML's structure level, and semantic segment detection at HTML's semantic level. We will discuss each step in detail in the following sections.

4.1. Codepage Transformation at HTML's Text Level

Before analyzing a downloaded Web page (HTML document), some preprocessing is necessary, since the characters of the document are typically encoded as a sequence of types following a particular character encoding when transmitted over the Web (HTML was designed to support different coding pages). The encoding may be either in a Unicode Transformation Format (e.g., UTF-8 [27]) that directly encodes unicode characters or in a legacy encoding (e.g., Windows-1252 [28]) that does not. In either way, we have to transform the receiving bytes into a homogeneous format for further analysis and processing. We chose to translate them into the Unicode format for simplicity. Our encoding and conversion procedure is summarized in the following four steps:

Step 1: Detect the character encoding of the downloaded Web page.
Step 2: Transform the Web page into meta-page represented by Unicode.
Step 3: Perform page adaptation based on obtained Unicode.
Step 4: Transform the adapted meta-page back into the original "character encoding" page.

4.2. CSS Localizing and Parsing at HTML's Tag Level

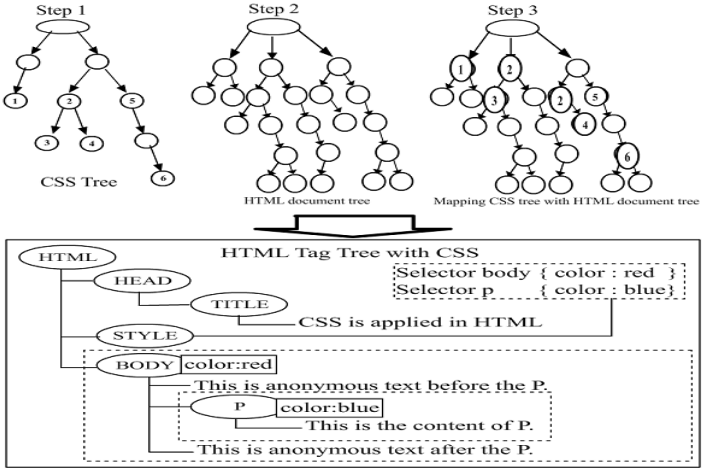

Fig. 3. CSS localization at HTML's tag level (Transformation from a CSS tree and an HTML document tree to a CSS annotated HTML tag tree)

HTML documents allow to specify presentation features in two ways, either inside of the corresponding HTML document as tag attributes (e.g.,"<table bgcolor=black>") or

via separate CSS documents [29]. CSS provides selectors (tags) to describe presentation features with patterns for HTML tags. Although CSS specifications could be embedded inside of an HTML document as a simplified version, many HTML pages adopt the way of using separate CSS files. Therefore, to properly understand the structure and presentation semantics of an HTML document, its corresponding CSS definitions shall be extracted for an examination.

Based on our previous Web page content decomposition algorithm [6], we have developed a CSS-enabled page parsing procedure. As shown in Fig. 3. The inputs of this transformation are a CSS tree and an HTML document tree; the output is a CSS-annotated HTML tag tree. CSS specification and inheritance information are automatically located and extracted at the HTML tag level. The style specification for every HTML tag is extracted from the associated CSS documents and attached to the tag. Note that the attachment is a plug-in relationship; so that the CSS information can be dynamically updated or removed from the HTML document tree. CSS localization can be summarized in three steps as follows:

Step 1: Search the input HTML text and locate CSS and inheritance information, then parse the CSS document and build a CSS tree.
Step 2: Parse the HTML document and build an HTML document tree.
Step 3: Integrate the above two outputs in a CSS-annotated HTML tag tree.

4.3. Structural Segment Detection at HTML's Structure Level

We then adopt a depth-first search strategy to traverse the annotated structure tree and detect structural segments. The algorithm is as follows. If a tag is not <body>, we check whether it is a dummy, an element, or an object. If it is none of these, the system moves to its next sibling tag. If the tag contains visible display information or if it is a container, the tag is considered a structure fragment. Otherwise, the system moves to the next tag. If a tag is <body>, we check whether it is a dummy. If it is, the system moves to the next sibling tag. Otherwise, we check whether it is an element or an object. If it is, the tag is a structure fragment.

