

SCALABLE CONTENT CREATION FOR MOBILE INTERNET

by

JIE ZHAO

B. Eng., Southwest Jiaotong University, China P. R., 1995

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 2003

© Jie Zhao, 2003

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical and Computer Engineering

The University of British Columbia
Vancouver, Canada

Date Oct. 6, 2003

Abstract

This thesis presents a new design of information architecture for the mobile Internet to achieve scalable content creation. The core originality of our design is a data structure for Web content encoding. With a unique tree hierarchy, this data structure enables the information architecture to take advantage of many open technologies for Internet content development, resulting in a dynamic adaptation of Web content based on variations in communication channel conditions and terminal device capabilities. Compared with proposals offered by other researchers, our design accomplishes content adaptation based on a more detailed analysis of the channel and device, an approach highly necessary for today's digital world of flourishing multimedia. As an empirical evaluation, a three-tier Web-browsing application is built to show that our solution can reach the adaptation objectives with an easy and cost-effective implementation. Thus, our design can be quickly employed in content creation for the mobile Internet. Furthermore, through extensions, our work can serve end-users through attractive mobile applications, while contributing significantly to the development of the mobile Internet.

Table of Contents

ABSTRACT	ii
TABLE OF CONTENTS	iii
LIST OF TABLES.....	vii
LIST OF FIGURES	viii
DEDICATION.....	x
ACKNOWLEDGMENTS	xi
CHAPTER I: INTRODUCTION.....	1
1.1 MOBILE INTERNET.....	1
1.2 NEED FOR NEW INFORMATION ARCHITECTURE.....	2
1.2.1 Internet Content	2
1.2.2 Information Architecture	3
1.3 THE PROBLEM AND VARIOUS SOLUTIONS	5
1.3.1 Problem Statement.....	5
1.3.2 Past Work	7
1.3.3 Thesis Objective	11
1.4 ORGANIZATION OF THESIS.....	12
CHAPTER II: SCALABLE CONTENT CREATION TECHNOLOGIES	13
2.1 XML	13
2.1.1 Overview	13
2.1.2 Distinctive Characteristics.....	14
2.1.3 Success on the Internet	16
2.1.4 XML Parser	17

2.1.5	Application of XML in Our Research	18
2.2	XSLT	19
2.2.1	XSL and XSLT	19
2.2.2	Application of XSLT in Our Research	20
2.3	XHTML	21
2.3.1	From HTML to XHTML	21
2.3.2	XHTML Modularization	22
2.3.3	XHTML-Family Markup Languages	25
2.3.4	Application of XHTML in Our Research	26
2.4	WAP	26
2.4.1	WAP and WAP Forum	26
2.4.2	WAP 2.0	27
2.4.3	Application of WAP in Our Research	28
2.5	JAVA	29
2.5.1	Java Programming Language and Platform	29
2.5.2	Application of Java in Our Research	31
2.6	CHANNEL DETECTION	35
2.6.1	Channel Detection on the Internet	35
2.6.2	Channel Detection on the Wireless Segment	36
2.6.3	Proposal of Channel Detection for a Combined Wired and Wireless Channel	37
2.7	MOBILE DEVICE DETECTION	38
2.7.1	Device Diversity	38
2.7.2	Device Detection Techniques	39

2.7.3	J2ME Profile.....	42
2.8	SUMMARY	45
CHAPTER III: A NEW DATA STRUCTURE FOR CONTENT ENCODING		47
3.1	OVERVIEW OF APPROACH.....	47
3.2	A NEW DATA STRUCTURE FOR CONTENT ENCODING	47
3.2.1	Decomposition of Web Content	48
3.2.2	Alternate Content Element	50
3.2.3	Adaptation Rules	55
3.2.4	A New Data Structure for Content Encoding.....	57
3.2.5	An “Image” Example.....	58
3.3	A SIMPLIFIED DATA STRUCTURE AS A PROOF OF CONCEPT.....	64
3.3.1	A Simplified Data Structure	65
3.3.2	Adaptation to the Environment.....	66
3.3.3	The Processing of the Simplified Data Structure	69
3.4	NOVELTY AND ADVANTAGES	71
3.5	SUMMARY	71
CHAPTER IV: A NEW INFORMATION ARCHITECTURE FOR CONTENT CREATION		73
4.1	HIGH-LEVEL DESIGN OF INFORMATION ARCHITECTURE	73
4.2	EXTENSION TO ADDRESS PERSONALIZATION ISSUES	79
4.3	NOVELTY AND ADVANTAGES	81
4.4	SUMMARY	81
CHAPTER V: EMPIRICAL EVALUATION.....		83

5.1	THREE-TIER WEB APPLICATION MODEL.....	84
5.2	“MOVIE” WEB-BROWSING APPLICATION	84
5.2.1	Network Configuration.....	84
5.2.2	Database Design	87
5.2.3	Java Servlet Design	94
5.3	RESULTS AND ANALYSIS	98
5.4	DISCUSSIONS AND FUTURE WORK.....	107
5.4.1	Improvements to Our Information Architecture.....	108
5.4.2	Extensions to Our Information Architecture	109
5.5	SUMMARY	110
CHAPTER VI: CONCLUSION		112
REFERENCES		113
APPENDIX		122
LIST OF ABBREVIATIONS AND ACRONYMS		122

List of Tables

Table II-1: Core Modules of XHTML.....	23
Table II-2: Structure Module.....	24
Table II-3: Text Module	24
Table II-4: HyperText Module	24
Table II-5: List Module	24
Table II-6: Toshiba A1304T, a 3G Handset (Partial).....	41
Table II-7: MIDP 2.0 on CLDC 1.0	44
Table III-1: Content Elements of “Movie.xml”	50
Table III-2: Content Element-to-Level Conversion (Partial)	67
Table III-3: Environment-to-Level Conversion (Partial)	69
Table V-1: “Movie” Table (Partial).....	89
Table V-2: “Content_Level_Conversion” Table (Partial).....	90
Table V-3: “CLC” Query (Partial)	91
Table V-4: “Env_Level_Conversion” Table (Partial)	92
Table V-5: “Env” Table.....	93
Table V-6: “ELC” Query.....	93

List of Figures

Figure I-1: Scalable Information Architecture	8
Figure II-1: "Movie.xml," an XML Document	14
Figure II-2: The Tree Structure of "Movie.xml"	16
Figure II-3: XSLT Transformation.....	20
Figure II-4: Architecture of XHTML Modularization	25
Figure II-5: WAP 2.0 Programming Model	28
Figure II-6: WAP's Optional Proxy Model Supports Network-Based Optimizations	28
Figure II-7: Java's "Write Once, Run Anywhere"	30
Figure II-8: Java Platform.....	30
Figure II-9: Java 2 SDK, Standard Edition Version 1.3.....	32
Figure II-10: Proposed Channel Detection Framework	38
Figure II-11: J2ME Configuration and Profile.....	43
Figure III-1: Many Alternate Elements for an Original Element.....	53
Figure III-2: Multiple Levels of Alternate Elements for an Original Element.....	55
Figure III-3: Adaptation Rules Built into Data Structure	56
Figure III-4: A New Data Structure for Content Encoding.....	58
Figure III-5: An Image Example (Part 1, Level-1 Processing)	59
Figure III-6: An Image Example (Part 2, Leve-2 Processing)	60
Figure III-7: A Simplified Data Structure for Web Content Element	66
Figure III-8: Adaptation Rule Implemented by Element Type and Element Attributes	66
Figure IV-1: High-Level Design for the Information Architecture.....	74
Figure IV-2: Step C, Device Detection	75

Figure IV-3: Step D, Channel Detection	75
Figure IV-4: Step E, Level Processing	76
Figure IV-5: Step F, Data Structure Processing	76
Figure IV-6: Step G, XML Production.....	77
Figure IV-7: Step H, XSLT Transformation	77
Figure IV-8: Information Architecture, with Personalization Scalability	80
Figure V-1: Three-tier Application and Network Configuration of “Movie” Application	85
Figure V-2: Sequence Diagram of “Movie Servlet”	95
Figure V-3: Test Case 1, Content at Level 0 (Partial View)	100
Figure V-4: Test Case 2, Content at Level 2	102
Figure V-5: An Image as Alternate Content Element for a Video File.....	102
Figure V-6: Test Case 3, Content at Level 3	104
Figure V-7: A Small Image as Alternate Content Element for a Video File and an Audio File	104
Figure V-8: Test Case 4, Content at Level 4	105
Figure V-9: A Small Image as Alternate Content Element for a Video File.....	105
Figure V-10: Test Case 5, Content at Level 5.....	106
Figure V-11: A Short Text as Alternate Content Element for a Video File.....	106

Dedication

To my parents, for their unending support to my intellectual development and lifelong appreciation of knowledge.

Acknowledgments

I wish to take this opportunity to express my gratitude to Professor Victor C. M. Leung and Professor Matthew J. Yedlin of the Department of Electrical and Computer Engineering of the University of British Columbia for their admirable guidance and firm support in my pursuit of a Master's degree during the past two years. Both supervisors have created an environment of trust, encouragement, and expertise for me throughout the research process.

