

國立交通大學

電機學院與資訊學院 資訊學程

碩士論文



樣式導向之行動瀏覽網頁產生技術

A Pattern-oriented Scheme for Mobile Web Page Generation

研究生：謝偉德

指導教授：袁賢銘 教授

中華民國九十五年七月

樣式導向之行動瀏覽網頁產生技術
A Pattern-oriented Scheme for Mobile Web Page Generation

研究生：謝偉德

Student：Wei-Te Hsieh

指導教授：袁賢銘

Advisor：Shyan-Ming Yuan

國立交通大學

電機學院與資訊學院專班 資訊學程



Submitted to Degree Program of Electrical Engineering and Computer Science
College of Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Computer Science
July 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年七月


樣式導向之行動瀏覽網頁產生技術

學生：謝偉德

指導教授：袁賢銘

國立交通大學 電機學院與資訊學院 資訊學程（研究所）碩士班

摘 要



由於近年來網路使用人口不斷增加，網際網路已經成為一個提供資訊、娛樂及商業活動的重要平台。隨著科技的發達，可以用來上網的裝置越來越多包括個人數位助理(PDA)、智慧型手機(Smartphone)及掌上型電腦(Pocket PC)。網際網路服務的對象不再只有桌上型及筆計型電腦的使用者，而且可以預見的是，未來會有更多不同裝置提供連結網際網路的功能。

目前存在的網頁，絕大多數都是針對一般電腦環境而設計如800*600以上的螢幕。對於行動裝置的環境而言，這些文件大多無法妥善的呈現。目前解決這個問題的方式大致上可分為兩種，一是針對行動裝置再重新設計新的網頁；二是透過一轉接系統將現有網頁轉換成適合行動裝置的網頁內容。以第一種方式而言，雖然可以讓網頁內容在行動裝置上做最好的呈現，但是加重了網頁設計人員的負擔。使用轉接系統不但省去網頁重新設計的工作，而且不需更動現有的網站系統。然而，一般的轉接系統所提供的網頁在行動裝置上的呈現品質，並不能滿足使用者的需求。

因此我們建構一個基於網頁樣式而轉換網頁的系統，期望能夠以最少的資源耗費提供行動裝置使用者良好的上網環境。

A Pattern-oriented Scheme for Mobile Web Page Generation

Student : Wei-Te Hsieh

Advisor : Dr.Shyan-Ming Yuan

Degree Program of Electrical Engineering and Computer Science

National Chiao Tung University

Abstract

Using mobile devices to surf WWW is popular gradually. However, the Web pages for desktop computers are not conforming to the mobile devices. There are many limitations on mobile devices, such as small screen and low bandwidth. Through using Web adaptation technologies, we adapt the pages to mobile devices. Web pages are transformed into small pages for displaying on small screen finely. In this paper, we introduce representative efforts adopting respective analyzing techniques. These content adapting implementations can automatically adapt web pages to mobile devices. However, there are two issues we must solve. One is content extracting and the other is adapting correctness. Users always only care about a part of the web content. Content adapting applications should provide a function to extract these parts from a web page. Besides, these implementations do not adapt content finely in some kinds of web. Therefore, we proposed a pattern-oriented Mobile Web Page Generation scheme. With our scheme, users can customize their own mobile page easily and rapidly. Finally, we implement a system to prove our scheme and compare with existing implementations.

Acknowledgement

本篇論文的完成，首先得感謝指導教授 袁賢銘教授的悉心指導，讓我順利完成論文。感謝口試委員們百忙之中，給予我許多的論文的建議，讓我受益良多。同時也感謝實驗室的高子漢學長、蔡紀暘同學，在我論文研究中，給了許多的建議，也從你們身上學到許多新的技術與思考方法。也感謝實驗室其它學長及同學給我的建議與幫忙。

兩年的時間很快就過去了，從開始決定進修到完成學業，一路上受到許多人的鼓勵與幫助。感謝宏碁的長官與同事對我的幫助與鼓勵，讓我順利進入在職專班。感謝昆盈的長官與同事對我的支持與包容，讓我順利完成兩年的學業。當然，一定要感謝班上的同學，在學期間互相幫忙與鼓勵，讓兩年的學生生涯充滿了回憶。最後要感謝我的家人及身邊的朋友，因為你們的支持與鼓勵，使我能夠順利完成學業。

Table of Contents

Acknowledgement	v
Table of Contents.....	vi
List of Figures	ix
List of Tables	xi
Chapter 1 Introduction.....	1
1.1. Preface	1
1.2. Motivation	1
1.3. Research Objectives	2
1.4. Research Contribution	2
1.5. Outline of the Paper.....	3
Chapter 2 Background.....	1
2.1. Document Object Model (DOM)	1
2.1.1. DOM.....	1
2.1.2. HTML DOM.....	1
2.2. Asynchronous JavaScript and XML (AJAX)	2
2.3. Muffin.....	3
2.4. Slicing Tree	3
2.5. Web Content Adaptation Technologies.....	4
2.5.1. Deployment	4
2.5.2. Authoring.....	6
2.5.3. Analysis	6
2.6. Related Works	12
2.6.1. WebAlchemist	13
2.6.2. Function-Based Object Mode.....	14

2.6.3.	Content Block Detection	15
2.7.	Commercial Product	16
2.7.1.	Opera Mini™	16
2.7.2.	Google Mobile Proxy	17
Chapter 3	Scheme.....	18
3.1.	Overview	18
3.2.	Page Segmentation	19
3.3.	Pattern String Generation	21
3.4.	Pattern Matching	23
3.5.	Page Generation.....	24
Chapter 4	System Design.....	25
4.1.	Overview	25
4.2.	Web-based Authoring Tool	26
4.3.	Pattern Manager.....	27
4.3.1.	Query Service	27
4.3.2.	Update Service.....	28
4.4.	Mobile Proxy	28
4.5.	Summary.....	29
Chapter 5	Implementation	30
5.1.	Web-based Authoring Tool	30
5.2.	Pattern Manager.....	30
5.3.	Mobile Proxy	31
5.4.	Summary.....	31
Chapter 6	Evaluation	32
6.1.	Usability Test.....	32
6.1.1.	Authoring Tool	32

6.1.2. Mobile Proxy	35
6.2. Summary.....	36
Chapter 7 Conclusion.....	37
7.1. Conclusion.....	37
7.2. Future Work.....	37
Chapter 8 Reference.....	39
Curriculum Vitae.....	41
Education.....	41
Experience	41



List of Figures

Figure 2-1 The comparison between classic and Ajax model	2
Figure 2-2 A sample of binary slicing tree	3
Figure 2-3 An example using the outlining transform.....	7
Figure 2-4 An example using the generalized transform.....	7
Figure 2-5 An example using the index transform	8
Figure 2-6 An example using the selective elision transform	9
Figure 2-7 An example using the Vision-based page segmentation.....	10
Figure 2-8 A FOM representation of the Web page.....	11
Figure 2-9 An example using the first sentence transforms.	11
Figure 2-10 An example using the image reduction and elision transform....	12
Figure 2-11 WebAlchemist.....	14
Figure 2-12 FOM for mobile devices.....	15
Figure 2-13 Content Block Detection.....	16
Figure 2-14 Browsing the CNN Website through Opera Mini™.....	17
Figure 2-15 Browsing the CNN Website through Google mobile proxy	17
Figure 3-1 The VIPS tree construction algorithm	19
Figure 3-2 The algorithm of getting the AOC value	20
Figure 3-3 An example of VIPS tree construction	20
Figure 3-4 An example of slicing tree construction	21
Figure 3-5 The algorithm of getting the node type.....	22
Figure 3-6 An example of Pattern String Generation	23
Figure 3-7 An example of Pattern Matching	24
Figure 3-8 An example of Page Generation	24
Figure 4-1 An overview of the system design	25

Figure 4-2 The design of Web-based Authoring Tool 26

Figure 4-3 The design of Pattern Manager 27

Figure 4-4 The design of Mobile Proxy 28

Figure 6-1 The authoring example on CNN Asia News..... 33

Figure 6-2 The authoring example on CNN Asia and Europe News 33

Figure 6-3 The authoring example on CNN World/Africa News..... 34

Figure 6-4 The authoring example on CNN World/MIDDLE EAST and CNN
World/Africa News 34

Figure 6-5 Browsing the CNN Asia and Europe News through Mobile Proxy
..... 35

Figure 6-6 Browsing the World/Africa and CNN World/Middle News through
Mobile Proxy 36



List of Tables

Table 2-1 The classification of content adaptation technologies.....	4
Table 2-2 The comparison between three different deployments.....	4
Table 2-3 The major techniques adopted by adaptation approaches	13



Chapter 1 Introduction

1.1. Preface

World Wide Web becomes an important platform for information dissemination, business transaction and digital entertainment recently. More and more people go online for various applications, for instance, shopping, news, auctions and games. Therefore, how to enable users to surf World Wide Web everywhere is a popular topic in information technology domain.

In order to serve the mobile users, Web designers must additionally prepare multiple Web pages for mobile devices formerly. Content adaptation technologies enable mobile user to browse through the existing Web pages designed for personal computers. Through these technologies, a Web designer only needs to author a unique edition for each Web page.



1.2. Motivation

The World Wide Web is becoming an important infrastructure for our life. There are abundant information and knowledge in existing Web pages. However, most of these Web pages are designed for PCs and not conforming to the mobile devices.

Using mobile devices to surf World Wide Web is popular gradually. There are many limitations on mobile devices, such as small screen and low bandwidth. Through using Web content adaptation technologies, we adapt the pages to mobile devices.

1.3. Research Objectives

In this section, three major research objectives are listed and introduced briefly.

Easy to Use

For reducing the learning time, the toolkits must be easy to use for Web author. For this purpose, these tools should be developed on a popular platform. Furthermore, these tools should provide a friendly user interface.

Fast to Mobilize

How to rapidly set up a mobile Web is a major topic of our studies. Page authoring engages the most time in mobile Web page constructing. The authors spend much time and many efforts on authoring a mobile page. In order to accelerate the page authoring, we can provide a convenient authoring tool.

Bandwidth Consumption

The existing Web pages designed for personal computer usually contain plentiful contents. However, they consume much transmission bandwidth while browsed by users. The data transmission could be reduced through content adaptation technologies.

1.4. Research Contribution

In order to attain the objectives mentioned in section 1.3, we provide a scheme and an implementation to prove the scheme. This paper discusses those issues about the web content adaptation and our corresponding solutions. The major contributions of this research are listed below:

1. A Web-based authoring tool allows a user to author the Web pages from different computers, and requires no pre-installation of any software.
2. Blocks in a Web page can be chosen correctly, under the premise that the layout

of a Web page doesn't change frequently.

3. A pattern-oriented mobile Web page generating system.

1.5. Outline of the Paper

There are seven chapters in this article. Following is a brief description of the content of each chapter:

1. In Chapter 2, we introduce the background of our solution and the related work including two commercial products and three academic efforts.
2. In Chapter 3, we introduce our scheme and the four major components of it.
3. In Chapter 4, we propose a system to achieve our scheme and make a brief introduction of the system design.
4. In Chapter 4, the implementation details of our system are described in this chapter.
5. In Chapter 6, to evaluate our scheme, we take a simple test and verify on the system.
6. In Chapter 7, we make a conclusion of this work and list several future works for reference.



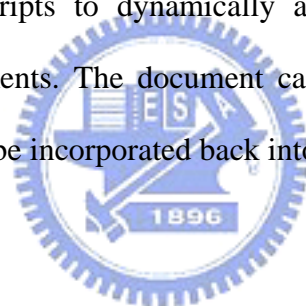
Chapter 2 Background

In this chapter, we introduce the background of our research and brief Web transformation technologies and those used in our scheme. Finally, we introduce several related works and commercial products.

2.1. Document Object Model (DOM)

2.1.1. DOM

The Document Object Model [1] is a platform – and language – neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page.