As shown in Fig. 4, our algorithm transforms an HTML tag tree on the left into an HTML structure tree on the right. Note that our algorithm makes judgment based on the traditional HTML structure patterns defined in HTML specifications (e.g., tags for arrangement and vision style). Domain criteria may be needed to support heterogeneous domain-specific features (e.g., local linguistic habits). It is out of the scope of this paper and will not be discussed here.

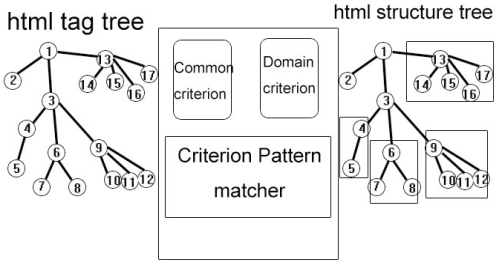

Fig. 4. Web page structure identification at HTML's semantic level

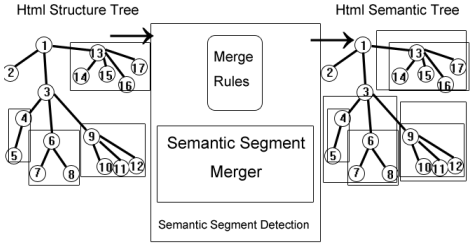

Fig. 5. Semantic segment detection at HTML's structure level.

Dummy structural unit requires attention. As defined below, Dummy does not represent any actual presentation data in a Web page. Therefore, dummy units can be removed to facilitate later semantic segment detection.

dummy := dummy* | white space* | ε

A pseudo-code algorithm of identifying and determining a dummy in an HTML document using such a definition is shown as below.

Input: An HTML node.
Requirement: Determine if the node is dummy.
Algorithm:
1. **for** each child node **do**
   (a) Determine if the child node is an element.
   Case 1: the child node is an element, Output result false.
   Case 2: the child node is not an element, determine if the child node is an object.
   Case 2(a): the child node is an object, Output result false.
   Case 2(b): the child node is not an object, recursively check if the child node is dummy
   Case 2(b)(1): the child node is dummy, continue.
   Case 2(b)(2): the child node is not dummy, Output result false.
2. If not yet output, Output result true.

4.4. Semantic Segment Detection at HTML's Semantic Level

Afterwards, we build a semantic segment merger to decide which parts (sub-trees) of the structure tree possess self-contained semantic meanings, as shown in Fig. 5. A segment is considered having semantic coherence if it exhibits one or more of the following four properties, say, to keep complete function (functionality related), readable typesetting (readability related), relationship of presenting (space and time related), and literary context (semantics related).

The merger function contains three major steps. In Step 1, it identifies parts of a structure tree that may become candidates for semantic segments, that is, semantic fragments. This step is manually conducted, based on visual effects of corresponding HTML representation. The process continues until all semantic fragments are identified. The next two steps aim to refine the manual identification. The key challenge is how to merge semantic fragments and add structural units, objects, elements, and blocks into a semantic segment. We identified three guidelines for determining a semantic segment: (1) it must retain the integrity of functionality; (2), it must retain the readability of article and vision; and (3), its comprising objects must retain temporal and spatial stratum.

In Step 2, we construct an algorithm, based on our semantic ontology model, to transform a refined structure tree into a segment tree. As shown by the following pseudo code, the algorithm identifies three types of structural objects: object clusters (presentation objects), arranging segments (presentation tags), and containing segments (grouping tags). While traversing the structure tree, if a node of a sub-tree is an element, it is marked as an object cluster. If a node is a presentation object, it is marked as an arranging segment. If a node is a functional object or a semantic object, it is marked as a containing segment.

Input: A web page node.
Requirement: Transform into a segment tree.
Algorithm:
1. Construct a refined structure tree.

```
    (a) Annotate semantic units (Dummy|Element|Object|Block|Page).
    (b) Parse the page.
    (c) Construct a structure tree.
2.  Annotate the node as object clusters (OC).
    (a) Check the node.
        Case 1: the node has child nodes.
          for each child node do
            recursively annotate the child node.
    (b) if the node is an element, annotate the node type as "OC".
3.  Annotate the node as arranging segment (AS) & containing segment (CS)
    (a) Check the node.
        Case 1: the node has child nodes.
          for each child node do
            recursively annotate the child node.
    (b) Check the code of the node.
        Case 1: the node is an object.
          Case 1(a): the node is PresentationObject, node type is "AS".
          Case 1(b): code type is "CS" (including FunctionalObject or SemanticObject)
```

In Step 3, we detect semantic segments. Our previous UOI detection algorithm [6] studies how to merge two UOI candidates. Based on our semantic ontology model, we refine our algorithm [6] to automatically identify semantic segments. The input of the algorithm is the constructed segment tree from Step 2; the output will be identified semantic segments as shown in Fig. 5 on the right-hand side. Our refinement is a two-phase process. In phase 1, the segment tree is traversed and all UOI candidates are annotated. In phase 2, UOI candidates incrementally and iteratively merge with its UOI candidate children, UOI candidate siblings, and UOI candidate parents to form UOI groups. The process continues until all semantic fragments are determined.