Dr. Lee Iverson, Dr. Son Vuong, Dr. Qiong Chen, and Hui Wang, all experts in Computer Science and Computer Engineering, and my father offered constructive suggestions and expressed constant interest in my thesis research. My colleagues Zhanping Ying, Ying Luo, and Li Ma of the University of British Columbia and Huayi Wang of Carleton University shared with me their knowledge in mobile communication. Kristin Loheyde and Gary Barclay helped me immensely with my thesis writing. Their generous assistance is appreciated and will remain deep in my memory.

This work is supported by grants from TELUS Mobility and the Advanced Systems Institute of British Columbia and by the Canadian Natural Sciences and Engineering Research Council under Grant CRD 247855-01.

Chapter I: Introduction

1.1 Mobile Internet

The enormous success of the Internet and wireless communication technologies in the past decade has stimulated a global interest in extending the World Wide Web (WWW) to the mobile world. On the one hand, the Internet has been omnipresent in everyday life and has fundamentally changed our ways of accessing information, conducting business, and entertaining ourselves. On the other hand, with the rapid development and deployment of wireless personal communication systems, a rapidly expanding worldwide mobile community has emerged, demanding information, entertainment, and business services wherever and whenever. The mobile Internet is the technology to make this come true [1].

Recent developments in wireless communication technologies have made the mobile Internet possible. For example, from the network point of view, current Second Generation (2G) digital cellular networks are capable of providing wireless access to the Internet, with the Wireless Application Protocol¹ (WAP) and i-mode² as pioneering examples. Furthermore, the Third Generation (3G) networks are being developed to offer a seamless mobile telecommunication services worldwide, including wideband multimedia data services [2; 3]. Another example, from the terminal point of view, mobile handheld devices with Web

¹ What is WAP?, 2002, Open Mobile Alliance, 16 July 2003

<<http://www.wapforum.org/what/index.htm>>.

² What is i-mode?, 29 Jul. 2003, NTT DoCoMo, 16 July 2003

<<http://www.nttdocomo.com/home.html>>.

accessing capabilities are already widely available on the market. The Meta Group has predicted that within the next few years, more than half of Internet accesses will be made through wireless portable terminals instead of personal computers or workstations [4].

The mobile Internet is still a young technology facing many challenges [5].

Nevertheless, with its benefits over the traditional Internet, the technical potential, and the demand from the general public, the mobile Internet will force its way into a multi-billion dollar business, and ultimately escalate the world into a new information age where human beings enjoy a rich variety of information with a completely personalized experience.

1.2 Need for New Information Architecture

1.2.1 Internet Content

Internet content or, simply speaking, a Web page, is a varied collection of information materials such as texts, images, tables, audio files, video files, and so on, that is organized in a hypertext format. HyperText is the organization of information units into connected links. For example, the Web contains an enormous amount of information connected by an enormous number of hypertext links. It is said that on the Internet, “content is king” because content browsing is not only the most commonly used Internet application but also the enabler of many other Internet applications such as e-business, portals, online gaming, and many more. “Good content” is defined as a collection of abundant, well-presented subject matter organized for interesting and convenient access [6]. Thus, not only is the subject material important, but also the user experience when interacting with the content. An example of the importance of computer-human interaction is the increasing dominance of multimedia information on the Internet as a result of the public’s demand for “good content.” Defined as “more than one concurrent presentation medium,” multimedia presentations

combine some or all of the following media types: text, sound, still and/or animated graphic images, video images, and multiple display areas [7]. The rapid advancement in digital encoding/decoding technologies allow for even more new media types to be developed to meet the needs of an increasingly more critical and Internet savoring user population.

Similarly, the mobile Internet relies heavily on good content as well. However, the mobile Internet has its own challenges not shared by the traditional wired Internet. The success of turning many advanced technical functionalities, such as browsing, gaming, entertainment, and portals, into one-button- or one-click-away amenities on the mobile terminals, is not as simple as on fixed terminals such as personal computers which are equipped with advanced display and navigation capabilities. The success of mobile Internet applications and services depend heavily on how easy it is to browse and locate the services and obtain relevant information using the mobile communication technologies. Clearly, a good scheme of content creation and utilization will ease the acquisition of information on mobile terminals and in turn promote the mobile Internet business. This is the goal that the information architecture aims to achieve.

1.2.2 Information Architecture

In this thesis, the information architecture of the Internet and the mobile Internet is concerned with how the original content is encoded and stored in the Web content server, how it is decoded and used at the client, and how it is communicated via the communication channel . In the example of a Web-browsing application, the source content is encoded in HyperText Markup

Language³ (HTML) in the Web server and is parsed and displayed by the Web browser at the client. The communication between the client and server is generally based on HyperText Transfer Protocol⁴ (HTTP). The client initiates an HTTP request, and the server processes the request, generates the Web content, and sends it back to the client via the HTTP response.

Despite many similarities with the Internet, the mobile Internet is confronted with unique problems owing to the properties of the wireless communication system. First of all, the wireless channel is notoriously known for its narrow bandwidth (BW), long latency, and high transmission errors. Secondly, mobile terminals are severely restricted in terms of processing power and information display capabilities. Last, but not least, the mobility of the terminal, the large disparity among the terminals themselves, and personalization issues, are additional challenges that the traditional Internet has not encountered.

The information architecture that is currently deployed on the Internet was not designed to accommodate these restrictions in the wireless communication system. Internet content is encoded in HTML which only works well with Web browsers that require strong computing resources. In addition, HTML is a markup language that mixes content with display instructions. (A “markup,” or “tag,” is a sequence of characters and/or symbols that a document author inserts at certain places in a text processing file to indicate how the file

³ Related W3C Work HyperText Markup Language (HTML) Home Page, Jul. 2003, W3C, 16 July 2003 <<http://www.w3.org/MarkUp/>>.

⁴ HTTP - HyperText Transfer Protocol, 16 Apr. 2003, W3C, 16 July 2003 <<http://www.w3.org/Protocols/>>.

should look when it is displayed or printed or to specify the document's logical structure. A markup language is a language that defines a set of markups.) These characteristics make HTML unsuitable for the mobile Internet and have caused a lack of contents suitable for mobile terminals. In fact, this is by far the biggest obstacle that hinders the development of the mobile Internet [9; 10]. Hence, it is imperative to create a new information architecture for the mobile Internet.

1.3 The Problem and Various Solutions

1.3.1 Problem Statement

WAP was the first and the dominant technology to allow mobile Internet applications on mobile handsets. Since its advent in 1997, WAP has gone through many developments. Within a few years, WAP-based Internet services have evolved from simple news-headline-type browsing into a large variety of applications, such as mobile-commerce, mobile-banking, enterprise access, location-smart services, entertainment, and messaging [11]. There are a lot of studies on the deployment of WAP in various industries for mobile applications such as mobile games, robotic control, monitoring systems, health and telemedicine, and so forth [12–21].

The information architecture of WAP mainly includes the Wireless Markup Language⁵ (WML) as the encoding language of Web content and the micro-browser installed on the terminal that displays the WML content [22; 23]. Although it did give rise to some mobile Internet applications, WAP was soon found unfavorable in real implementation.

⁵ Wireless Markup Language Version 2 Specification, WAP-238-WML-20010911-a, Jul. 2001, WAP Forum, 16 July 2003 <<http://www.wapforum.org/what/technical.htm>>.

The most severe weakness of the WAP architecture is that the micro-browser can only use content encoded in WML. This means that all of the content based on HTML that is already available on the Internet has to be completely re-encoded into WML before it can be displayed on the mobile terminal. Needless to say, any new content must be created in two versions as well, one in HTML, the other in WML.

To avoid having to write multiple versions of the same content, many kinds of middleware were developed to transcode HTML into WML. The techniques used are called content conversion techniques. They include HTML simplification, text summarizing, Web document fragmentation, image transcoding, and so forth [24; 25]. These techniques can be found in use in several commercial content conversion products, such as IBM's WebSphere Transcoding Publisher⁶, Spyglass' Spyglass Prism⁷, Jataayu Software's ZAP2WAP⁸, and Argogroup's WAP Tool⁹. The major advantage of using these content converters is the speed with which the Web content documents are brought to the market. However, this is

⁶ WebSphere Transcoding Publisher, Feb. 2001, IBM, 16 July 2003 <http://www3.ibm.com/software/pervasive/products/mobile_sols/transcoding_publisher.shtml>.

⁷ Spyglass Prism 3.3.0 Fact Sheet, May 2001, OpenTV, 16 July 2003 <<http://products.datamation.com/wireless/dt/916162661.html>>.

⁸ Jataayu Launches ZAP2WAP – HTML to WML Converter, Aug. 2000, Jataayu Software, 16 July 2003 <<http://www.jataayusoft.com/08-11-2000.htm>>.