2.1.2. HTML DOM

The HTML DOM [2] is the Document Object Model for HTML. It defines a standard set of objects for HTML, and a standard way to access and manipulate HTML documents.

The HTML DOM views HTML documents as a tree structure of elements. All elements, along with their text and attributes, can be accessed and manipulated through the DOM tree.

2.2. Asynchronous JavaScript and XML (AJAX)

Asynchronous JavaScript and XML (AJAX) [3] [4], is a Web development technique for creating interactive Web applications. The intent is to make Web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire Web page does not have to be reloaded each time the user makes a change. This is meant to increase the Web page's interactivity, speed, and usability.

Ajax isn't a technology. It contains several technologies:

- standards-based presentation using XHTML [5] and CSS [6];
- dynamic display and interaction using the Document Object Model;
- data interchange and manipulation using XML [7] and XSLT [8];
- asynchronous data retrieval using XMLHttpRequest;
- and JavaScript [4] binding everything together.

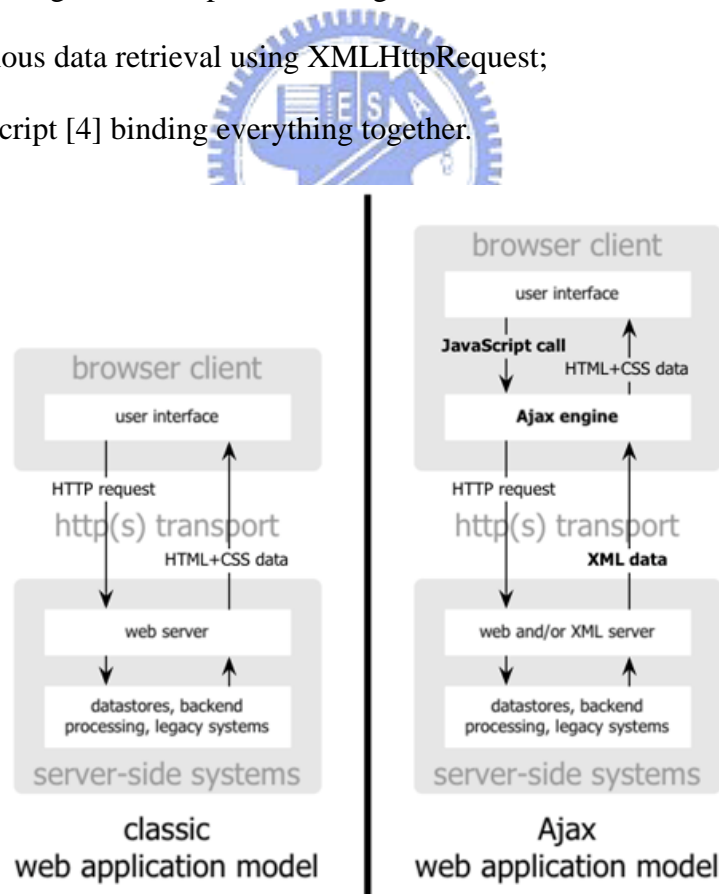


Figure 2-1 The comparison between classic and Ajax model

2.3. Muffin

Muffin [9] is a Web filtering system written entirely in Java. With this system, we can filter any HTTP data sent and received by Web Browsers. Muffin is freely available under the GNU General Public License [10]. Users can write our own filters in Java [11] using the provided filter interfaces

2.4. Slicing Tree

A slicing floorplan [12] is a decomposition of a block with horizontal and vertical cuts. We can represent it as a binary tree, called a binary slicing tree. An internal node in the slicing tree represents a cut, “+” nodes represent the horizontal cuts or and “*” nodes represent the vertical cuts. A leaf node in the tree represents a block that has no cut through.

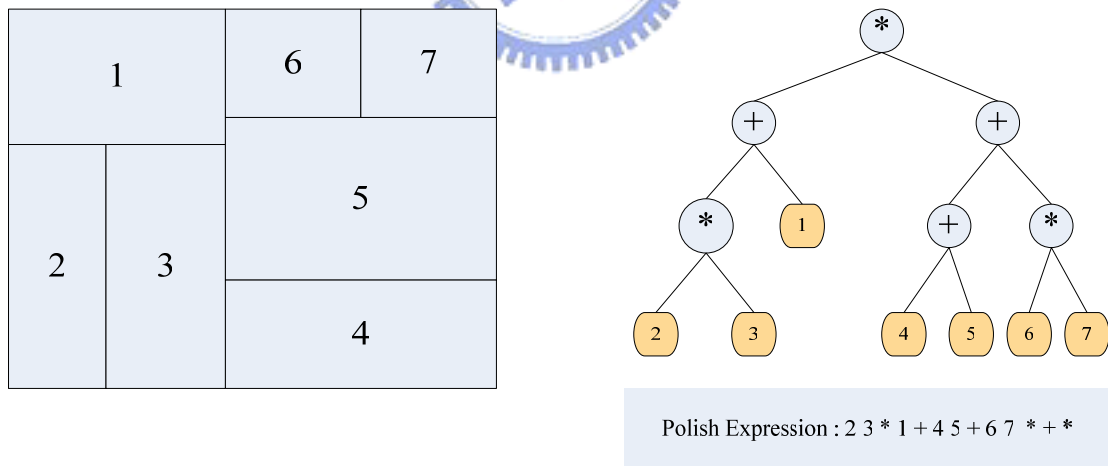


Figure 2-2 A sample of binary slicing tree

2.5. Web Content Adaptation Technologies

In recent years, numerous Web content adaptation technologies were proposed. The following we will introduce these technologies from three aspects, deployment, analysis and authoring. Table 2-1 shows the classification of content adaptation technologies according to three aspects.

Table 2-1 The classification of content adaptation technologies

Item	Scheme		
Deployment	Server-side	Proxy-side	Client-side
Authoring	Manual		Automatic
Analysis	Dom-based	Vision-based	Component-based

2.5.1. Deployment

In this section, we will introduce the three deployment scheme listed in Table 2-2.

Table 2-2 The comparison between three different deployments

Item \ Scheme	Server-side Deployment	Proxy-side Deployment	Client-side Deployment
Transformation Quality	Average	Average	Good
Performance	Good	Average	Poor
Construction Efforts	Poor	Good	Poor
Browsing Scope	Poor	Good	Good

2.5.1.1. Server-Side Deployment

In this deployment scheme, the Web server provides a dedicated adaptation module. This dedicated module fetches Web pages from the original server, convert it, and then transfer the adapted result to the client. By using this scheme, the Web designers can specially author their Web pages for mobile devices. All the contents are transferred in the server side. Therefore, the consumption of network bandwidth is less than other two schemes.

2.5.1.2. Proxy-Side Deployment

We can deploy the page adaptation module on a proxy server; the advantage is that modification isn't necessary for client devices and Web servers.

When a client proposes a Web page request via the proxy server, the proxy server fetches and transforms the corresponding Web page from the original server. The client receives the transformed Web pages from the proxy server. This deployment scheme improves user perceived delay: the proxy server usually has a better Internet connection and a much faster processor than mobile devices.

2.5.1.3. Client-Side Deployment

We can implement page adaptation algorithms as plug-ins for current mobile Web Browsers. However, the mobile devices often have limited hardware and software capabilities. While using the client-side deployment schemes, we must consider these limitations. In this scheme, the plug-in could fully employ the capabilities of mobile devices. We could obtain the best visual quality by adopting the client-side deployment scheme.

2.5.2. Authoring

2.5.2.1. Manual Re-authoring

In manual re-authoring, Web authors prepare multiple versions of a Web page targeted to various platforms, including the Wireless Application Platform. Although this scheme produces high-quality pages for specific devices, the number of Web page accessible via mobile devices is limited. Because no one can predict how Web surfing will progress.

2.5.2.2. Automatic Re-authoring

Through auto-transforming software, automatic re-authoring transform Web pages designed for the personal computers without artificial operation. It re-authors the document via a series of transformations so that it can be appropriately displayed on the device. This scheme can be deployed either on the client, on the server or on a proxy server.

2.5.3. Analysis

2.5.3.1. Dom-based Analysis

Outlining Transform

On the World Wide Web, many documents, such as e-books are structured well. As shown in Figure 1, outlining transforming [13] removes the content of each section from these documents, and then creates a sub-page containing the content removed. In the final step, the section header is covered into an anchor tag linking to the corresponding sub-page.

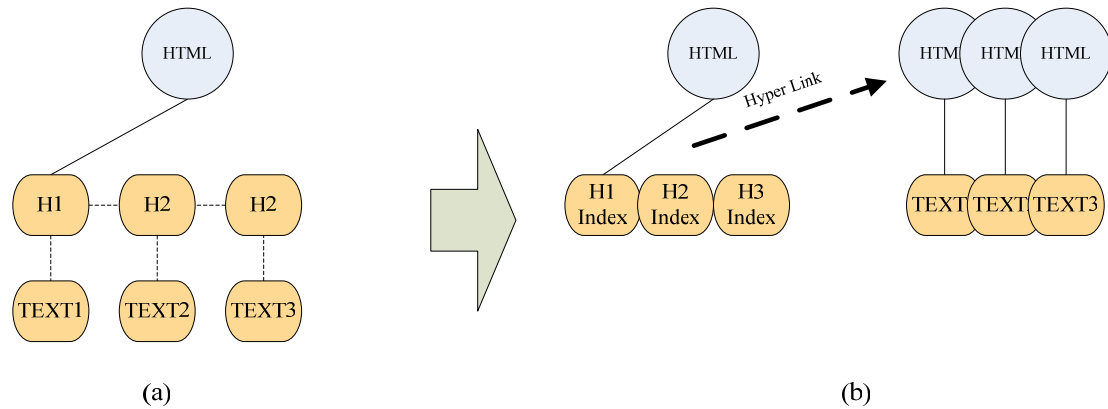


Figure 2-3 An example using the outlining transform (a) The original web page contains section tags; (b) Each section is transformed to a sub-pages

General Outlining Transform

Grouping and summarizing repeated layout patterns, the generalized outlining transform [14] reduces the space of a Web page. Figure 2 illustrates the principle of generalized outlining transform. In order to find repeated layout patterns of a Web Page, grouping function of the transform traverses the DOM tree to find and collect repeated object sequences. After grouping repeated layout patterns, the summarizing function replaces the repeated layouts with hyperlink link to corresponding sub-page. Generalized Outlining Transform performs a fine web adaptation in complex web pages with multiple repeated layout patterns.

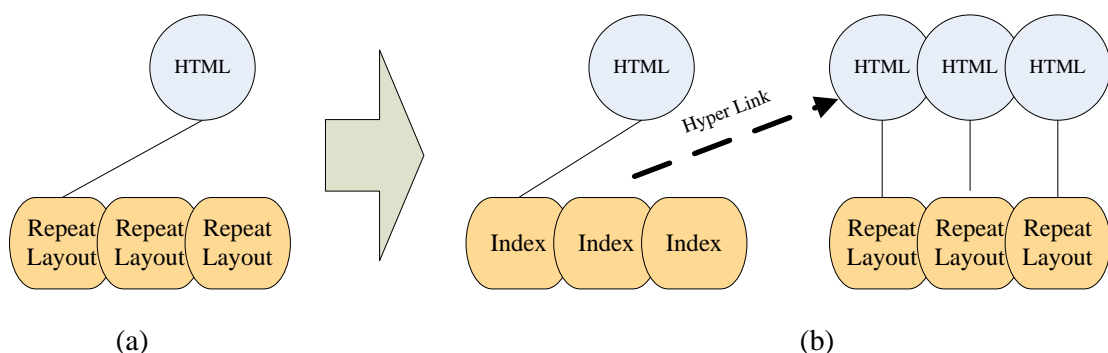


Figure 2-4 An example using the generalized transform (a) The original Web page contains repeat layouts; (b) The page is transformed into an index page and several sub-pages.