4.5. Device-Aware Web Page Adaptation

The outcomes of the aforementioned four-step procedure are stored as an annotated segment tree, as shown at the top of Fig. 6. The concept of Web page adaptation is to allow each UOI to occupy one screen and make all UOIs to be displayed serially. In other words, the original multi-column layout will be adapted into a UOI-based single-column layout. The example in Fig. 6 shows that a segment tree of an HTML page comprises four UOIs encapsulating all presentation units. Assuming that each UOI fits on a PDA screen, the four UOIs are displayed onto the PDA screen sequentially, as shown in the lower left part of Fig. 6 in a single-column layout.

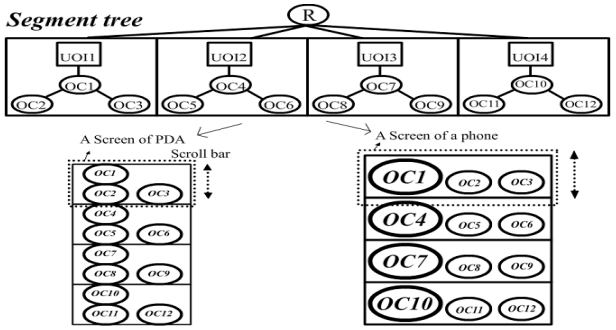

Fig. 6. UOI-based Web page adaptation.

If a large UOI in the Web page cannot fit into a small-size screen (e.g., a cell phone as shown on the lower right of Fig. 6), the scales and positions of the comprising 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. 6, OC3 should be presented to the right of OC2 (parallel), and OC2 should be presented after OC1 (serial). Layout adaptation should maintain these implicit relationships.

Our algorithm is to adapt a vertical layout into a horizontal layout, while emphasizing the first presentation objects in a group. As shown in Fig. 6, 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, while OC1 has been highlighted at a significant position (e.g., occupying 40% of the horizontal space of a screen). Thus, their implicit inter-relationships are maintained.

5. EXPERIMENTS AND DISCUSSIONS

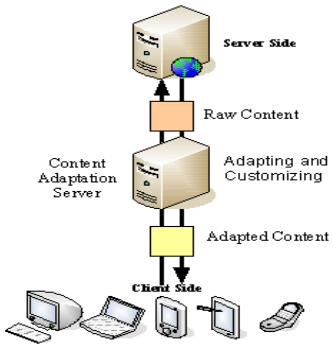

Fig. 7. Experimental set up

Fig. 8. Test results on Web page adaptation.

We implemented an HTML semantic segment detection service as a prototype to test our method. Then we designed and conducted a series of experiments to evaluate the performance of our semantic segment detection algorithms (through detection accuracy rate) and the mediator service (through execution time). The former aims to test the effectiveness of our algorithms of correctly identifying semantic segments; the latter aims to test the overhead of the algorithms.

Figure 7 shows our experimental set up. Our content adaptation service was deployed on an Apache Web server, on an IBM server machine equipped with an Intel 3.4GHz CPU and 1.24GB RAM. All implementations run on Red HAT Linux 7.3 servers. Our tested receiving devices include a notebook (NB) with MS Windows XP installed, a Windows Mobile 5.0 Pocket PC Emulator, and a smart phone 2003 SE. All the tests between testing devices and our mediator service go through a wireless network or a wired network. The connection between the service and the tested websites are through a wired network.

Retrieval operations are conducted by using and without using our content adaptation service. If our content adaptation service is turned off, our semantic segment detection process will be bypassed. For comparison, our baseline approach is the original Web

page viewed in a client-side handheld browser adapted by the browser itself; our comparing object is the resulted page adapted by our proposed server-side adaptation mechanism. As shown in Fig. 8, on the left side for PDA and on the right side for cell phone, Web page delivery using our approach shows significantly better results: original multi-column page content is adapted into single-column display to better suit smaller screens.