⁹ WAP Tool V1.1, Oct. 2001, Argogroup, 16 July 2003 <<http://www.argogroup.com/products/waptool/>>.

only a temporary solution for the mobile Internet application development due to some obvious limitations: These tools often need labor intensive author intervention, create additional processing for the existing content, and result in a large loss of information [26]. (To write configuration rules can be almost as burdensome as to create new Web sites specific to wireless devices due to difficulty in setting up the transcoding process [27].) Moreover, the conversion approach is not a scalable one. Considering the rapid proliferation of wireless devices other than WML compatible mobile phones, for example, personal digital assistants (PDA), TV set-top boxes, smart phones, et cetera, if we were to stick to the content conversion scheme, we would have to keep developing converters.

As the mobile Internet is playing a more and more important role, there is a strong incentive to create a new information architecture which can generate content that is scalable for a wide variety of clients with diverse capabilities in different network environments. The information architecture should be able to dynamically adapt the Web content according to the device and network. Moreover, taking into account the ever-increasing multimedia content and applications on the Web, the information architecture is expected to be highly scalable in terms of supporting multimedia and any media types that will arise in the future.

1.3.2 Past Work

A fundamental requirement for such a scalable information architecture is “one source content, multiple presentations,” as shown in Figure I-1. Guided by this principle, a unified content encoding scheme must be present, so that the content writer only needs to write “one source content.” In addition, some adaptation functions should be developed to “transform” the source content into “multiple presentations” to suit the variations in the communication channel, the terminal capability, the user preference, and the user’s location.

Furthermore, the channel, device, and user ought to output their information relevant to content processing to the adaptation functions.

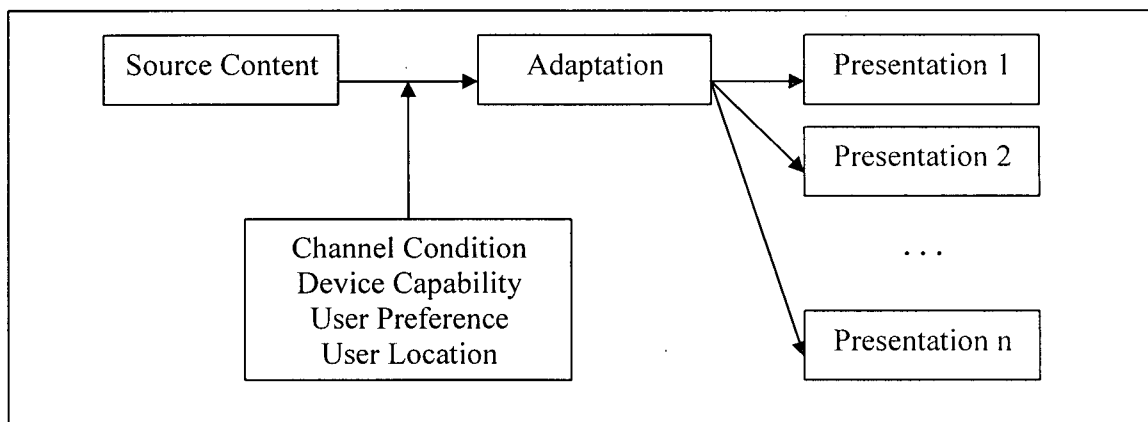


Figure I-1: Scalable Information Architecture

One technology that can be used in information architecture for scalable content creation is Extensible Markup Language¹⁰ (XML), which deals with scalability issues for multiple types of clients. The most distinctive attribute of XML is the separation of content and presentation. This makes XML ideal in providing a coherent platform to support the information architecture of the mobile Internet. Using XML, numerous solutions of the information architecture for the scalable content creation for the mobile Internet have been proposed in the past few years. They can be categorized into four groups, namely, the multiple Extensible Stylesheet Language Transformations¹¹ (XSLT) transformation, new

¹⁰ Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, Oct. 2000, W3C, 16 July 2003 <<http://www.w3.org/TR/REC-xml>>.

¹¹ XSL Transformation (XSLT) Version 1.0, W3C Recommendation, Nov. 1999, W3C, 16 July 2003 <<http://www.w3.org/TR/xslt>>.

markup languages, new browsing experience, and micro-browser enhancement. These solutions will be described in the following four paragraphs.

The multiple XSLT transformation is by far the most popular approach. It is illustrated in many publications as a fast means of dynamic content creation for multiple terminals [4; 27; 28]. XSLT has the power of transforming an XML document into any text format. Using this advantage, a single copy of XML-encoded content can be transformed into different formats for suitable presentations on corresponding end devices, such as HTML for personal computers or WML for WAP phones. The advantage of this approach is its simplicity. All the content provider needs to do is to encode the source contents in XML and write a number of XSLT stylesheets to perform the transformation. Nevertheless, the extent of scalability is purely dependent on the number of XSLT stylesheets. Thus, it is difficult to cope with the large and ever-increasing variations among the mobile terminals.

The second most popular solution is to build new markup languages that encompass application-specific and presentation-specific information for various devices into the content creation phase. A typical example of this is presented in Michael A. Blackstocks's Master's thesis "Markup for Transformation to Mobile Device Presentation Markups Using XML" where the author invented a new markup language, "Mobile Markup," for developing simple Web applications for HTML, WML, and Voice Extensible Markup Language¹² (VoiceXML).

¹² Voice Extensible Markup Language (VoiceXML) Version 2.0, W3C Candidate Recommendation, Feb. 2003, W3C, 16 July 2003 <<http://www.w3.org/TR/2003/CR-voicexml20-20030220/>>.

The idea is to separate application objects, purpose, and presentation into sections within the markup and provides an abstraction for the purpose of an HTML Web page, a WML deck, or a VoiceXML dialog to accomplish its goals [29].

The third solution for scalable content creation assumes a brand new browsing experience for users of small, mobile terminals. Bill N. Schilit, Jonathan Trevor, David M. Hilbert, and Tzu Khiau Koh, in "Web Interaction Using Very Small Internet Devices," propose that the correct Web-browsing model for small Internet devices is "navigation + action," not "browsing + action" which is suitable for desktop computers [30]. In this model, the information on a Web page is processed by a middleware into "links" and is displayed in a folder-like structure. Associated with each link are services such as "email content," "read content," "print content," and "fax content" which the end-user can act upon. The end-user navigates through the folders to find a link of interest and selects the action he or she wishes to perform. This new browsing model best supports directed browsing in which case users follow a series of links to reach the specific content. However, as the authors admit, casual browsing does not work well because if a user does not have a clear target when surfing the Web, he or she generally cannot know which route to follow in order to reach the intended content.

The fourth solution takes advantage of the development of the micro-browser on the terminal. There are quite a number of enhancements that have been done to make mobile terminals more powerful in terms of content assumption [31]. The incorporation of XHTML, the support of sporadic-access using Synchronization Markup Language (SyncML) [32] and WAP Data Synchronization (WAP-Sync) [33] and the use of voice via VoiceXML can significantly improve the user experience of mobile Internet applications. Unlike the other

three solutions, this solution does not require network support, which makes things simple. Nonetheless, since the wireless terminals are inevitably restricted in their display size, processing power, and battery capacity, the capabilities of the micro-browser can only have a very limited impact on the scalable content creation. Thus, this solution is often combined with the others to reach a good scheme for scalable information architecture.

To summarize, the four solutions have achieved content scalability over multiple devices to a certain extent, but significantly minimized the variations in the terminals and generally ignored the condition of the wireless communication channel. In these solutions, the terminals are roughly categorized based on the markup languages used for presentations. A more relevant classifying scheme would be the hardware and software specifics of a terminal which truly determine its processing and displaying capability. Due to the mushrooming multimedia content on the Web, which is highly sensitive to channel conditions, it is imperative that the channel be included into the solutions.

1.3.3 Thesis Objective

To promote mobile Internet services, this thesis aims to propose a new information architecture to facilitate content scalability over a wide range of environment factors which include communication channel conditions, terminal device capabilities, and personalization issues. This multi-dimensional consideration demands a great many presentations for a single original content. To this end, like the current solutions available, the contents stored in the server will include all the information that the most capable client is able to consume under the most favorable channel conditions. However, unlike those solutions, which are generally concerned about the development of adaptation functions or new languages, the proposed focus of our research on the information architecture is how to structure the source

content in an intelligent way so as to achieve a high level of scalability. Given the growing proliferation of multimedia content on the Web, requirements for this goal include a high scalability over media types, minimal information loss during the adaptation, and the potential of providing Quality of Service (QoS) support. By looking at the formation of the data structure and analyzing the feasibility of its implementation, the viability and commercialability of this new and exciting solution will become evident.

1.4 Organization of Thesis

The reminder of the thesis is structured as follows: Chapter II gives a background of the technologies used in scalable content creation. Chapter III presents the specific details of a new data structure for source content encoding. Chapter IV describes the novel information architecture based on our newly invented data structure. Chapter V explains an implementation of the information architecture and test results followed by discussions. Finally, Chapter VI summarizes our research and concludes the thesis.

Chapter II: Scalable Content Creation Technologies

The mobile Internet has been evolving quickly, embracing many new technologies throughout its progress. The conception of a novel information architecture for scalable content creation is no different. This chapter will present the fundamentals of the core technologies that are used for developing scalable content for the mobile Internet. Through an analysis of the characteristics and exploration of potential uses of these core technologies, we can explain how they can be applied to scalable content creation.