Indexed Segment Transform

This transform segments the content of the original page into a sequence of small sub-pages that adapting the small screen. In index segment transform [13], we look for logical elements in the page, such as lists, paragraphs, or tables; then fill output sub-pages with these elements while the page is full. When a single logic element is too large to put in the sub-page, index segment transform performs a secondary segmentation. After generating output pages, an index page will be constructed contains anchor tags link to appropriate sub-page.

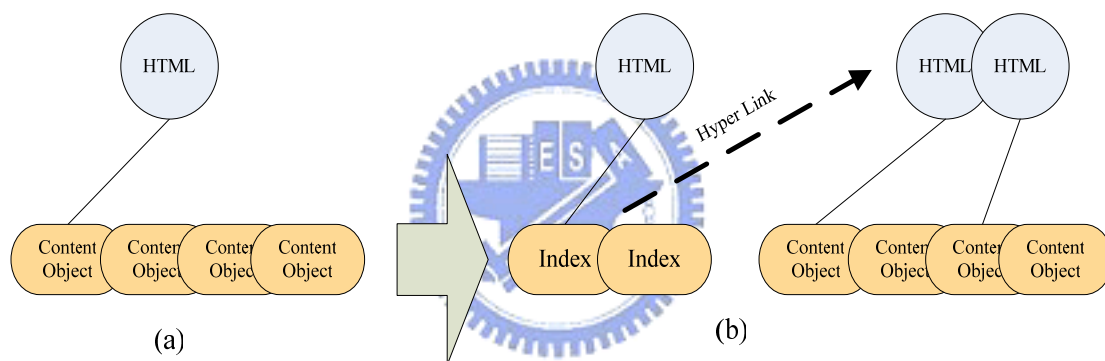


Figure 2-5 An example using the index transform; (a) The original Web page contains many objects; (b) The page is transformed into an index page and several sub-pages.

Selective Elision Transform

This transform keeps the major cells and elides other cells from the table structure. To select and preserve the important table cells, selective elision transform [14] analyzes the syntactic attributes such as font size, cell width and amount of elements. For example, the font size of headline is the largest one in the news Web-side, and selective elision transform will keep the table cell containing headline news while adapting a news Web. For displaying normally, the transform also keeps the table structure as much as possible.

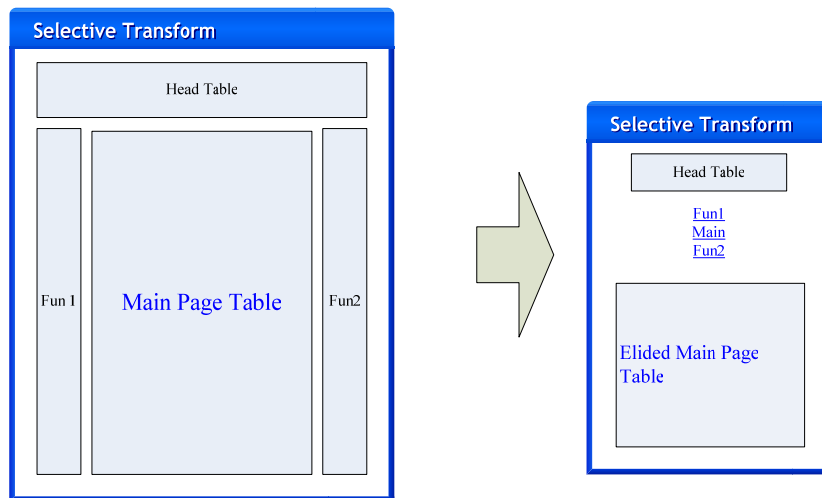


Figure 2-6 An example using the selective elision transform (a) The original Web page contains some table cells; (b) The major cell is preserved and the others are omitted.

2.5.3.2. Vision-based Analysis

Vision-based page segmentation

Using the spatial and visual cues, The Vision-based page segmentation (VIPS) algorithm [15] segments the Web pages into several semantic parts. To segment Web pages, VIPS algorithm consists of the following three steps in a round: block extraction, separator detection and content structure construction. In the first round of segmenting, the Web page is segmented into several big blocks. If the block size is large than the pre-defined size, VIPS algorithm performs the same segmenting process on each big block recursively. The segmenting process finishes while each block size was small than the pre-defined size.

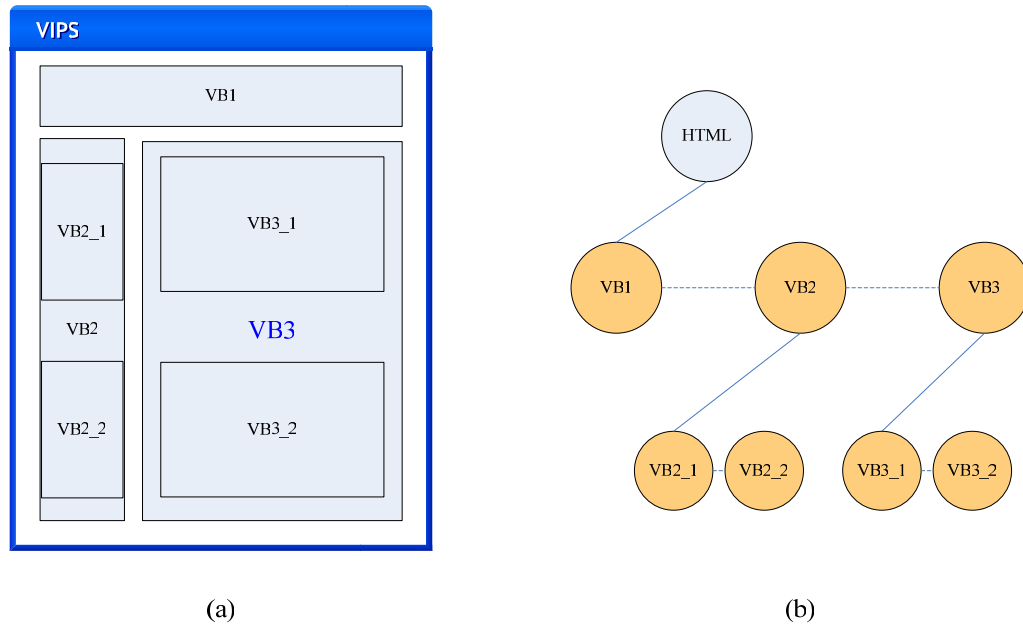


Figure 2-7(a) the segmentation of Web page; (b) tree-based representation of the segmented page

2.5.3.3. Component-based Analysis

Function-Based Object Model

In the Function-Based Object Model (FOM) [16] theory, we attempt to understand the author's intention by identifying the two functions, basic function and specific function of each object. Basic FOM represents an Object by its basic functional properties, layout format and media type. Specific FOM represents the Object by category, such as information, navigation, and interaction. In Web content adaptation, FOM transforms each content object base on its Basic FOM and Specific FOM.

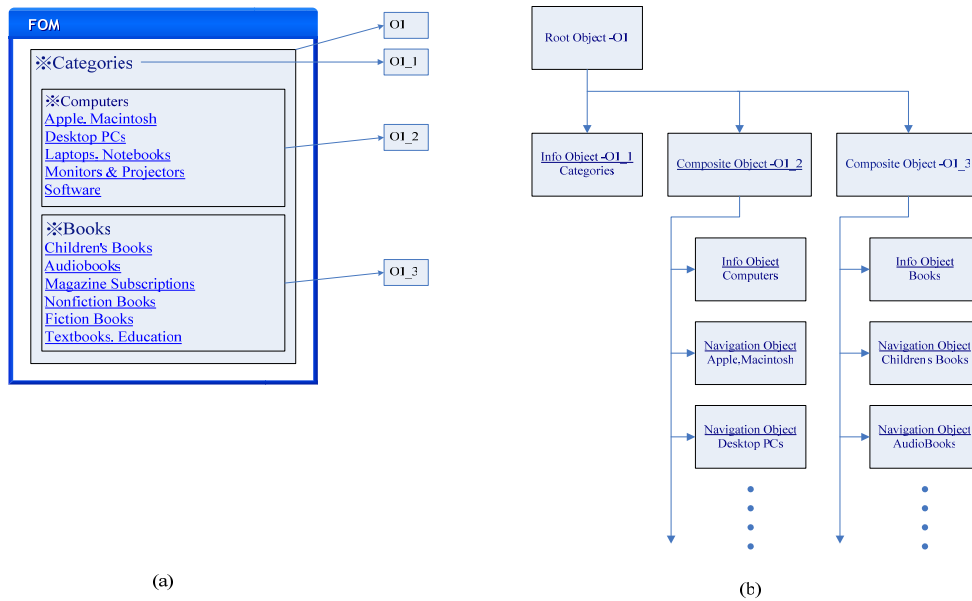


Figure 2-8 A FOM representation of the Web page

First Sentence Elision Transform

First sentence elision [13] is a convenient technique of reducing text blocks without section headers. This technique replaces each text blocks with its first sentence and makes the first sentence into a hypertext link to the text block.

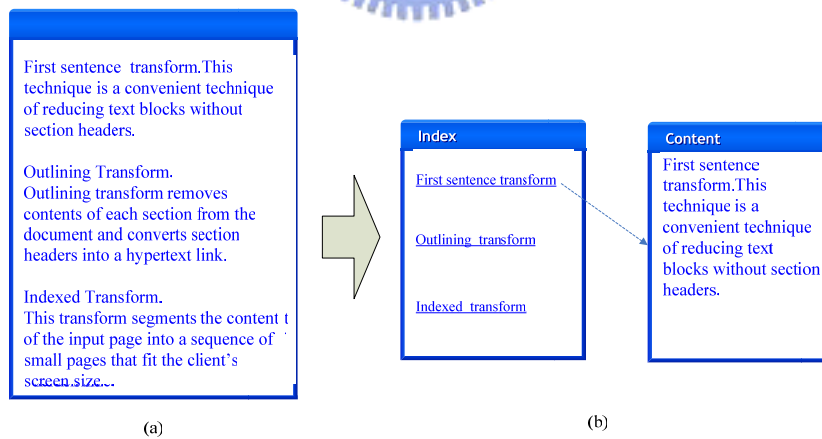


Figure 2-9 An example using the first sentence transforms. (a) The original page contains several text blocks; (b) each text block is replaced by the first sentence in the index page.

Image Reduction and Elision Transform

Images usually occupy much space of a Web page. Web adapting systems usually employ image reduction and elision transforms [13] in order to reduce the page space. To reducing the image size, these transforms scale down images with pre-defined scaling factors and makes reduced images hyperlink links to each original image.

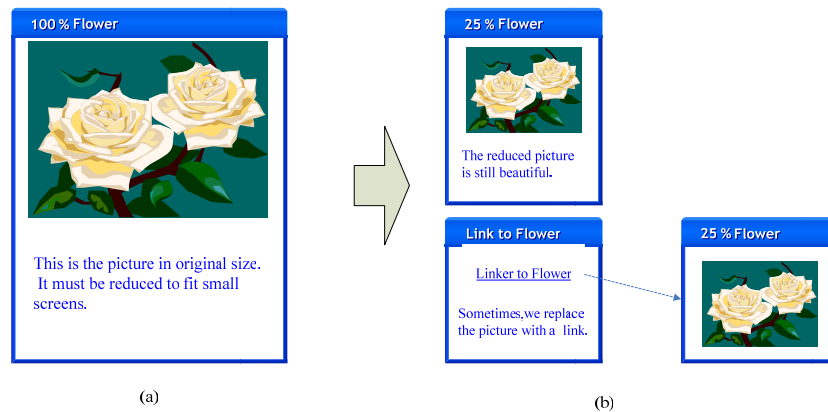


Figure 2-10 An example using the image reduction and elision transform; (a) The original Web page contains a large image; (b) the image is reduced into a small one.