We built our test bed by randomly selecting 50 popular Web sites from a combination of five categories: academia, business, entertainment, news, and general-purpose portals. From the 50 selected Web sites, we randomly selected 200 Web pages. Then we performed both quantitative and qualitative evaluations over the test bed.

5.1. Quantitative Evaluation

5.1.1. Precision

We designed experiments to evaluate whether our method can successfully detect all semantic segments of a Web page, that is, the effectiveness of our method. For each of the 200 Web pages, we first used an NB as a receiving device to download the page. Second, we manually identified all semantic segments of the retrieved Web page to establish a target baseline (DetectedSemanticSegmentNumbers). Third, we used a PDA as a receiving device to download the same page, while using our mediator service in the middle, as a test sample case. For each manually identified semantic segment, we verified whether it is detected in the test sample case. If it is, then the number of correctly identified semantic segments (CorrectNumbers) is increased by 1. After checking all semantic segments, we calculated the correct rate using the following formulae:

$$CorrectRate(\%) = \left(\frac{CorrectNumbers}{DetectedSemanticSegmentNumbers} \right) \times 100\%$$

For each of the 200 Web pages, we ran our experiment and recorded a correct rate. For each Web site, we accumulated the correct rate of its individual pages and calculated an average to represent the correct rate of the site.

Table 1. Test results on correct rate of semantic segment detection

Category	Web Site	Correct Rate(%)	Category	Web Site	Correct Rate(%)
Academia 87.48%	Harvard	83.3%	News station 81.6%	Speedyclick	85.7%
	Berkeley	100%		CNN	92.1%
	MIT	100%		Google new	62.26%
	NCU	66.6%		MSN Weather	80%
	Nkfust	87.5%		BBC	92.3%
Business 86.04%	Microsoft	75%	General-purpose portal 91.2%	Yahoo	100%
	Ebay	75%		Yahoo Taiwan	100%
	Earthlink	100%		Msn	91.2%
	Bizrate	75%		Hotmail	100%
	Amazon	98.38%		Google	100%
	Iwin	80%		AltaVista	92.85%
	Espn	75%		MyPoints	100%
	Apple	100%		Go	81.8%
	ASUS Taiwan	75%		Aol	75%
	NTT	75%		Americangreetings	87.5%
	B'z	100%		Lycos	85%
	Inaba	90%		Infospace	90%
	IBM	100%		Looksmart	100%
	Notiemail	86.3%		Angelfire	83.33%
Entertainment 92.4%	Windowsmedia	100%		Search Web Services	80%
	Flowgo	84.61%		Sciam	93.7%
	Bluemountain	90.9%		Bestspider	100%
	Ivillage	100%		Flavorpill	100%
	Netscape	85.71%		Intermix	100%
	Mapquest	100%		Swirve	65%

Table 2. Test results on execution overhead of semantic segment detection.

Category	Web Site	Second(s)	Category	Web Site	Second(s)
Academia	Harvard	0.265625	News station	Speedyclick	0.25
	Berkeley	0.21875		CNN	0.390625
	MIT	0.234375		Google new	0.453125
	NCU	0.28125		MSN Weather	0.28125
	Nkfust	0.3125		BBC	0.25
Business	Microsoft	0.296875	General-purpose portal	Yahoo	0.5625
	Ebay	0.296875		Yahoo Taiwan	0.65625
	Earthlink	0.234375		Msn	0.28125
	Bizrate	0.265625		Hotmail	0.265625
	Amazon	0.546875		Google	0.25
	Iwin	0.265625		AltaVista	0.265625
	Espn	0.34375		MyPoints	0.296875
	Apple	0.265625		Go	0.3125
	ASUS Taiwan	0.234375		Aol	0.328125
	NTT	0.265625		Americangreetings	0.296875
	B'z	0.21875		Lycos	0.25
	Inaba	0.296875		Infospace	0.265625
	IBM	0.265625		Looksmart	0.296875
	Notiemail	0.296875		Angelfire	0.265625
Entertainment	Windowsmedia	0.25		Search Web Services	0.265625
	Flowgo	0.21875		Sciam	0.328125
	Bluemountain	0.28125		Bestspider	0.28125
	Ivillage	0.3125		Flavorpill	0.3125
	Netscape	0.40625		Intermix	0.296875
	Mapquest	0.3125		Swirve	0.28125

Table 1 shows the list of our 50 tested Web sites and their corresponding correct rate, indicating the correct percentage of detecting the semantic segments in the site using our

method. The numbers show a total of 87.744% in average over our test bed. The results also show that our method performs well in most of the Web site analysis, no matter whether a site belongs to the category of academia (87.48%), business (86.04%), entertainment (92.4%), news (81.6%), or general-purpose portals (91.2%).