2.1 XML

2.1.1 Overview

One of the most recent Internet technologies is XML, an extremely simple text format markup language. It is derived from Standard Generalized Markup Language¹ (SGML), a meta-language that defines how a document can be described in terms of its logical structure without the consideration of how it should be displayed [34]. XML was initiated by the XML Working Group of the World Wide Web Consortium² (W3C) in 1996. The latest specification is XML 1.0 Second Edition, created in October, 1998. XML was originally designed for large-scale electronic publishing, but with a goal “to enable generic

¹ ISO 8879:1986, Information Processing -- Text and Office Systems - Standard Generalized Markup Language (SGML), 1986, Intl. Organization for Standardization, 16 July 2003 <<http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=16387&ICS1=&ICS2=&ICS3=>>.

² The World Wide Web Consortium site is at <<http://www.w3.org/>>.

SGML to be served, received, and processed on the Web in the way that is now possible with HTML.” Since XML came into existence, W3C has been dedicated to making XML a straightforward tool to use over the Internet that is quick to develop, easy to process, and scalable over applications [35].

2.1.2 Distinctive Characteristics

One of the most outstanding characteristics of XML is the separation of content and presentation. Unlike HTML, XML simply structures a collection of data to store in a text format document. It does not deal with how the data looks. Below is an example of an XML document showing the structure-oriented characteristic of XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<movie>
  <movieTitle> Titanic </movieTitle>
  <movieDescription> James Cameron's "Titanic" is an epic, action-packed romance
    set against the ill-fated maiden voyage of the R.M.S. </movieDescription>
  <leadingActor>
    <leadingActorName>Leonardo DiCaprio</leadingActorName>
    <leadingActorLink>www.leonardodicaprio.com </leadingActorLink>
    <leadingActorPhoto>leodPicGif.gif</leadingActorPhoto>
    <leadingActorBio> Leonardo Dicaprio is... </leadingActorBio>
  </leadingActor>
</movie>
```

Figure II-1: "Movie.xml," an XML Document

This “Movie.xml” describes the contents of a movie object, which includes several elements such as a movie title, a movie description, and a collection of information on the leading actor. The <leadingActor> block in turn contains several “children” elements such as the name of the actor, a link to the actor’s homepage, a photograph of the actor, and a biography. All the information is pure text wrapped in XML tags and organized in a hierarchical structure; however, having no tags which function as a specific presentation instruction, this XML document gives no clues about how its data will be presented.

Another significant characteristic of XML is that every XML document is a “tree.” Every data component in the XML is called an entity. The document begins with a “root,” or “document entity.” Underneath the root, there can be children entities including elements, comments, and/or processing instructions. Every entity, including the root, is delimited by start- and end-tags, nested properly within each other. A document satisfying this structural principle is said to be “well-formed” [36]. As a result, every XML document is a well-formed tree. Taking our “Movie.xml” in Figure II-1 as an example, it is easy to draw a tree representation of this document by a quick analysis of the markup tags in the document. The tree structure of “Movie.xml” is shown in Figure II-2. The root node of the tree represents the whole document. The leaves of the tree, such as “movieTitle,” “movieDescription,” and the children of “leadingActor,” contain all the real information of “Movie.xml.” The “well-formedness” of the XML tree helps to make the processing of an XML document simple.

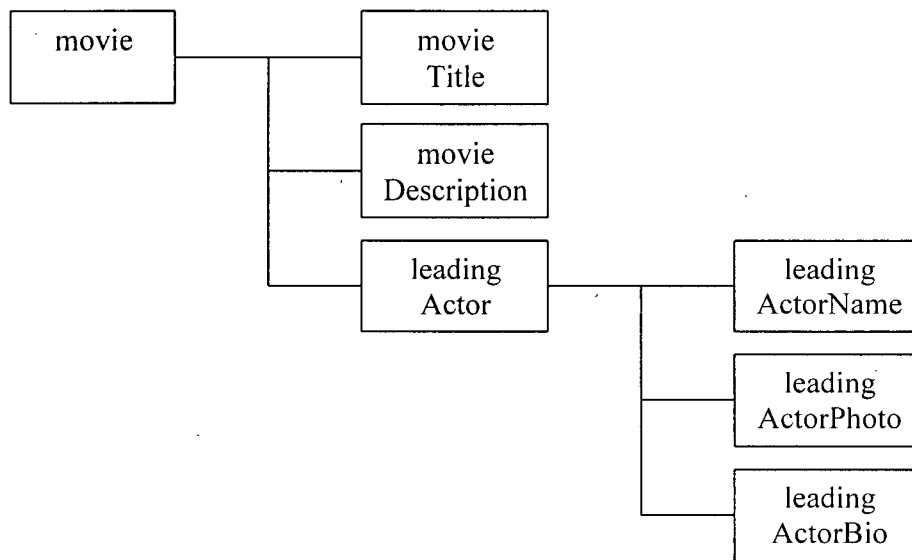


Figure II-2: The Tree Structure of “Movie.xml”

A third important characteristic of XML is that it is a highly extensible language due to the fact that XML allows the author to define markup tags as well as the document structure. Since there is no restriction as to what tags must be used nor how the document should be structured, it is very easy to create and extend an XML document based on the specific application the XML document is used for.

These three distinctions are among the characteristics that make XML an easy-to-use, remarkably scalable tool for data storage and transfer. With these advantages, XML can be used on any hardware platform, under any operating system, and in any software application that can process simple text.

2.1.3 Success on the Internet

The Web is a massive jungle of different computing hardware, software, and middleware. Such a setting is an ideal stage for XML to exhibit its power. The simplicity and flexibility of XML has gained rapid recognition in Web development. In the relatively short time since it was created, XML has been adopted by a large number of software and

middleware vendors and, as W3C points out, has become “the universal format for structured documents and data on the Web” [37]. Not only is XML overwhelmingly popular in the storage and transformation of Web content, but it has found extensive usage in Web applications for describing business logic, integration components, and even security policies [38]. Moreover, it has given rise to many new markup languages which support novel Web applications. One comprehensive example is Web Services³, a hotspot in Web development nowadays. Web Services is increasingly fortified by employing XML to standardize data formats, publish contents, and exchange information on the Internet. The Web Services Description Language⁴ (WSDL) is derived from XML and is the foundation of Web Services.

2.1.4 XML Parser

XML purely describes data. How that data is used is completely dependent on the individual user, who must write a program to send, receive, or display the data in the XML document. Before that happens, the XML document needs to be read into the computer’s memory so that the data can be made available for the user program to use. This process is done by the XML parser.

³ Web Services Activity, Aug. 2003, W3C, 16 July 2003 <<http://www.w3.org/2002/ws/>>.

⁴ Web Services Description Language (WSDL) 1.1, Mar. 2001, W3C, 16 July 2003 <<http://www.w3.org/TR/wsdl>>.

There are two major types of XML parsers, the Simple API for XML⁵ (SAX) and the Document Object Model⁶ (DOM). The SAX parser is an open source Application Programming Interface (API) that scans an XML document from start to end in a serial manner, and, upon encountering an element, sends event notifications to the program that is responsible for processing the element. The DOM parser is a W3C recommendation. It reads the whole XML document into the computer's memory and represents it in a tree structure. DOM API allows programs to act upon the elements by means of accessing and updating the nodes and leaves of the tree.

2.1.5 Application of XML in Our Research

Exemplifying the heterogeneity of the Web to its extreme, the mobile Internet should incorporate XML in its information architecture. We will use XML as the data holder of the source content. We will see how the tree structure of the XML document can lead to an elegant processing of the source content to achieve our desired scalability and dynamic adaptation of the Web content.

⁵ About SAX, 2002, SAX Official Web Site, 16 July 2003

<<http://www.saxproject.org/>>.

⁶ Document Object Model (DOM), Jun. 2002, W3C, 16 July 2003

<<http://www.w3.org/DOM/>>.

2.2 XSLT

2.2.1 XSL and XSLT

In our research of Web content creation, an XML-encoded content document must be transformed into other types of “displayable” formats, for example, HTML or WML. This conversion is performed by a family of languages called the Extensible Stylesheet Language⁷ (XSL). XSL is a suite of W3C recommendations developed by W3C’s XSL Working Group for defining XML document transformation and presentation. It consists of three individual languages: XSL Transformations (XSLT), XML Path Language⁸ (XPath) and XSL Formatting Objects⁹ (XSL-FO). We concern ourselves mostly with XSLT because it provides the essential functionality for document format transformation.