2.6. Related Works

The Designers of adapting systems usually apply several transform heuristics to adapt Web content finely. Yonghyun Hwang employs outlining transform and generalized outlining transform to group page contents. Yu Chen uses content block detection, separator detection and generalized outlining transform to segment Web pages. We will introduce three approaches in this section, and each approach adopts the individual content analyzing strategy.

Table 2-3 The major techniques adopted by adaptation approaches

Approaches	Major analysis techniques
WAP adaptation base on FOM	Function-Base Object Model
Adapting Web Pages for Small-Screen Devices	Vision-based page segmentation Generalized outlining transform
WebAlchemist	Improve outlining transform Generalized outlining transform Selective elision transform First sentence elision transform Image Reducing and elision transform Index segmentation transform

2.6.1. WebAlchemist

WebAlchemist [14] is a structure-aware Web transforming system that contains six transform heuristics listed in table 2-3. By employing the first three structure-aware transforming heuristics, WebAlchemist preserves the structure of Web pages in the transforming process. With transforming manager that decides how the six heuristics are used and the order of using an individual heuristic, WebAlchemist performs an adaptive Web transformation.

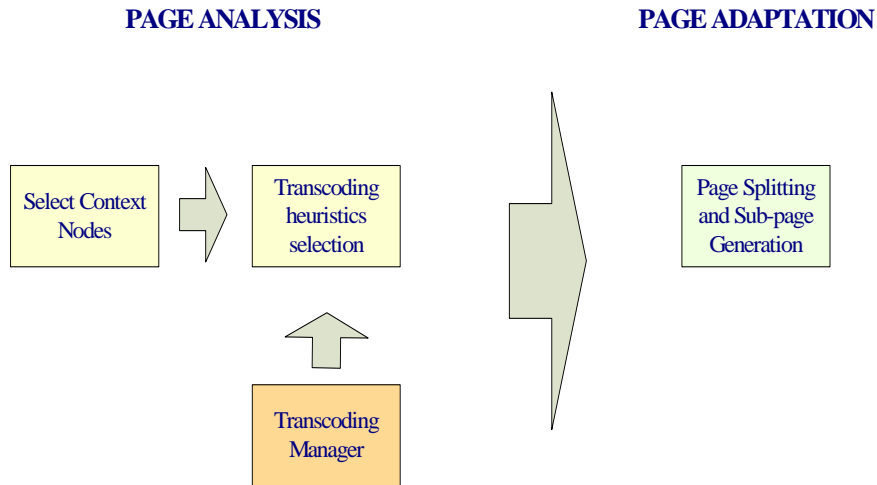


Figure 2-11 WebAlchemist

2.6.2. Function-Based Object Mode

WAP adaptation base on FOM [16] [18] represents Objects with their function properties and categories. For this purpose, this approach takes Basic FOM and Specific FOM Generation in page analysis phrase. Object detection is the first stage of Basic Object Generation. The properties of each Basic Object and the Composed Objects are detected in this stage. After Basic Object Generation, Specific Object Generation classifies each Basic Object into several specific functions.

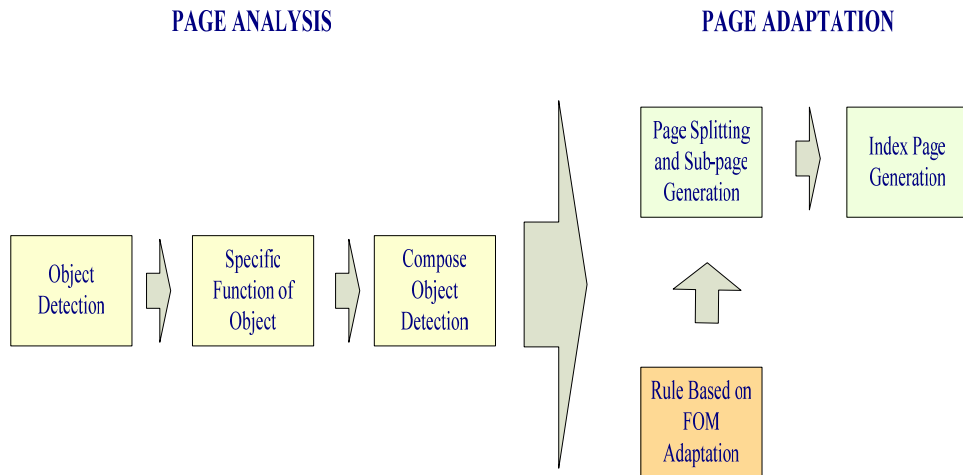


Figure 2-12 FOM for mobile devices

2.6.3. Content Block Detection

In this approach, the page analysis stage comprises three steps. In the beginning, we analyze the HTML DOM tree and segments pages to header, footer, side bar and body blocks; then splitting content blocks base on explicit separators inside each high level blocks. Eventually we separate each content block by implicit separators.

In the first step named High Level Content Block Detection [17], the analyzer traverses the DOM tree from the `<body>` node to leaves to select appropriate nodes and put them into a high level content block. In some cases, the page author puts several high level content block into one node. The analyzer moves one level down to consider its child nodes as units in this situation.

In the second step, analyzer look for explicit separators to partition each content block into several pieces. HTML tags such as `<hr>`, `<table>`, `<td>` and `<div>` are identified as explicit separators. By searching these tags, the analyzer can segment a content block into sub-blocks.

Following the explicit separators detection, the analyzer finds implicit separators to separate the content in each block. The Web page is segmented into small blocks adapting small screen finally.

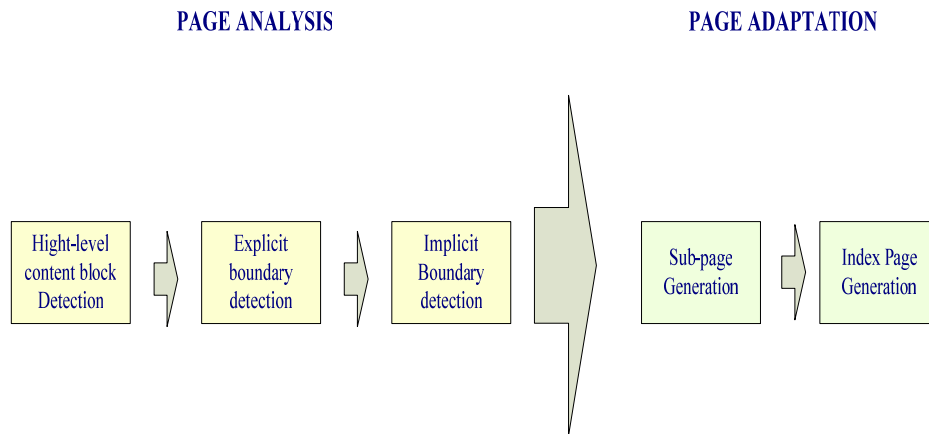


Figure 2-13 Content Block Detection

2.7. Commercial Product

2.7.1. Opera Mini™

Opera Mini™ [19] is a fast and tiny Web Browser that allows users to access the full Internet on their mobile phone. Opera Mini™ is a typical client-based deployment scheme. Through Opera Mini™, users could surf all the World Wide Web directly. As shown in Figure 2-14, we browse a page of the CNN Website [20] via Opera Mini™.



Figure 2-14 Browsing the CNN Website through Opera Mini™

2.7.2. Google Mobile Proxy

Google [21] [22] recently developed a mobile proxy service. This Proxy service could slim down Web pages while users visit them from their mobile device. As shown in Figure 2-15, the page of the CNN Website was transformed into a small page.

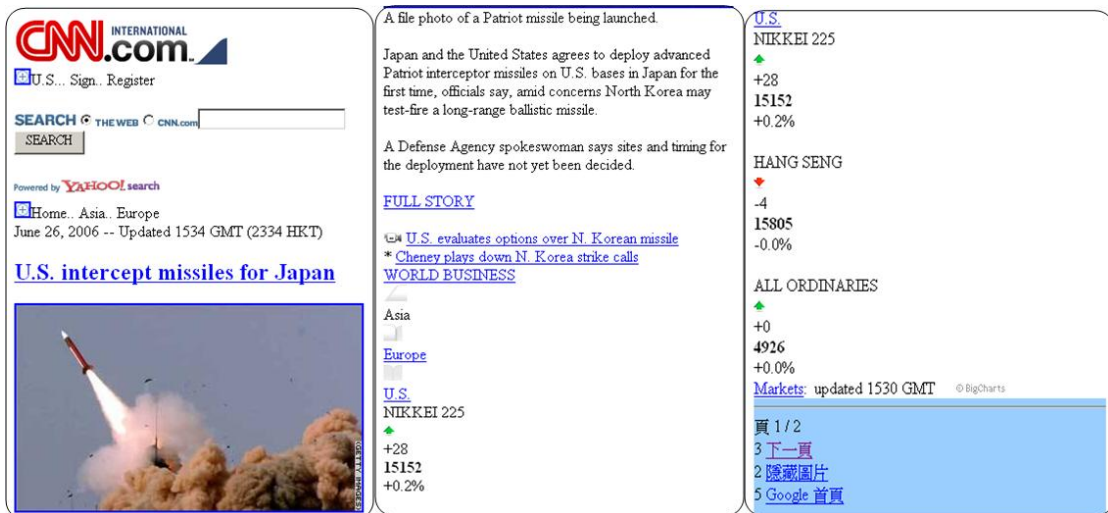


Figure 2-15 Browsing the CNN Website through Google mobile proxy

3.1. Overview

Our scheme consists of four major procedures: page segmentation, pattern string generation, pattern matching and page generation. In the first step – page segmentation, we implement the VIPS algorithm to segment the target page into small blocks and construct a VIPS tree. In second step – pattern string generation, we transform the VIPS tree into a slicing tree and generate a pattern string – the Polish Expression [12] of this binary slicing tree. In the third step – Pattern Matching, we compare this pattern string with all existing pattern strings in the database. If there is any existing pattern string matched, we take the annotations of the existing pattern as result. In the fourth step – page generation, we generate the result page according to the source page and the annotations.