5.1.2. System Execution Time Evaluation

In our experiments, we examined the execution time of our content adaptation service to investigate the overhead of our algorithms. Our service includes four consecutive procedures at four HTML's levels: text level, tag level, structure level, and semantic level. We embedded code in our service to calculate the execution time of individual levels. Specifically, we define service execution time on a specific Web page as the amount of time spent on executing the tasks at the four levels, respectively: parsing HTML page, localizing CSS, detecting page structure, and saving semantic pages. For each of the 200 Web pages, we use an NB to retrieve the page and record the system execution time in seconds. For each Web site, we accumulate the system execution time of its individual pages and calculate the average to represent the system execution time of the site.

Table 2 shows the list of our 50 tested Web sites and their corresponding system execution time. The average system execution time of the entire test bed is 0.3034375 second. We then calculated the individual average execution time of the four levels. For example, the execution time of HTML page parsing is defined as the average execution time of the corresponding step over the entire test bed. As shown in Fig. 9, the HTML code parsing and page transformation tasks each occupies about 35% of the total system execution time; the page structure detection step occupies about 20%; the CSS localization task occupies about 10%. Our results imply that further efforts may be needed on decreasing overheads on HTML code parsing and page transformation algorithms.

We also found that the time distribution over the four levels may vary, depending on the design and writing styles of specific Web pages. For example, the site yahoo.com.tw contains comprehensive CSS information, while CNN.com carries little CSS information. As a result, as shown in Fig. 10, the former site (a) spends significant time on CSS localization (37%) but less time on Web page structure detection (10%); the latter (b) spends little time on CSS localization (4%) but significant time on Web page structure detection (20%).

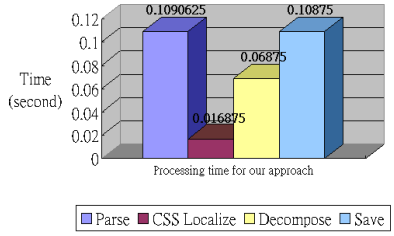

Fig. 9. Execution overhead of the four levels.

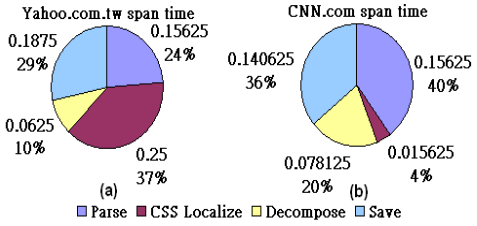

Fig. 10. Time distribution on four levels for two Web sites: (a) yahoo.com.tw, (b) CNN.com.

5.1.3. Comparison with Related Methods

We designed experiments to compare the effectiveness of our semantic segment detection method with two other related approaches: View Small Device [13] and UOI [6].

The former was chosen because their approach takes into consideration of semantic meanings of some HTML tags; the latter was our previous work on the step 4 of our procedure. We implemented these two algorithms as two individual services. In the system, by switching our content adaptation service to these two services, we can repeat the experiments in the same contexts.

For each of the 200 Web pages, we conducted the same precision test procedure using three methods. In contrast with our 87.744% correct rate, the View Small Device [13] system achieved a 55% correct rate, and UOI achieved a 78% rate. The reason why our method exceeds the View Small Device approach is that, our method systematically studies semantic meanings of HTML tags and yields a tag-oriented ontology. The reason why our method exceeds UOI method alone is that, our first three steps help prepare a segment tree that facilitates semantic segment detection using the UOI algorithm.

We also found that both View Small Device and UOI had some difficulties with some types of Web pages. For example, View Small Device system does not perform well on MSN, Google, Yahoo, Earthlink, Speedyclick, Netscape, Angelfire, CNN, and Windowsmedia; UOI does not perform well on CNN, BBC, IBM, NotiEmail, Intermix, and Swirve. Our method does not have difficulty understanding these Web pages.