XSLT is a stylesheet language that transforms one XML document into another. A style sheet is a document that “describes how documents are presented on screens, in print, or perhaps how they are pronounced” [39]. Keeping in mind that an XML document is a tree, an XSLT stylesheet is a set a pattern-matching rules for converting a source tree into a result

⁷ The Extensible Stylesheet Language Family (XSL), Aug. 2003, W3C, 16 July 2003 <<http://www.w3.org/Style/XSL/>>.

⁸ XML Path Language (XPath) Version 1.0, W3C Recommendation, Nov. 1999, W3C, 16 July 2003 <<http://www.w3.org/TR/xpath>>.

⁹ Formatting Objects, Extensible Stylesheet Language (XSL) Version 1.0, W3C Recommendation, Oct. 2001, W3C, 16 July 2003 <<http://www.w3.org/TR/xsl/slice6.html#fo-section>>.

tree. The transformation process, as depicted in Figure II-2, is carried out by an XSLT processor that takes in the XML source and the stylesheet, traverses the source tree to find matching elements indicated in the stylesheet, and then instantiates templates, which become parts of the result tree according to the transformation rules. XSLT does not place any restriction on the templates or the instantiation, therefore the structure of the result document can be completely different from that of the source. During the transformation, elements in the source tree can be removed, edited, filtered, sorted, or rearranged, and new elements can be added arbitrarily. This flexibility gives XSLT its immense power in the field of data presentation.

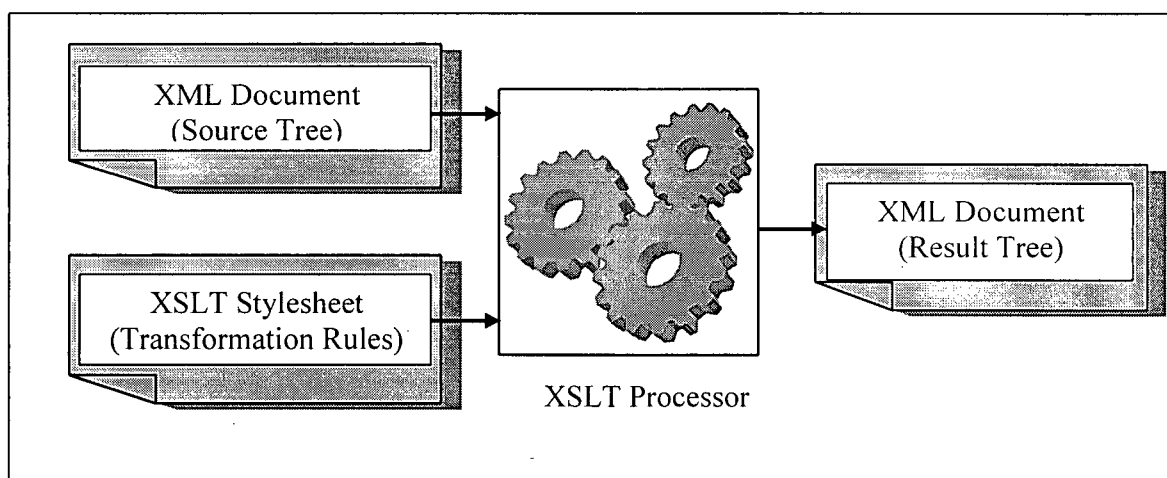


Figure II-3: XSLT Transformation

2.2.2 Application of XSLT in Our Research

Generally, XSLT is used whenever there is a need to display XML-encoded data. We will use XSLT in our research of scalable content creation to transform the XML-encoded content into presentable formats for various terminals, particularly in our prototype implementation to convert an XML document into an XHTML document.

2.3 XHTML

2.3.1 From HTML to XHTML

HTML is widely understood to be the working language for Internet content publishing. It is a set of structural and semantic tags which, when inserted in a text document, instruct the Web browser to display the content of the document. HTML was originally created for the transfer of scientific and technical documents for electronic publishing. Its markup tags were to help non-document specialists to author simple documents. Later, support for hypertext and multimedia were added into HTML. The adoption of HTML as the content authoring tool on the Web has made HTML extremely popular. Both HTML and browser technology have developed significantly over the past few years, bringing a richer user experience through the support of hypertext and multimedia.

Although HTML is non-proprietary, in order to compete for market share, Web browser vendors have introduced many optional elements for HTML to work with their highly specialized products. This flood of new elements has caused compatibility problems for HTML to be used on different platforms. For example, many end-users have had the experience of being able to see a piece of animation on a Web page when a certain type of Web browser is used, while a still image or nothing at all when another browser is used. As a result, Web content encoded in HTML is more and more dependent on the particular Web browser to achieve the maximum presentation effect. This dependency had greatly hindered the development of HTML, and severely restricted the distribution of Web content and applications, particularly in today's ever-increasingly heterogeneous computing environment.

As one solution, W3C initiated the Extensible HyperText Markup Language¹⁰ (XHTML) to replace HTML. The essential objective of XHTML is to achieve general user-agent inter-operability in order to overcome the compatibility problem of HTML. The first recommendation of XHTML, XHTML 1.0, was an incarnation of HTML4 in XML, and a “tidy” HTML which meets all the structural requirements of an XML document. As an XML application, XHTML can leverage all the benefits of XML, especially the virtue of being extensible and scalable. However, through modularization, XHTML has become more than just XML-conforming HTML. In fact, it is “a family of current and future document types and modules that reproduce, subset, and extend HTML” [40]. This becomes apparent in our discussion of XHTML Modularization and XHTML-Family Markup Languages.

2.3.2 XHTML Modularization

The design goal of XHTML, to achieve general user-agent inter-operability, is accomplished through a method called modularization [41]. Because XHTML is an XML application, it is simple to introduce new markup tags into XHTML. With this process, subsets of HTML can be created and extended to form XHTML-conforming modules. An XHTML module is a set of existing HTML elements (or tags) or future elements which offer a specific functionality. The modules can be combined with each other and with the desired feature sets of different user agents to form different XHTML subsets or different XHTML-conforming document types, or XHTML-family markup languages. These languages can

¹⁰ XHTML 2.0, W3C Working Draft, May 2003, W3C, 16 July 2003

<<http://www.w3.org/TR/xhtml2/>>.

then give rise to different presentations of the content. As an example, the five tables below explain the core modules of XHTML. They are the modules that must be included in every XHTML-family conforming document type [42].

Structure Module	The Structure Module defines the major structural elements for XHTML. This module is the basic structural definition for XHTML content.
Text Module	This module defines all of the basic text container elements, attributes, and their content model.
HyperText Module	The HyperText Module provides the element that is used to define hypertext links to other resources.
List Module	The List Module provides list-oriented elements.

Table II-1: Core Modules of XHTML [43]

Elements
body
head
html
title

Table II-2: Structure
Module [43]

Elements
abbr
acronym
address
blockquote
br
cite
code
dfn
div
em
h1
h2
h3
h4
h5
h6
kbd
p
pre
q
samp
span
strong
var

Table II-3: Text Module [43]

Element
a

Table II-4: HyperText
Module [43]

Elements
dl
dt
dd
ol
ul
li

Table II-5: List
Module [43]

2.3.3 XHTML-Family Markup Languages

The modularization of XHTML facilitates the creation of new markup languages. A certain XHTML-family markup language, a collection of XHTML modules, is used to encode the Web content so as to best fit the capabilities of a certain type of user-agent. Every such language must include the core modules “and may also use other XHTML-provided modules, other W3C-defined modules, or any other module that is

authoring language suitable for resource-constrained devices such as mobile phones. It extends XHTML Basic by adding some presentation elements and support for internal style sheets. The WAP Forum has been using XHTML MP as its client-specific markup language in the latest WAP 2.0 Specification [45].

2.3.4 Application of XHTML in Our Research

In our research of scalable content creation, the original XML-encoded Web content is transformed by XSLT into XHTML. Dominant Web browsers such as Internet Explorer and Mozilla are used to display the XHTML content on a personal computer. A mobile Internet browser that supports XHTML MP is used to display the same content on the mobile device. Thanks to the portability of XHTML, only one XSLT stylesheet needs to be created. The transformation result can be used by both thick and thin Web clients, for example, personal computers and mobile phones respectively.

2.4 WAP

2.4.1 WAP and WAP Forum

WAP is an open specification originated at the end of the last century to empower mobile users to access the Internet using wireless technology. WAP is developed by the WAP Forum, an industry association encompassing most of the world's handheld device manufacturers, wireless communication network vendors and operators, information technology hardware and software producers, and international standard-making bodies [46].