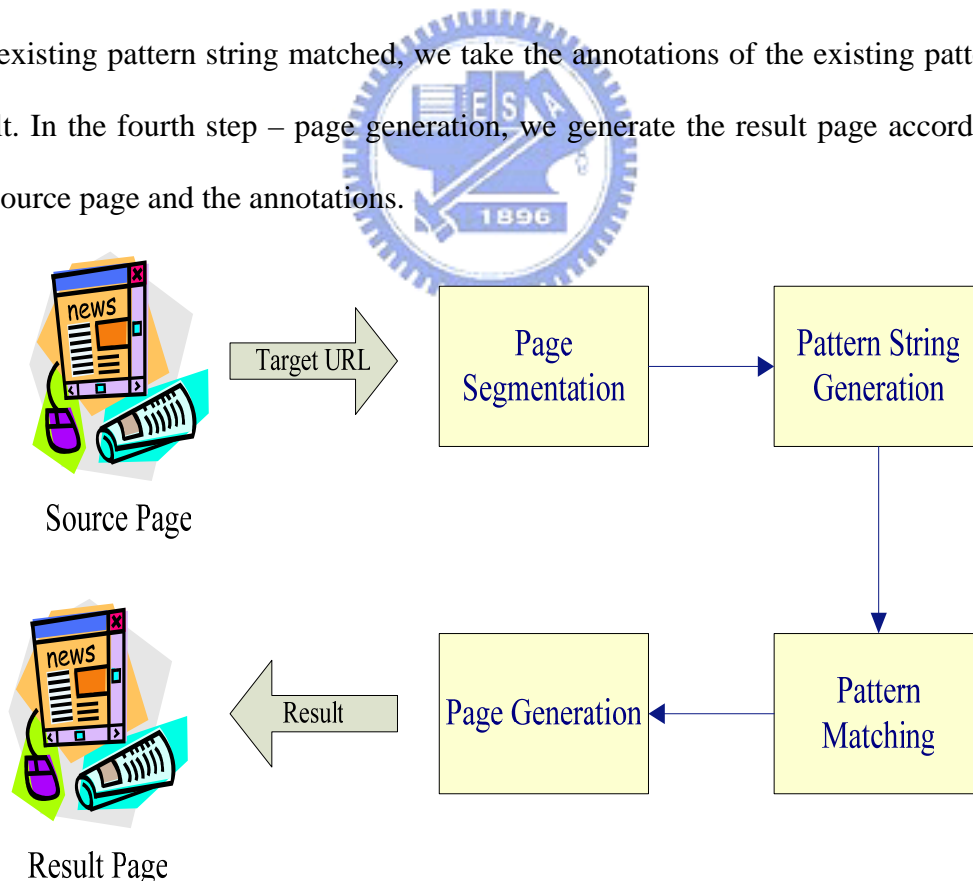


Figure 3-4 An overview of Pattern-oriented Scheme for Mobile Web Page Generation

3.2. Page Segmentation

In this step, we implement parts of VIPS algorithm and change several method of it for our purpose. We segment the Web page into blocks and retain the hierarchical structure. To extract the content blocks, we find explicit separators by analyzing the presence of tags such as *<TABLE>*, *<TR>*, *<TD>*, and *<DIV>*. In general, every node in the DOM tree can represent a visual block. Different from VIPS algorithm, we using the AoC (Area of Content) value instead of DoC (Degree of Coherence) value. Each node will be assigned an AoC value to indicate how many square measures are occupied by the content within the node. We extract the content blocks recursively while the AoC value of the node is great than pre-defined AoC. After all the blocks are processed, we obtain a vision-based content structure tree.

```
VipsTreeConstruct(Node treeNode, VipsNode vipsNode) {  
  
    if(treeNode.AocValue < Pr-define AOC ) return;  
    else{  
        for EACH child of treeNode {  
            Construct a VipsNode and Put this child Node in it;  
            VipsTreeConstruct(this ChildNode, current VipsNode);  
        }  
    }  
  
}
```

Figure 3-1 The VIPS tree construction algorithm

```

var getAocValue(Node) {
    var value = 0;
    if(Node is Text Node) {
        return Length * FontSize;
    }
    else if(Node is Image Node) {
        return ImageSize;
    }
    else if (Node is Element Node){
        for EACH child of Node {
            value = value + getAocValue(this ChildNode);
        }
    }
    return value;
}

```

Figure 3-2 The algorithm of getting the AOC value

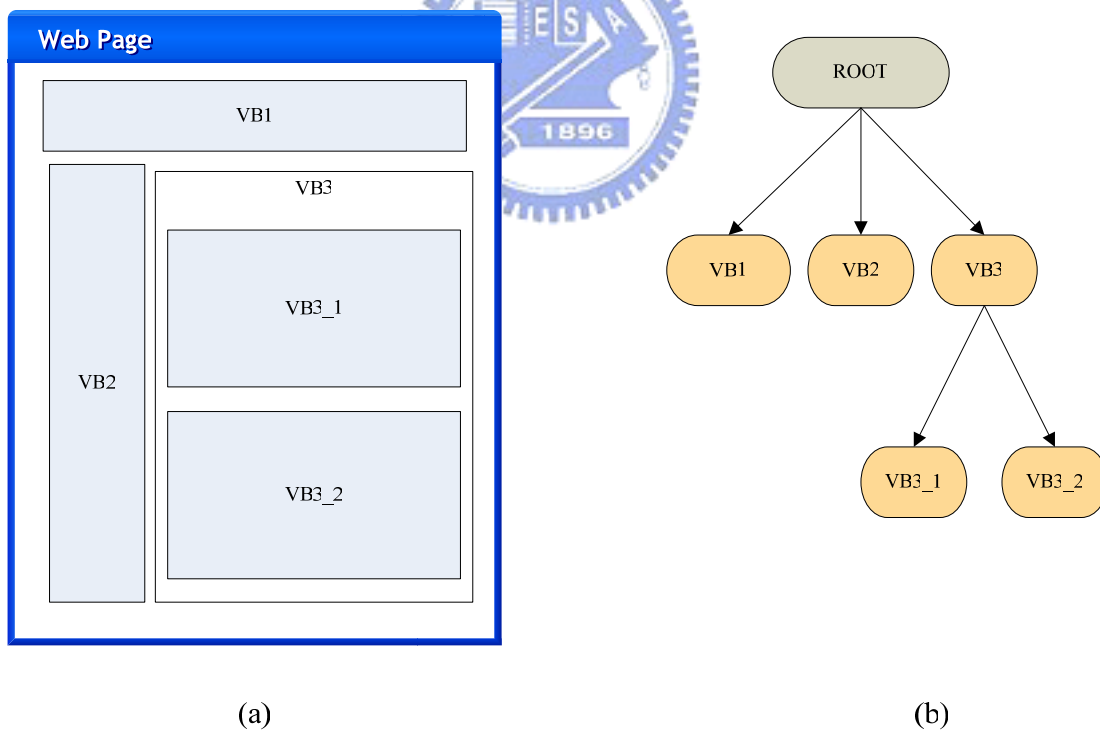


Figure 3-3 An example of VIPS tree construction

3.3. Pattern String Generation

In order to represent the page structure in a string, we transform the VIPS tree to a binary slicing tree. After the binary slicing tree is constructed, we perform a postorder traversal of it and obtain a string – Polish Expression.

For transforming the VIPS tree to a slicing tree, we perform a level-order traversal of the VIPS tree. In the traversal, we construct corresponding slicing tree node according to different states of VIPS tree node. If the VIPS node has next sibling node, we construct an operator node and put these two VIPS tree node as its child nodes. If the VIPS node has child nodes, we construct an operator node to the slicing tree. Furthermore, if the VIPS node is a left node, we construct a content node. According to the tag name of the node, we assign the operator type to each operator node. The node with tag name “TD” is assigned to a “*” symbol, and the node with other tag is assigned to a “+” symbol.

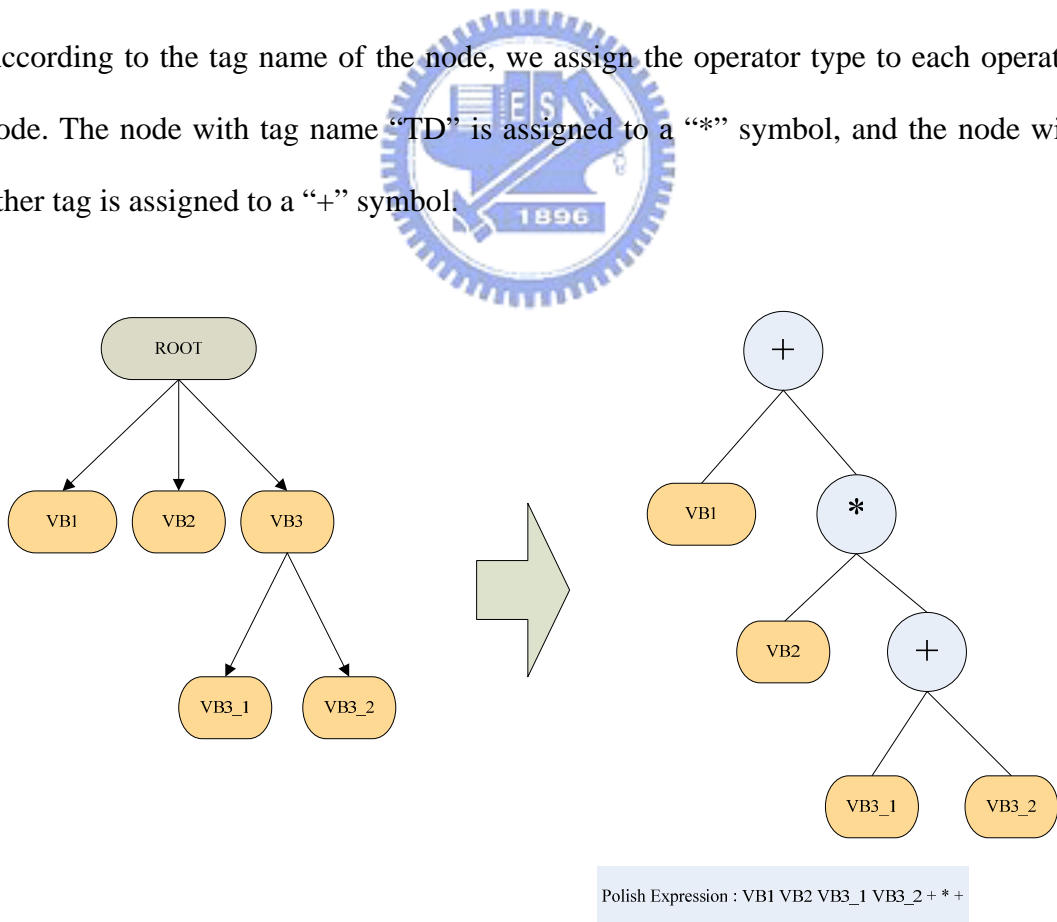


Figure 3-4 An example of slicing tree construction

```

var getNodeTypes(Node) {
  if(Node is Text Node) {
    if(ParentNode of Node is Anchor Node)
      return Index Type;
    else
      return Content Type;
  }
  else if(Node is Image Node or Object Node) {
    return Content Type;
  }
  else if (Node is Element Node) {
    var index,content;
    for EACH child of Node {
      var type = getNodeTypes(ChildNode);
      if(type is index)
        index = index + getAocValue(ChildNode);
      else
        content = content + getAocValue(ChildNode);
    }
    if(index >= content)
      return Index Type;
    else
      return Content Type;
  }
}

```

Figure 3-5 The algorithm of getting the node type

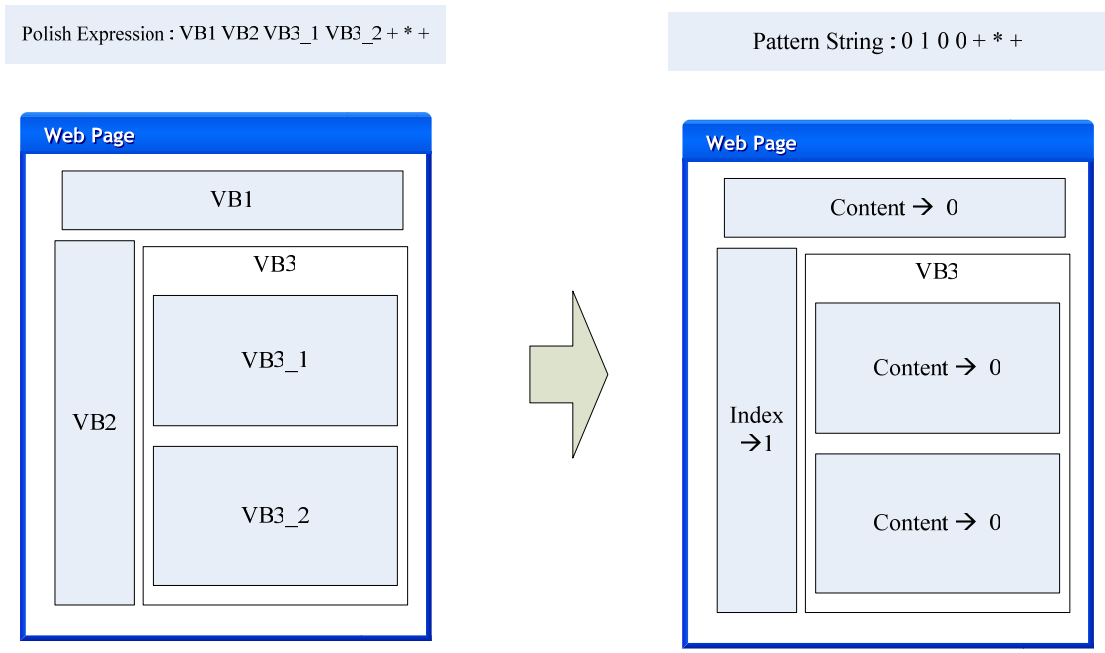


Figure 3-6 An example of Pattern String Generation

3.4. Pattern Matching

The Polish Expression represents a slicing tree structure in a string. We can compare the tree structures by comparing their Polish Expression strings. If there are two trees that have the same Polish Expression strings, they have the same tree structure. However, the tree structure of Web pages is usually complex. It is difficult to find two pages that have the same pattern string. For Web page similar comparisons, we compare two pattern strings from beginning, and find out the number of the same characters. We divide this number by the string length and obtain a similar value. If this similar value large than the pre-defined value, these two pattern are matched.