5.2. Qualitative Evaluation and Survey

For each Web page in our test bed, we retrieved it on both a desktop browser and a PDA screen, using our content adaptation service. The displays on both screens are saved in a pair as a test case. In Spring 2007, we employed a class of 47 freshmen students as volunteers to verify the entire package of 200 test cases. A tutorial was given to the entire class of students prior to the experiment. For each test case, a student volunteer was asked to compare the original desktop screen shot and the adapted PDA screen shot, and rate the adaptation performance in the following three levels: satisfied, tolerable, and unacceptable. For each level, the student volunteer was asked to justify his/her rating in detail. In this way, we ensure that their ratings make more sense.

Satisfied: This level implies that a student considers an outcome of page adaptation is satisfactory. Students are asked to give further ratings from the perspectives of page analysis, semantic segment detection, and component re-composition.

Tolerable: This level implies that a student considers the outcome of the Web page adaptation is acceptable; however, there exist minor errors that do not affect the student's overall understanding of the page. The students are also asked to clearly list the errors they detect.

Unacceptable: This level implies that a student considers the adapted display is far from the original Web page and is confusing. The students are also asked to specify the erroneous places.

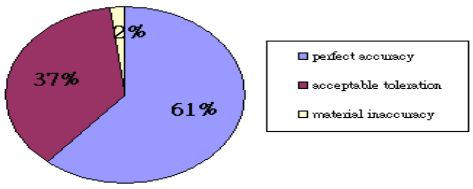

Fig. 11. Distribution of student survey results.

Every student volunteer followed such a procedure to rate and justify his/her answers. We gave the students two weeks to finish the task. At the end of the experiment, we received all 47 students' answers for all 200 test cases. Accumulating all of their ratings and filled questionnaires, we concluded that the students are highly satisfied with our adapted Web pages. As shown in Fig. 11, 61% of the pages are marked as satisfied; 37% of the pages are marked as tolerable; only 2% of the pages could not be fully understood.

5.3. Discussion and Lessons Learned

We adopted DOL HTML parser to parse HTML documents at the tag level. The parser is able to detect and correct many HTML syntax errors. Compared to other HTML clean-up tools, DOL HTML parser shows decent parsing ability, while it also creates tag trees. However, it still may not function well for some tool-generated HTML code. In addition, DOL HTML parser may not detect some implicit errors that can only be detected through semantic relationship analysis.

Our mediator service works well with scripts when tags <script>/<noscript> appear in pairs; because <no-script> secures the display position of the elements indicated by <script>. However, scripts may produce dynamic Web page that cannot be decided at the time of static semantic segment detection. How to ensure that dynamically generated information conforms to the semantic segments relationship remains challenging. We plan to investigate the problem by looking inside of the syntax of java script language in our future research.

From our experiments, we found that some semantic relationships between segments still stay undiscovered. Our semantic segment detection method is based on structural semantic relations in HTML documents. If authors do not link two segments through any structural settings (e.g., a person's descriptions and his/her picture are put into two non-related tables), our system may not detect their relationship. Currently, we use keyword matching to help detect semantic relationships between segments (e.g., an article and its figures). If the figure uses a synonym of a keyword, the current version of our mediator system cannot detect their relationship either. We will explore a solution in our future research.

6. CONCLUSIONS

In this paper, we present an approach that automatically detects semantic segments in Web pages. Our experiments demonstrate the effectiveness of our method. Our presented solution can be directly applied to improve mobile Web surfing that remains a problem due to connection speed, available memory and display size of handheld devices. The angle of our approach fills a gap in the area where more research is necessary. Our conducted experiments on execution time focused on identifying the bottleneck phase of our page segmentation and adaptation algorithm. Our future work will compare our execution time with other systems including both client and server side implementations. In the current experimental design, the way correct rate is calculated focuses on the semantic units that are manually pre-identified. We have noticed some semantic units that are detected by our algorithm but are not manually identified (e.g., by a mistake). Our future work will examine those exceptional cases to study the mistakes that our algorithm may

make. We realize that some assumptions are important for the accuracy of semantic object detection. For example, one assumption is that semantically related segments normally may share some common keywords, as we discussed earlier. We plan to conduct a systematic study on the rules of how to identify the semantic relationships between presentation segments. We plan to investigate a semantics-equipped HTML parsing mechanism to replace the currently used DOL HTML parser. Furthermore, we plan to explore a way to handle java scripts more effectively, i.e., how to handle scripts that may generate dynamic Web page at run time.

ACKNOWLEDGEMENTS

This work is supported by National Science Council, Taiwan under grant NSC95-2520-S-008-006-MY3 and NSC 96-2628-S-008-008-MY3.