From a technical point of view, WAP provides an application environment and a suite of communication protocols for wireless devices to access the Internet and advanced telephony services so as to create mobile Internet applications and services. WAP is applicable to thin-client wireless devices such as, but not limited to, cell phones, pagers, two-

way radios, smart phones, PDAs, and communicators. WAP works with most wireless networks such as CDPD, GSM, GPRS, CDMA, TDMA, and 3G [47]. WAP has developed some applications for mobile users with wireless terminals, such as content browsing, mobile messaging, location-based services, mobile business, and many more, although at a rudimentary level. As a growing technology, WAP is constantly absorbing and developing new protocols to become a key and powerful technology for the mobile Internet.

2.4.2 WAP 2.0

In the summer of 2001, the WAP Forum released Version 2.0 of the Wireless Application Protocol [48]. This move resulted in bringing the wireless world closer to the Internet. By embracing the existing Internet technologies, WAP 2.0 can provide a connectivity model on a broader range of communication networks and ease the development of wireless applications. It does so by incorporating TCP into its communication protocol stack and migrating from WML to XHTML as the content encoding language. An important effect of this improvement is the change in the WAP programming model. In the earlier versions of WAP, namely WAP 1.0 and WAP 1.1, a WAP proxy (often called a WAP gateway) was required to handle the conversion between the communication protocols used in the wireless network and the standard TCP/IP protocol used in the wired network, the Internet. WAP 2.0, however, does not require such a proxy any more, since the communication between the client and the Web server can be conducted using HTTP 1.1 directly over TCP/IP. The WAP 2.0 programming model is shown in Figure II-5. It is clear to see that this model is closely aligned with the Web programming model. Nevertheless, although the deploying of a WAP proxy is no longer mandatory, it can optimize the communications process and may offer mobile service enhancements [49]. The WAP's

Optional Proxy Model is shown in Figure II-6.

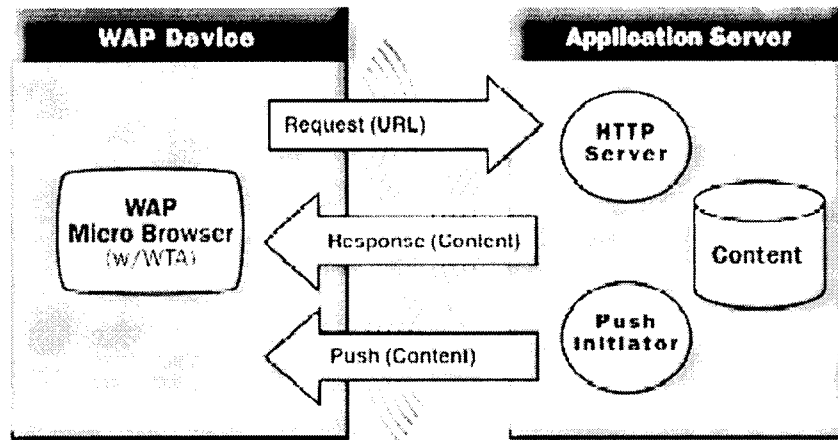


Figure II-5: WAP 2.0 Programming Model [49]

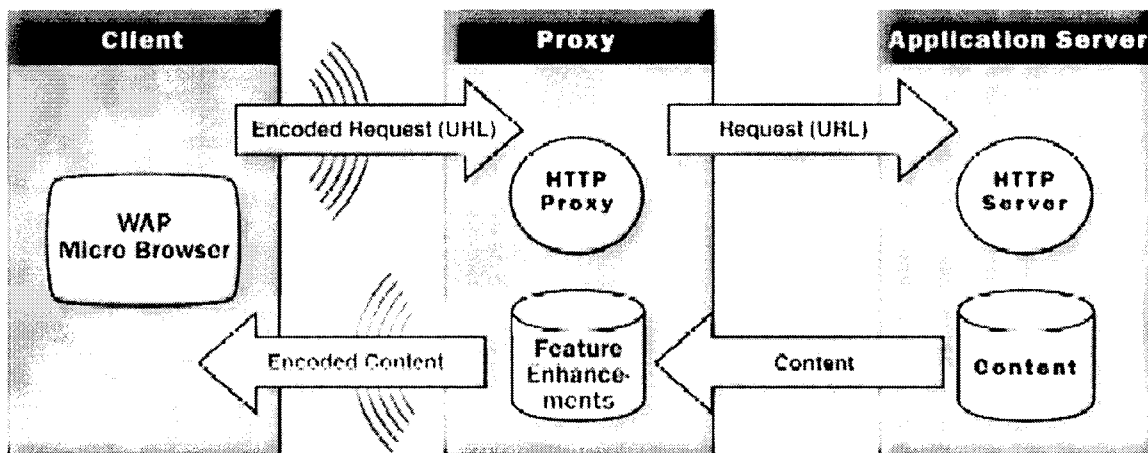


Figure II-6: WAP's Optional Proxy Model Supports Network-Based Optimizations [49]

2.4.3 Application of WAP in Our Research

Since WAP supports many wireless networks and devices, it is a favorable platform to use for the construction of information architecture for the mobile Internet. Specifically, we will use the Web/WAP 2.0 programming model, which means that the Web content stored at the Web server can be encoded in XHTML and can be delivered directly to the client via HTTP over TCP/IP without any format translation or protocol conversion.

2.5 JAVA

Java is a very attractive technology and the most popular development tool for today's Web application due to its ease of use, portability, and a rich resource of APIs. In our research, we will use numerous Java technologies, briefly introduced in this section, to conduct the empirical evaluation of our information architecture.

2.5.1 Java Programming Language and Platform

Java is both a programming language and a platform. Firstly, as a programming language, Java has numerous attractive attributes such as simplicity, object orientation, portability, and security [50]. Compared with many other languages, especially its object-oriented counterparts such as C++, Java is far more flexible in terms of operating systems upon which programs are written. This can be best reflected by Java's well-known slogan: "Write once, run anywhere." This portability is achieved by Java's unique way of compiling and interpreting the source program written in the Java language. The source program is first compiled into an intermediate, operating system-independent language called "Java bytecode." The bytecode, in turn, is interpreted by an interpreter called the "Java Virtual Machine" (VM) installed on the computer's operating system. The bytecode can be run on any implementation of the Java VM and thus can run on any computer that has a Java VM. Figure II-7 depicts Java's "write once, run anywhere" property.

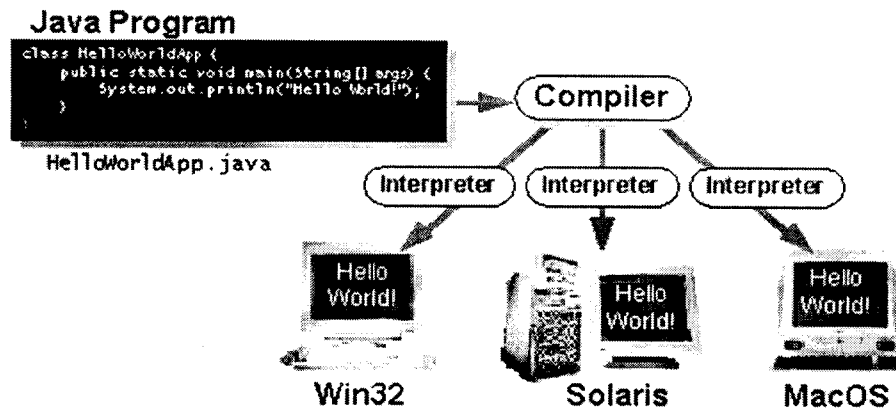


Figure II-7: Java's "Write Once, Run Anywhere" [50]

Secondly, Java is a platform. A platform is usually the hardware and/or software environment in which a program can run. Examples of platforms are Linux, WindowsXP, and MacOS. Most of the platforms are a combination of the operating system and hardware. The Java platform, however, is different, because it is a software-only platform that runs on top of other hardware-based platforms. The Java platform has two components: the Java Virtual Machine and the Java Application Programming Interface (Java API). The Java VM insulates Java programs from the operating systems of the computer while the Java API, grouped into packages of related classes and interfaces, offers a large collection of ready-made software components which provide many useful capabilities [50]. Figure II-8 shows the components of the Java platform and how they are related to the operating system and Java application programs.

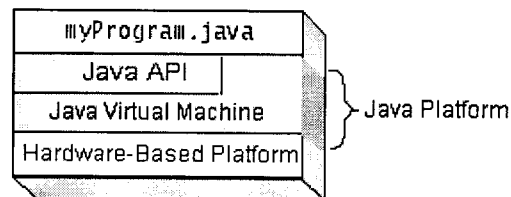


Figure II-8: Java Platform [50]

2.5.2 Application of Java in Our Research

Java has many specific technologies. In our research, we will use a few of them to develop a prototype Web application to evaluate our design of the information architecture. The technologies to be used are: Java 2 Standard Edition¹¹ (J2SE), Java 2 Mobile Edition¹² (J2ME), Java Servlet¹³, Java API for XML Processing¹⁴ (JAXP), and Java Document Object Model¹⁵ (JDOM). They are chosen because of their capability and efficiency in Web application development as well as in scalable Internet content creation. The five paragraphs below will describe them in detail.

The Java 2 Standard Edition is at the core of the Java technology and will be used in our research for Java application development. J2SE includes the essential compiler, tools, runtimes and APIs for developing applications in the Java programming language. It provides software developers with a platform for rapid application development and cross-

¹¹ Java 2 Platform, Standard Edition (J2SE), Aug. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/j2se/>>.

¹² Java 2 Platform, Micro Edition (J2ME), Jul. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/j2me/>>.

¹³ Java Servlet Technology, Jul. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/servlet/index.html>>.

¹⁴ Java API for XML Processing (JAXP), Jun. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/xml/jaxp/>>.

¹⁵ JDOM, Jun. 2003, JDOM, 16 July 2003 <<http://www.jdom.org>>.

platform compatibility. J2SE has a rich set of APIs to provide a wide range of functionality, such as the Essentials, Applets, Networking, Internationalization, Security, JavaBeans, Object Serialization, Java Database Connectivity (JDBC), 2D and 3D Graphics, Accessibility, Servers, Collaboration, Telephony, Speech, and Animation. Figure II-8 depicts the components in the J2SE version 1.3 platform. It can be seen that the Java 2 Runtime Environment (JRE) consists of the virtual machine, the Java platform core classes, and supporting files. The whole J2SE package includes the JRE and development tools such as compilers and debuggers [51].

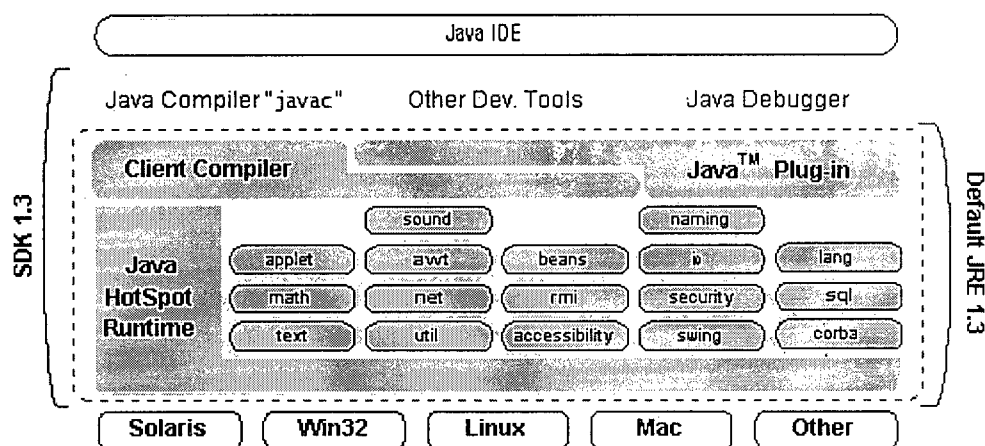


Figure II-9: Java 2 SDK, Standard Edition Version 1.3 [51]

In order to provide compelling Java technology solutions for manufacturers building all kinds of devices across the spectrum, from cell phones to desktops, a small Java application environment, Java 2 Micro Edition (J2ME), was created. J2ME is a framework for the deployment and use of Java technologies in today's rapidly expanding post-PC world. In our research, we use "J2ME Profile" as a means to classify terminal devices. J2ME will be discussed in more detail in Section 2.7, "Mobile Device Detection."

Java Servlet is the Java platform technology of choice for extending and enhancing Web servers. A servlet is a piece of Java source program that runs on the server side to