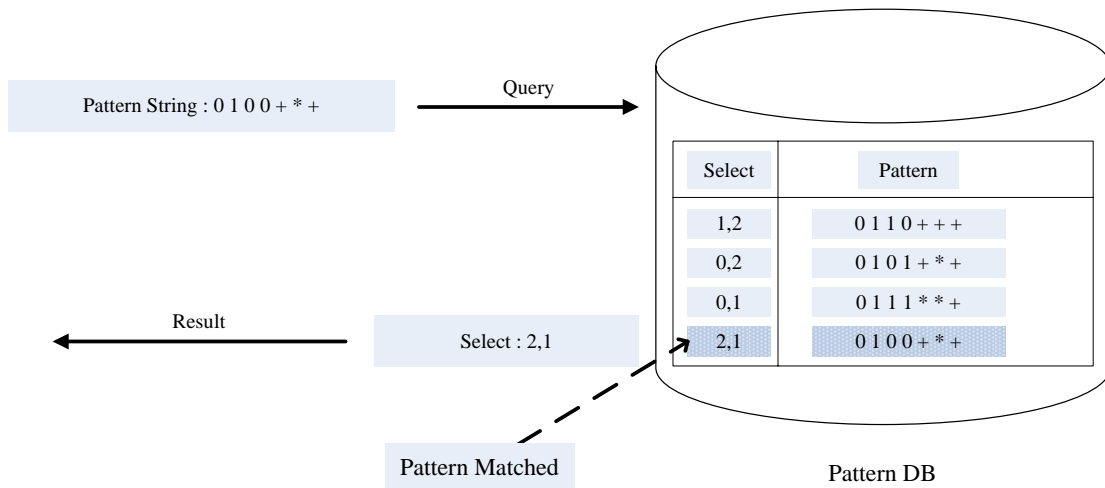


Figure 3-7 An example of Pattern Matching

3.5. Page Generation

After the pattern matching process, we obtain a page annotation for the Web page. As the Figure, the annotation contains a sequence of block numbers separated by commas. A block number is the position of the block in the pattern string. According to this sequence, we represent these blocks as the output page.

Polish Expression : VB1 VB2 VB3_1 VB3_2 + * +

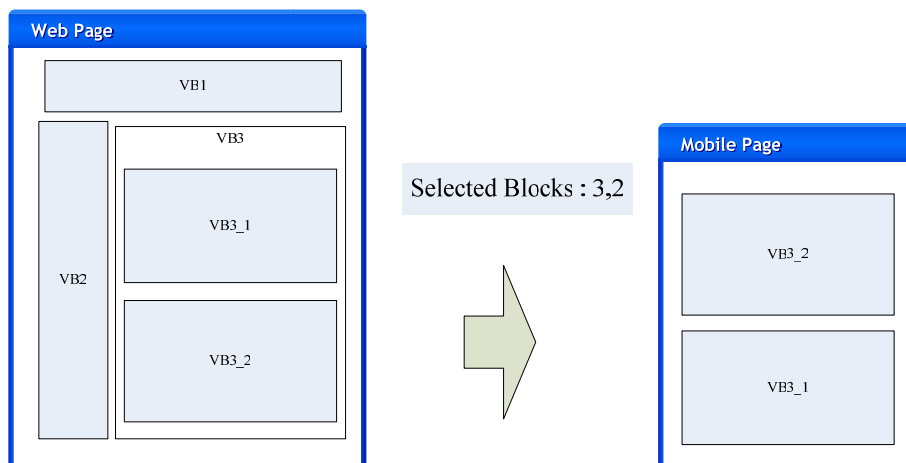


Figure 3-8 An example of Page Generation

Chapter 4 System Design

4.1. Overview

In order to achieve the objectives listed in section 1.3, we adopt the Page Web Page Tailoring Tool [23] and implement our Pattern-oriented scheme on this tool. With a Web-based authoring tool, users could easily author the Web pages through the Web Browser. By developing on Web platform, we can deploy the tool without installing other additional software. We adopt a Web proxy server for transformation functions. With this proxy-based deployment, we can construct the system in a short time and without any modification on the software of mobile clients or Web servers.

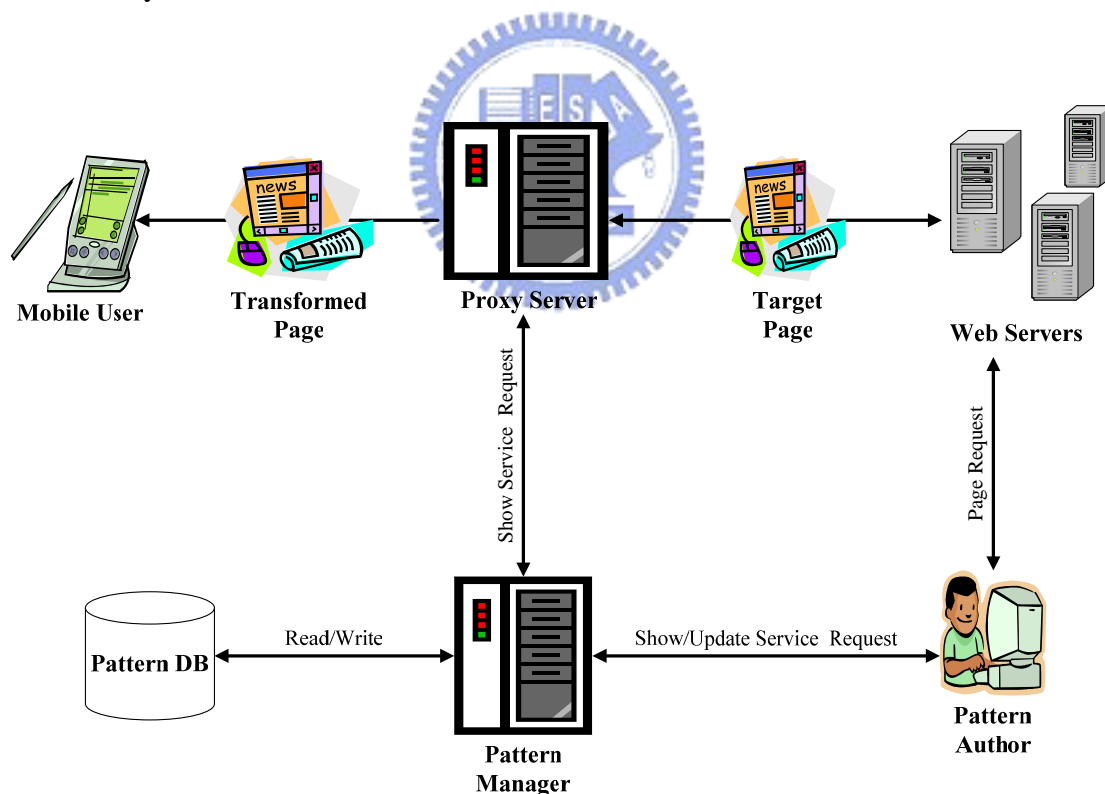


Figure 4-1 An overview of the system design

4.2. Web-based Authoring Tool

We provide a Web-based Authoring Tool for Web page authors. This tool enables authors to edit the Web page through the Web Browser, such as Internet Explorer and Mozilla Firefox. By using this tool, users specify which blocks of the Web page should be retained and what is the final arrangement of them. All the authoring result about this page and the pattern string will be saved in a remote database.

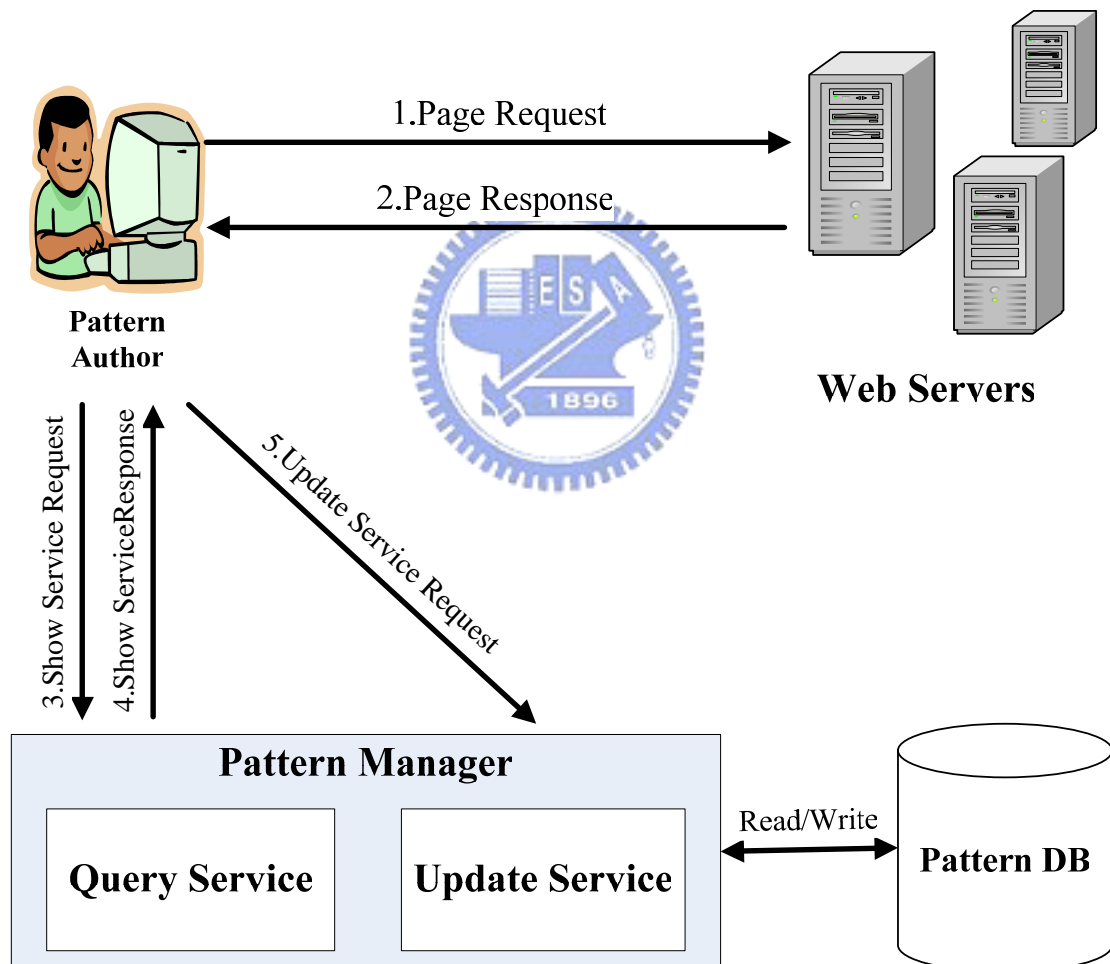


Figure 4-2 The design of Web-based Authoring Tool

4.3. Pattern Manager

4.3.1. Query Service

Pattern Manager provides this service for the other two components to retrieve page cuts, a pattern string and annotations of the Web page. The Web-based Authoring Tool would make a Web service call after launching. In addition, Mobile Proxy does the same thing whenever receiving a HTTP request.

For using this service, we must send the URL as the parameter to Pattern Manager. The Pattern Manager would use this value to retrieve and segment the target Web page, generate a pattern string, find the most similar pattern string in the database, and return the annotations.