REFERENCES

[1] G. Singh, L. Denoue, and A. Das, "Collaborative Note Taking using PDAs Gurminder", *Journal of Information Science and Engineering*, 2005, 21(5): pp. 835-848.
[2] M.K. Kim and K.Y. Jee, "Characteristics of Individuals Influencing Adoption Intentions for Portable Internet Service", *ETRI Journal*, 2006, 28(1): pp. 67-76.
[3] Oracle, "Oracle Application Server Wireless", Available from: http://www.oracle.com/technology/tech/wireless/index.html.
[4] Sun, "Java System Portal Server Mobile Access", Available from: http://www.sun.com/software/products/portal_srvr/index.xml.
[5] IBM, "WebSphere Transcoding Publisher", Available from: http://www-306.ibm.com/software/pervasive/transcoding_publisher/.
[6] S.J.H. Yang, J. Zhang, and I.Y.L. Chen, "A UOI-Based Content Adaptation Method for Improving Web Content Accessibility in the Mobile Internet", *ETRI Journal*, 2007, 29(6): pp. 794-807.
[7] A. Moon, H. Kim, H. Kim, and S. Lee, "Context-Aware Active Services in Ubiquitous Computing Environments", *ETRI Journal*, 2007, 29(2): pp. 169-178.
[8] G. Berhe, L. Brunie, and J.M. Pierson, "Modeling Service-Based Multimedia Content Adaptation in Pervasive Computing", in Proceedings of *the First Conference on Computing Frontiers on Computing Frontiers*, 2004, pp. 60-69.
[9] L.Q. Chen, X. Xie, W.Y. Ma, H.J. Zhang, H.Q. Zhou, and a. H.Q. Feng, " "DRESS: A Slicing Tree Based Web Representation for Various Display Sizes", 2002.
[10] L. Ramaswamy, A. Iyengar, L. Liu, and F. Douglis, "Automatic Fragment Detection in Dynamic Web Pages and Its Impact on Caching", *IEEE Transactions on Knowledge and Data Engineering*, 2005, 17(6): pp. 859-874.
[11] S. Yu, D. Cai, J.-R. Wen, and W.-Y. Ma, "Improving Pseudo-Relevance Feedback in Web Information Retrieval Using Web Page Segmentation", in Proceedings of *the Twelfth International World Wide Web Conference (WWW)*, May 20-24, 2003, Budapest, Hungary, pp. 11-18.
[12] W3C, "Cascading Style Sheet", Available from: http://www.w3.org/Style/CSS/.
[13] Y. Chen, W.-Y. Ma, and H.-J. Zhang, "Detecting Web Pages Structure for Adaptive