support a request/response computing model that is commonly used in Web applications. In such a model, a client sends a request message to a server and the server responds by sending back a reply message. Web developers can use the Java Servlet API to create servlets to enhance the responses to the requests from Web clients. Java servlets can do many tasks, like processing HTML forms, managing middle-tier processing, connecting to existing data sources, maintaining services, and so on. Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases which will be applied in the empirical evaluation in our research. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, re-usability, and crash protection. Since they have made many advanced Web applications possible, Java servlets are a popular choice for building interactive Web applications today. Third-party servlet containers are available such as Apache Tomcat Web Server¹⁶, Microsoft Internet Information Services¹⁷ (IIS), and others [54]. We shall use Aapche Tomcat Web Server as the servlet container and develop a servlet to test our research findings.

Java API for XML Processing (JAXP) is a set of Java APIs that support the processing of XML documents using DOM, SAX, and XSLT. JAXP enables applications to

¹⁶ Apache Tomcat, The Apache Jakarta Project, 2002, Apache Software Foundation, 16 July 2003 <<http://jakarta.apache.org/tomcat/>>.

¹⁷ Internet Information Services, 2003, Microsoft Corp., 16 July 2003 <<http://www.microsoft.com/WindowsServer2003/iis/default.msp>>.

parse and transform XML documents independent of a particular XML processing implementation. The benefit of using JAXP instead of a particular XML processing tool is to give the developer the flexibility to swap between XML processors. As different applications may call for different XML processors (such as high performance versus memory conservative parsers), it is advantageous to XML-enable Java applications without making application code changes. Consequently, JAXP has been adopted by many developers to rapidly and easily develop e-commerce services, application integration, and dynamic Web publishing. Similarly, we will use JAXP in our research, to simplify the application development and achieve a flexible result.

DOM, introduced earlier in this chapter, can be viewed as a tree representation of the XML document. DOM is very useful when programmers want to keep the entire document in memory and allow random access to the entire tree. Unfortunately, DOM suffers from a number of design flaws and limitations that make it less than ideal as a Java API for processing XML [56]. These constraints make DOM a clumsy and difficult tool to use. An alternative, Java Document Object Model (JDOM), is an open-source, tree-based, pure Java API for processing XML documents that eliminates DOM's unwieldiness. JDOM provides a way to represent an XML document for easy and efficient reading, manipulation, and writing. It has a straightforward API that makes it lightweight, fast, and optimized for Java programmers. Most developers find JDOM cleaner, simpler, and far more intuitive to use than DOM. In many ways, JDOM is a significantly improved replacement for the DOM technology. Thus, we shall use JDOM as the XML parser in our research.

2.6 Channel Detection

One goal of our research is to achieve content scalability over communication channels. This requires us to build into our information architecture the capability of detecting the conditions of the channel from the Web server to the client. More specifically, BW, end-to-end delay, and jitter are the conditions that affect the Internet multimedia content most. Therefore, the availability of their values is required for scalable content creation [57]. BW and end-to-end delay affect the download speed of contents such as texts and images, while jitter is a more serious issue for real-time multimedia streaming content. Nevertheless, the effects of jitter can be largely eliminated by introducing a setup delay [58].

2.6.1 Channel Detection on the Internet

Because the Web is extremely heterogeneous and dynamic in nature, it is difficult to accurately detect the communication channel between two computers on the Web [59]. Nevertheless, extensive theoretical studies and empirical approaches have been undertaken during the past few years and have yielded some satisfactory methods to detect or estimate the channel BW, round-trip time, and end-to-end delay. Packet-pair is a well-studied method to measure the BW between two end points on the Internet [60]. For the past ten years, many extensions have been proposed to make this technique more accurate, scalable, and agile in adapting to BW changes [61–65]. The measurement of end-to-end delay has also received enormous attention because of the necessity to manage the performance of the Internet and the applications running over it. In “Identification of the Internet End-to-End Delay Dynamics Using Multi-Step Neuro-Predictors,” Alexander G. Palos summarizes many measuring techniques and implements an empirical strategy for the identification of the end-to-end delay and round-trip time dynamics for a source destination pair on the Internet using

recurrent neural networks [66]. Furthermore, Pawan Goyal, Simon S. Lam, and Harrick M. Vin, the authors of “Determining End-to-End Delay Bounds in Heterogeneous Networks”, propose a method for determining an upper bound on the end-to-end delay for a variety of sources, which is especially useful for supporting multimedia applications in a heterogeneous environment [57]. To summarize, using the techniques mentioned above, we can fairly accurately determine the conditions of the wired channel on the Internet.

2.6.2 Channel Detection on the Wireless Segment

In general, mobile Internet applications rely on communication over a channel that involves both wired and wireless segments. For scalable Web applications, the channel conditions of the wireless segment also need to be detected. In “Wireless Awareness for Multimedia Applications,” L. Cheng and I. Marsic point out that, to achieve behavior adaptation from the application’s point of view, the application should “know” the conditions of the wireless network, namely, BW, latency, and jitter. These are the same parameters that need to be collected for the wired segment [67]. However, the acquisition of these parameters from a wireless link is far more complicated due to low BW, high Bit Error Rate (BER), and user mobility [68].

Fortunately, there are a few solutions to the detection of wireless channels. One effective approach stems out of one distinct advantage of the upcoming 3G networks: the use of the rich set of wireless channel condition information that is already available on the network. Byoung-Jo J. Kim, in “A Network Service Providing Wireless Channel Information for Adaptive Mobile Applications: Part I: Proposal,” develops a prototype, Wireless Channel Information (WCI), a network service that collects Physical/Medium Access Control (MAC) layer parameters for wireless channel conditions and outputs the

parameters to the adaptive mobile applications to help with the adaptation decision making [69]. This entails the WCI service to interface with various network elements in the wireless access network to collect parameters related to channel conditions for a mobile device.

Based on these parameters, the WCI service abstracts clearly defined values of parameters such as BW and delay in the formats that applications can use. Communicating directly with adaptive mobile applications in the server network through standard protocols, the WCI service is an effective means to detect the conditions of the wireless channel.

2.6.3 Proposal of Channel Detection for a Combined Wired and Wireless Channel