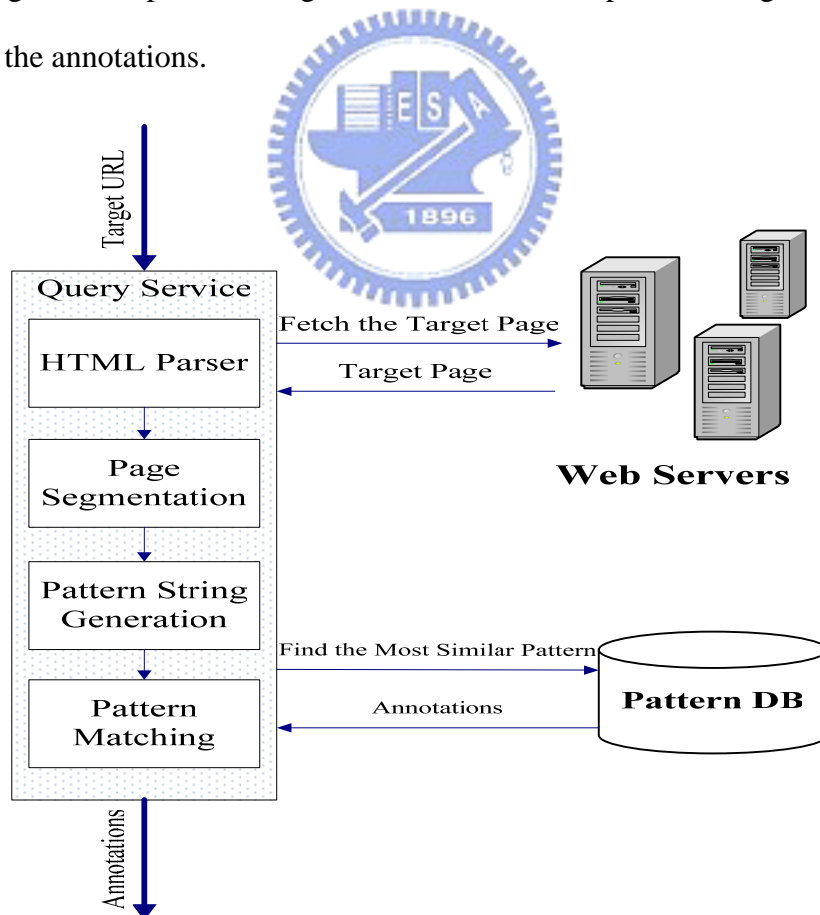


Figure 4-3 The design of Pattern Manager

4.3.2. Update Service

The Web-based Authoring Tool updates or adds the annotations of a Web page via the update service. When a user finishes the page authoring, the Web-based Authoring Tool will make this Web service call accompanied by a URL of the current page.

4.4. Mobile Proxy

As shown in Figure 4-4, the Mobile Proxy monitors every HTTP request and fetches the target page. Then it makes a Web service call (query service) with the request URL as the parameter to Pattern Manager. According to the returned annotations, the Mobile Proxy reconstructs the target Web page and responds to the mobile user.

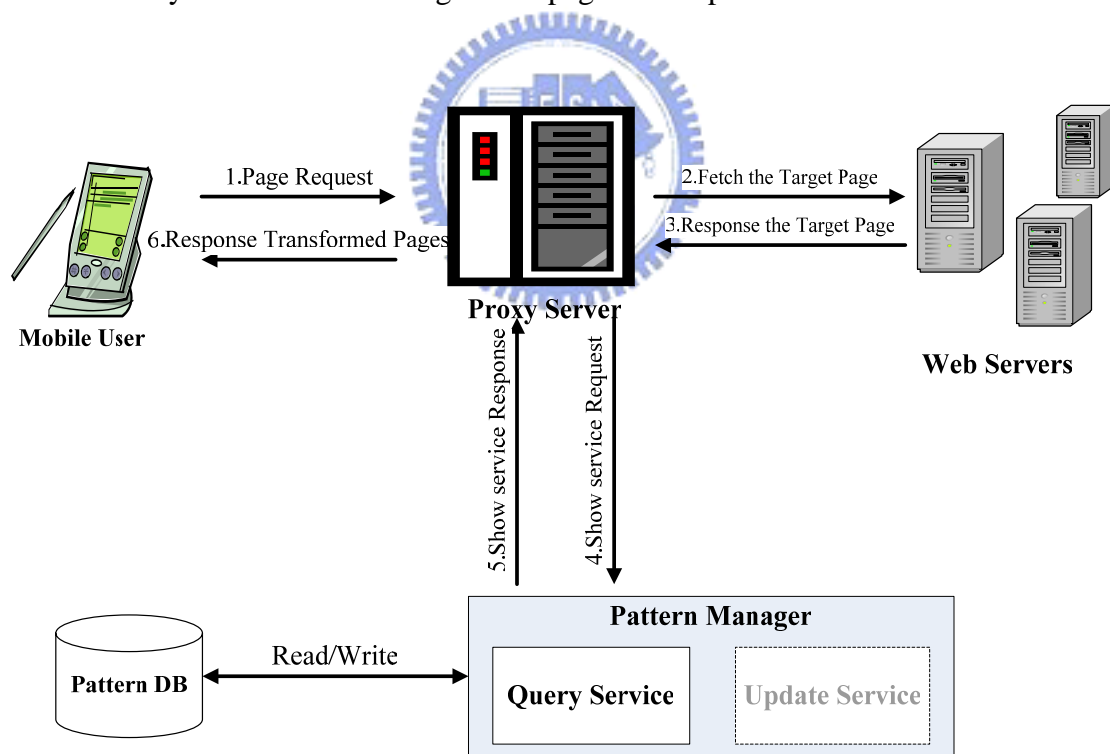


Figure 4-4 The design of Mobile Proxy

4.5. Summary

In this chapter, we introduce the design of the mobile page generating system. The design concepts and functions provided by each component are detailed in separate sections.



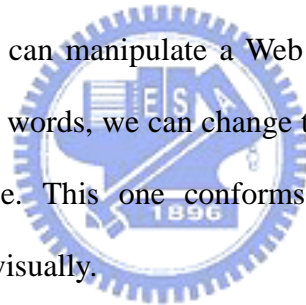
Chapter 5 Implementation

In this chapter, we will describe the implementation details of the three major components in our system.

5.1. Web-based Authoring Tool

We use JavaScript to implement the Web-based Authoring Tool, because of the following considerations:

1. JavaScript was designed to add interactivity to HTML pages.
2. JavaScript is the most popular scripting language on the internet, and works in all major Browsers, such as Internet Explorer, Mozilla, Firefox, Netscape, Opera
3. By using JavaScript, we can manipulate a Web page via the Document Object Model interface. In other words, we can change the appearance of a Web page to reflect the user's choice. This one conforms to the second objective, i.e. personalizing Web page visually.



5.2. Pattern Manager

In order to process the four major step of our scheme, we adopt Java Servlet Technology [24] to implement the Pattern Manager. Java Servlet is a module of Java code that runs in a server application to answer client requests. By using Java Servlet technology, the Pattern Manage has several advantages:

1. **Efficient.** With traditional CGI [25], a new process is started for each HTTP request. If the CGI program does a relatively fast operation, the overhead of starting the process can dominate the execution time. With servlets, the Java Virtual Machine [11] stays up, and each request is handled by a lightweight Java thread, not a heavyweight operating system process.

2. **Portable.** By adopting the J2EE platform [26], we can port our Pattern Manager on any application server which is J2EE compatible.
3. **Inexpensive.** There are a number of free or very inexpensive Web servers available that are good for "personal" use or low-volume Web sites. However, with the major exception of Apache, which is free, most commercial-quality Web servers are relatively expensive.

5.3. Mobile Proxy

Mobile Proxy in our system is based on Muffin. By implementing a filter of our own, we can modify the content of the Web page before it is sent back to the client according the annotations specified by the user.

The interfaces needed to be implemented in the filter logic class depend on what kind of filtering we want to do. In our case, two interfaces have been implemented in total: RequestFilter and ReplyFilter. The RequestFilter interface is implemented to filter requests before they go out to a server. And, on the contrary, the ReplyFilter is implemented to modify replies after a server responds.

5.4. Summary

In this chapter, we introduce the implementation details of the three major components of our mobile Web page generating system. The implementation concepts are detailed in separate sections.

Chapter 6 Evaluation

In this chapter, we evaluate our scheme in different aspects. A usability test will be taken and several samples will be shown in this section.

6.1. Usability Test

For proving the usability of our scheme, we select the CNN Website as target to test our system. Two test cases are adopted in the process. First, we choose the CNN Asia News as the Target page, author the page and save the result. After saving the result, we browse through a similar page – CNN Europe News and observe whether the authoring result does present correctly. Second, we browse through the CNN Asia News next day, and check the correctness of presentation. At the last, we also do the same test on CNN World/Africa and CNN World/Middle East.

6.1.1. Authoring Tool



As shown in Figure 6-1, we author the Web page and save the annotation. Figure 6-1(c) shows the authoring result which is stored in database. In Figure 6-2, the annotation of CNN Asia News page is applied to CNN Europe News page. Figure 6-3 and Figure 6-4 show the same process described above.



(A)

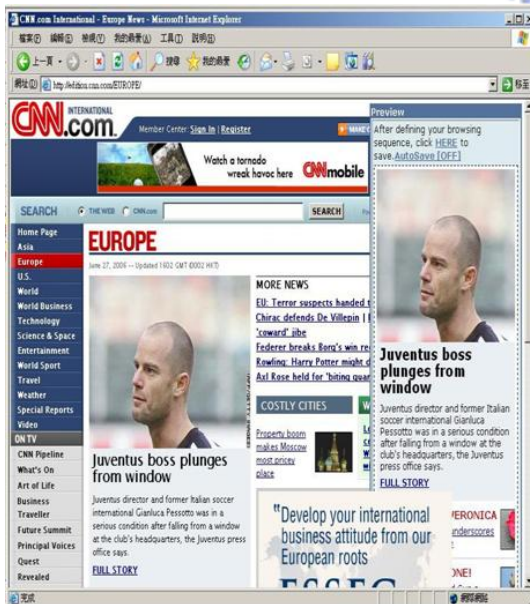


(B)

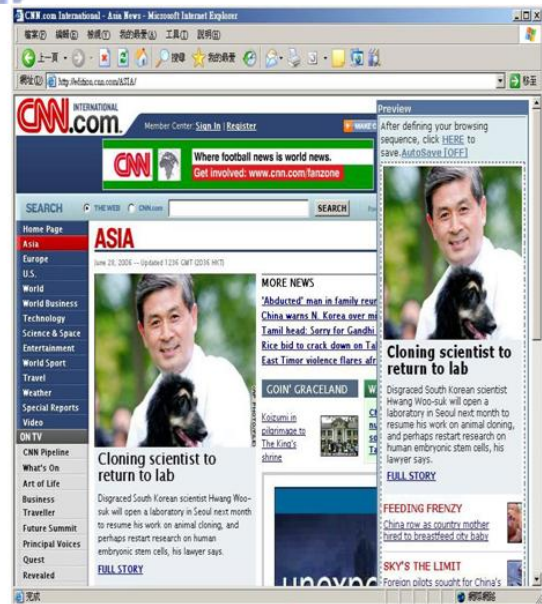
物件瀏覽器(B)	資料瀏覽器(D)	SQL編輯器(E)
name	url	node_position pattern
CNN.com International - Asia Ne	http://edition.cnn.com/ASIA/	8 01*-11011---1*10*-00**-1*111-----

(C)

Figure 6-1 The authoring example on CNN Asia News



(A)



(B)

Figure 6-2 The authoring example on CNN Asia and Europe News



(A)



(B)

name	url	node_position	pattern
CNN.com International - Asia Ne	http://edition.cnn.com/ASIA	8	01*-11011---1*10*-00**-*1*111-----
CNN.com - WORLD/africa	http://edition.cnn.com/WOR	8,9	01*-11011---**11111---1--**0000**000

(C)

Figure 6-3 The authoring example on CNN World/Africa News



(A)



(B)

Figure 6-4 The authoring example on CNN World/MIDDLE EAST and CNN World/Africa News

6.1.2. Mobile Proxy

As shown in Figure 6-5, we browse the CNN Asia and CNN Europe Web pages via the Mobile Proxy. Figure 6-6 shows the result page while browsing the CNN World/Africa and CNN World/Middle East Web pages through the Mobile Proxy.