Viewing on Small Form Factor Devices", in Proceedings of *the Twelfth International World Wide Web Conference (WWW)*, May 20-24, 2003, Budapest, Hungary, pp. 225-266.
[14] J. Chen, B. Zhou, J. Shi, H. Zhang, and Q. Fengwu, "Function-Based Object Model towards Website Adaptation", in Proceedings of *the 10th International Conference on World Wide Web (WWW)*, May 1-5, 2001, Hong Kong, China, pp. 587-596.
[15] S. Baluja, "Browsing on Small Screens: Recasting Web-page Segmentation into An Efficient Machine Learning Framework", in Proceedings of *the 15th International Conference on World Wide Web (WWW)*, May 23-26, 2006, Edinburgh, Scotland, pp. 33-42.
[16] O. Buyukkokten, H.G. Molina, A. Paepcke, and T. Winograd, "Power Browser: Efficient Web Browsing for PDAs", in Proceedings of *the SIGCHI Conference on Human Factors in Computing Systems*, Apl. 1-6, 2000, The Hague, The Netherlands, pp. 430-437.
[17] M. Hori, G. Kondoh, K. Ono, S.-i. Hirose, and S. Singhal, "Annotation-Based Web Content Transcoding", in Proceedings of *the 9th International World Wide Web Conference on Computer Networks*, 2000, Amsterdam, The Netherlands, pp. 197-211.
[18] Y. Xiao, Y. Tao, and W. Li, "A Dynamic Web Page Adaptation for Mobile Device Based on Web2.0", in Proceedings of *Advanced Software Engineering and Its Applications*, Dec. 13-15, 2008, pp. 119-122.
[19] Y.-W. Kao, T.-H. Kao, C.-Y. Tsai, and S.-M. Yuan, "A Personal Web Page Tailoring Toolkit for Mobile Devices", *Computer Standards & Interfaces*, Feb., 2009, 31(2): pp. 437-453.
[20] I. Colak, S. Sagiroglu, and H.T. Kahraman, "A User Modeling Approach to Web Based Adaptive Educational Hypermedia Systems", in Proceedings of *IEEE Seventh International Conference on Machine Learning and Applications*, Dec., 2008, pp. 694-699.
[21] A. Mikroyannidis and B. Theodoulidis, "Heraclitus: A Framework for Semantic Web Adaptation", *IEEE Internet Computing*, May, 2008: pp. 45-52.
[22] H.N. Chua, S.D. Scott, Y.W. Choi, and P. Blanchfield, "Web-Page Adaptation Framework for PC & Mobile Device Collaboration", in Proceedings of *the 19th International Conference on Advanced Information Networking and Applications (AINA)*, Mar. 28-30, 2005, Tamkang University, Taiwan, pp. 727-732.
[23] S.J.H. Yang and N.W.Y. Shao, "Enhancing Pervasive Web Accessibility with Rule-Based Adaptation Strategy", *Expert System with Applications*, August, 2007, 32(4).
[24] S.J.H. Yang and N.W.Y. Shao, "An Ontology Based Content Model for Intelligent Web Content Access Services", *International Journal of Web Service Research (JWSR)*, Apr.-Jun., 2006, 3(2): pp. 59-78.
[25] W3C, "Web Ontology Language (OWL)", 2004, Available from: http://www.w3.org/2004/OWL/.
[26] J. Backus, "CDC Algol-60 Version 5 Reference Manual", 1979, Available from: http://www.lrz-muenchen.de/~bernhard/Algol-BNF.html.
[27] UTF-8, "UTF-8", Available from: http://en.wikipedia.org/wiki/UTF-8.
[28] Windows-1252, "Windows-1252", Available from: http://en.wikipedia.org/wiki/Windows-1252.
[29] B. Bos, T. ÇelikIan, I. Hickson, and H.W. Lie, "Cascading Style Sheets, level 2 revision 1 CSS 2.1 Specification, W3C® (MIT, ERCIM, Keio)", Nov., 2006, Available.

Stephen J.H. Yang (楊鎮華) received his PhD degree in Computer Science from the University of Illinois at Chicago in 1995. He is now a Distinguished Professor of the Department of Computer Science and Information Engineering, and the Associate Dean of Academic Affairs at National Central University, Taiwan. Dr. Yang has published 2 books and over 160 journal articles and conference papers. His research interests include Web services, Web 2.0, software engineering, knowledge engineering, semantic Web, and context aware ubiquitous computing. Dr. Yang is currently the Co-Editor-in-Chief of the International Journal of Knowledge Management & E-Learning, and the Associate Editor of the International Journal on Artificial Intelligence Tools, and the International Journal of Systems and Service-Oriented Engineering. In addition, Dr. Yang is on the Advisory Board of Athabasca University, the International Journal of Educational Technology & Society, and the International Journal on Digital Learning Technology. He is also on the Editorial Board of the International Journal of Web Services Research, the International Journal of Knowledge and Learning, the International Journal of Educational Technology & Society, the International Journal of Multimedia Data Engineering & Management, and the Journal of Emerging Technologies in Web Intelligence.

Jia Zhang (張嘉) received her Ph.D degree in Computer Science from University of Illinois at Chicago in 2000. She is now an Associate Professor of Department of Computer Science at Northern Illinois University. Zhang has co-authored 1 book and has published over 100 refereed journal articles, book chapters, and conference papers. She is an Associate Editor of the IEEE Transactions on Services Computing (TSC) and International Journal of Web Ser-vices Research (JWSR). Zhang serves as Program Vice Chair of IEEE International Conference on Web Services (ICWS 2006-2009). Her current research interests center around Services Computing. She is a member of the IEEE.

Stella T.C. Tsai (蔡子辰) received his M.S. degrees in Department of Computer Science and Information Engineering from National Central University in 2007. His research interests include content adaptation and mobile learning.

Jeff J.S. Huang (黃正旭) received his M.S. degree in Department of Computer Science and Information Engineering from Tamking University in 2000. He is now a PhD student in Department of Computer Science and Information Engineering from National Central University. His research interests include Web 2.0, social networking, social computing, context aware ubiquitous computing, CSCL, CSCW, e-learning and knowledge management.