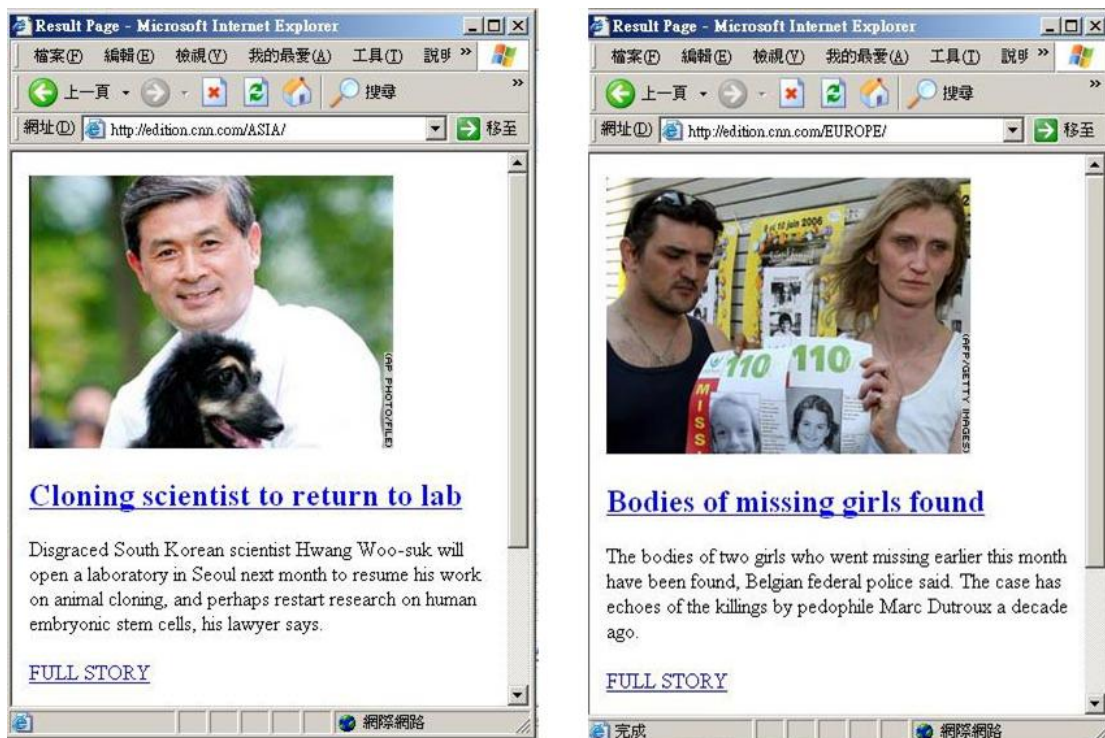
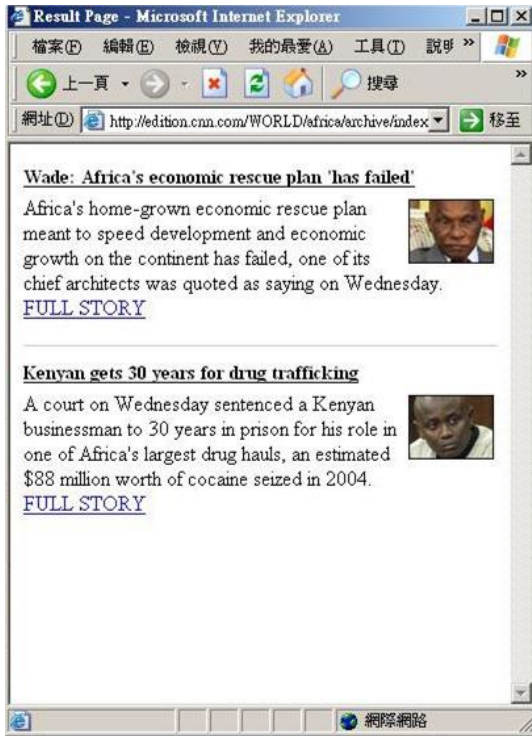
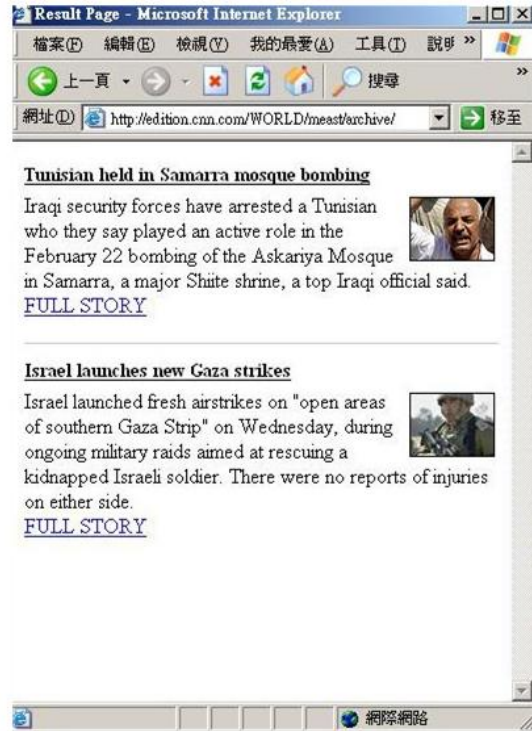


Figure 6-5 Browsing the CNN Asia and Europe News through Mobile Proxy



(A)



(B)

Figure 6-6 Browsing the World/Africa and CNN World/Middle News through Mobile Proxy



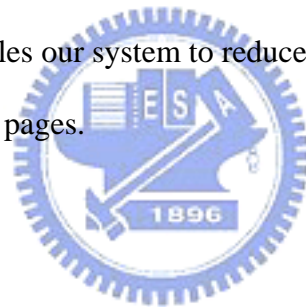
6.2. Summary

In this chapter, we introduce the evaluation details of our mobile Web page generating system. Several evaluation examples are described in this chapter.

Chapter 7 Conclusion

7.1. Conclusion

We have introduced our solution for mobile page generation in preceding chapters. Our scheme provides an easy and fast solution for mobile proxy construction. The Web-based authoring tool lets authors complete the authoring work easily. By using this tool, authors can customize the page to their favorite layout disposition. By employing the pattern manager, we reduce the authoring efforts. They needn't author every page on their Web application. The similar web pages can use the same authoring result. Therefore, a mobile proxy can be deployed in a short period. The proxy-based deployment enables our system to reduce the data transmission and prove the presentation speed of Web pages.



7.2. Future Work

There are several improvements available for our scheme. First, the present pattern matching method is comparing the pattern strings from beginning. Although we can find many type of similar Web Page by adopting this method. There are several types which are unable discovered. A finer matching method should be proposed to improve this defect.

Second, the current authoring tool only enable user to decides if the content blocks present on the result page. We will provide several transform heuristics, such as image reduction transform and first Sentence elision transform. Users can select these heuristic on content blocks.

Third, a user preferences function will be combined with the Pattern Manager in the future. Users can customize their own mobile pages and save these preferences in

database. Our System will deliver the custom-made mobile pages for the specific users.

Finally, we will integrate the PUML [27] technology into the Page Generation to support various mobile devices in the future. With the PUML, we can generate mobile pages according to the capabilities of each mobile device.



Chapter 8 Reference

- [1] W3C. Document Object Model (DOM), <http://www.w3.org/DOM/>
- [2] W3C. HTML 4.01 Specification, <http://www.w3.org/TR/html4/>
- [3] Ajax: A New Approach to Web Applications,
<http://www.adaptivepath.com/publications/essays/archives/000385.php>
- [4] Script.aculo.us JavaScript framework, <http://script.aculo.us/>
- [5] W3C. XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition), <http://www.w3.org/TR/xhtml1/>
- [6] W3C. Cascading Style Sheets, level 1, <http://www.w3.org/TR/REC-CSS1>
- [7] W3C. Extensible Markup Language (XML) 1.0 (Third Edition),
<http://www.w3.org/TR/REC-xml/>
- [8] W3C. XSL Transformations (XSLT) Version 1.0. <http://www.w3.org/TR/xslt>
- [9] Muffin World Wide Web Filtering System, <http://muffin.doit.org/>
- [10] The GNU Operating System, <http://www.gnu.org/>
- [11] Java Technology, <http://java.sun.com>
- [12] M. Lai and D. Wong. Slicing tree is a complete floorplan representation. In DATE '01: Proceedings of the conference on Design, automation and test in Europe, pages 228–232.
- [13] T. Bickmore, A. Girgensohn, and J.W. Sullivan, Web Page Filtering and Reauthoring for Mobile Users, Computer J., vol. 42, no. 6, 1999, pp. 534-546.
- [14] Y. Hwang, E. Seo, and J. Kim, WebAlchemist: A Web Transcoding System for Mobile Web Access in Handheld Devices, Proc. ITCOM, SPIE-The Int'l Soc. for Optical Eng., 2001, pp. 37-46.
- [15] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, VIPS: a vision-based page segmentation algorithm, Microsoft Technical Report, MSR-TR-2003-79, 2003.

- [16] J. Chen et al., Function-Based Object Model Towards Website Adaptation, Proc. 10th WWW Conf., ACM Press, 2001, pp. 587-596.
- [17] Yu Chen, Xing Xie, Wei-Ying Ma, Hong-Jiang Zhang, Adapting Web Pages for Small-Screen Devices, IEEE Internet Computing, vol. 09, no. 1, pp. 50-56, Jan/Feb, 2005.
- [18] WAP FORUM, <http://www.wapforum.org/>
- [19] Opera Mini™ simulator,
<http://www.opera.com/products/mobile/operamini/demo.dml>
- [20] CNN.com International, <http://edition.cnn.com/>
- [21] Google, <http://www.google.com/>
- [22] Google Mobile Content Proxy (still lack of an official name),
<http://www.google.com/gwt/n>
- [23] Chi-Yang Tsai, Shyan-Ming Yuan, “Web Page Tailoring Tool for Mobile Devices”, 國立交通大學電資學院碩士班論文，民國 95 年 6 月
- [24] Java Servlet Technology, <http://java.sun.com/products/servlet/>
- [25] Common Gateway Interface, <http://www.w3.org/CGI/>
- [26] Java EE At a Glance, <http://java.sun.com/javaee/>
- [27] Tzu-Han Kao, Yung-Yu Chen, Tsung-Han Tsai, Hung-Jen Chou, Wei-Hsuan Lin, Shyan-Ming Yuan, PUML and PGML: Device-independent UI and Logic Markup Languages on Small and Mobile Appliances Lecture Notes in Computer Science (LNCS) of Springer-Verlag, The 2005 IFIP International Conference on Embedded And Ubiquitous Computing (EUC-05), Nagasaki, Japan, 6-9 December 2005. (SCI)

Curriculum Vitae

Education

September 2004 ~ July 2006 National Chiao Tung University, M.S., Master Program of Computer Science

September 1993 ~ June 1997 National Chung Cheng University, B.S., Computer Science and Information Engineering

Experience

February 2004 – Now KYE Systems Corp., Sr. Engineer, MIS Div

April 2000 - February 2004 Acer Incorporated, Project Engineer, Smart Card Div

August 1999 - April 2000 VIA Technologies, Inc., Field Application Engineer, Department of Technology Support