Given the findings discussed above, we propose a channel detection framework for our content creation information architecture as drafted in Figure II-9. We assume that WCI service is available on the wireless network and WCI outputs dynamic data to external applications via the WCI Service Gateway. We build a Channel Detection Module (CDM) into the Web server, whose main task is to handle the creation of Web content. The CDM is responsible for collecting the conditions of the channel between the Web server and the client. If the client is on a wired connection, the CDM measures the BW and end-to-end delay using the methods introduced in Section 2.6.1. If the client is on a wireless link, the CDM will do two things: 1) perform channel detection between the Web server and the WCI Service Gateway using the same methods as for the wired client and 2) obtain the wireless channel information via the WCI Service. After receiving the “raw” statistics from both the wireless and the wired segments of the channel, the CDM consolidates them, generates a meaningful set of data describing the channel conditions, and outputs that data to the Web server. In this way, we can successfully detect the conditions of a combined wired and wireless channel.

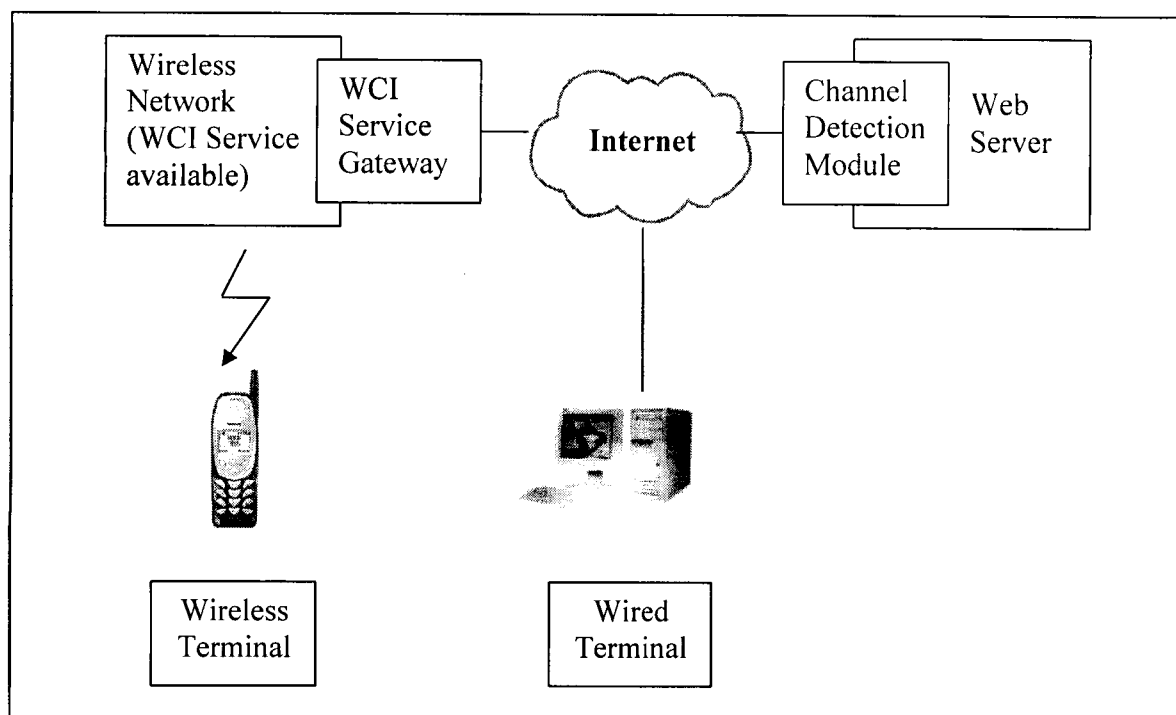


Figure II-10: Proposed Channel Detection Framework

2.7 Mobile Device Detection

2.7.1 Device Diversity

In addition to the Internet, the surge of mobile public communication systems over the past decade is another quantum leap in the development of information technology. For example, ten years ago a cell phone was considered to be a fancy luxury or a technical wonder, but now it is almost a daily accessory, as common as a wallet or watch. Moreover, the classes of mobile communication devices have reached far beyond cell phones. As the J2ME Web site explains, “While new classes like smart cellular telephones, pagers, and PDAs proliferate, traditional consumer electronics including televisions, VCRs, CD players, and game machines are also becoming smarter and gaining new networking capabilities” [70].

Nowadays, mobile handheld devices with Internet connectivity are widely available on the market. For instance, the embedding of mini Web browsers into mobile devices is so

common nowadays that it has become a “default” practice of the terminal vendors¹⁸. This is because as the mobile Internet flourishes, no vendor wishes to lose potential customers due to the lack of Web-accessing functionality in their products. However, the challenge of these proliferating wireless terminals is the rapid increase in the diversity they bring to device capabilities, which include CPU, memory, multimedia application support, and display characteristics such as screen size, resolution, color, and sound. Device diversity is one of the major scalability issues on the mobile Internet waiting to be solved.

2.7.2 Device Detection Techniques

In order to achieve content scalability over multiple devices, the device characteristics relevant to content display must be revealed to the Web server. Currently, there are several commonly used techniques for this purpose. Specifically for our content browsing application, device detection can be accomplished by conveying the client’s device capability parameters to the server in an HTTP request while then the server uses a Java Servlet (or PERL or ASP) to intercept the request and perform device characteristic recognition. More specifically, this recognition is accomplished in a number of ways. To name three: 1) employment of the User-Agent String (UA-String) in the HTTP request header, 2) use of W3C’s Composite Capability/Preference Profiles¹⁹ (CC/PP) to code the device profile in Resource Description Framework²⁰ (RDF) and then include the device profile in the HTTP

¹⁸ 3G News Information and 3G Store Homepage, 2003, www.3g.co.uk, 16 July 2003
<<http://www.3g.co.uk/Phones.htm>>.

request payload, and 3) inclusion of a Uniform Resource Locator (URL) in the HTTP payload while this URL is a link to the device profile on the Internet. Among the three, the most efficient is the UA-String method. The UA-String, included in the HTTP request header, indicates the type of Web browser that is installed on the Web client. Fortunately, the UA-String can be extended to contain more information other than the browser type, and the extension is very simple: it requires only the addition of new name/value pairs. For instance, “color/enabled” indicates that the device is capable of displaying colors, “memory/128K” reports that its memory size is 128KB (Kilo Byte(s)), and “screen/2.1×1.1 inches” tells the how long and wide the screen is. The name/value pair approach is not only easy, but also scalable because there is no restriction as to what names can be added. Thus, the UA-String method is a flexible and attractive method for revealing device information from the client.

This flexibility may tempt us to add as many name/value pairs as we want. As an illustration, Table II-6 below is a typical listing for device hardware configuration. All the “items” and their “values” in this table could be added into the HTTP request header, however, this would require the server to be able to “understand” all the possible items of all

¹⁹ CC/PP Composite Capability/Preference Profiles (CC/PP): A user side framework for content negotiation, W3C Note, Jul. 1999, W3C, 16 July 2003

<<http://www.w3.org/TR/NOTE-CCPP/>>.

²⁰ Resource Description Framework (RDF), Apr. 2003, W3C, 16 July 2003

<<http://www.w3.org/RDF/>>.

kinds of terminals. Due to the large number of available device models and the rapid development of new ones, this approach may quickly become difficult to control. Thus, we need a way to categorize the terminals based on qualities relevant for content browsing.

Item	Value
Product name	A1304T
Manufacturer	Toshiba Corporation
Size Approx.	49(W) × 93(H) × 25(D)mm
Battery time	Approx. 110 minutes
Colors	Blue, White
Display capacity	260,000 colors approximate
Capacity of data folder	Approx. 5 MB (or 500 files)
Capacity of email outbox	Approx. 400 KB (or 200 emails)
Capacity of email inbox	Approx. 400 KB (or 200 emails)
Maximum number of characters	5,000 Japanese font characters (sent per message)
Maximum number of characters	5,000 Japanese font characters (received per message)
Capacity of attached files (Sending)	total of 100KB (up to five files)
Capacity of attached files (Receiving)	up to 100KB (up to five files)
Picture characters accommodated	497 picture characters (including 96 moving)
Music	Up to 40 polyphonic chords
...	...

Table II-3: Toshiba A1304T, a 3G Handset (Partial) [71]

2.7.3 J2ME Profile

The recent introduction of J2ME gives us a means for device categorization because it uses a “J2ME Profile” to classify mobile devices according to their hardware capabilities. To give a little background, J2ME is the edition of the Java 2 platform targeted at consumer electronics and embedded devices. Since most mobile terminals fall into this category, J2ME allows developers to use Java technologies to write programs suitable for mobile devices. Specifically, J2ME uses programming specifications and a special virtual machine called a Kilobyte Virtual Machine that allows J2ME encoded programs to run on a mobile device [72]. There are two levels of programming specifications: The first level, J2ME Configuration, is comprised of a virtual machine, core libraries, and certain APIs. Currently, there are two J2ME configurations: the Connected Limited Device Configuration²¹ (CLDC) and the Connected Device Configuration²² (CDC), targeted respectively at two broad categories of devices, CLDC for those with 128–512KB of memory available for the Java technology environment and applications, and CDC for those devices with over 512KB available for the Java technology environment and applications. The second level of programming specifications, J2ME Profile, is a specification that details the Java technology APIs necessary to provide a complete runtime environment for a specific kind of device. Since a J2ME profile is built on top of and utilizing an underlying J2ME configuration, the

¹⁹ CLDC Specification, May 2003, Sun Microsystems, Inc., 16 July 2003

<<http://java.sun.com/products/cldc/>>.

²⁰ Connected Device Configuration, Jul. 2003, Sun Microsystems, Inc., 16 July 2003

<<http://java.sun.com/products/cdc/>>.

profile can be thought of as selecting classes from APIs to form a complete application environment to meet the needs of specific industry segments. Figure II-11 below shows the components of J2ME Configuration and J2ME Profile [73].

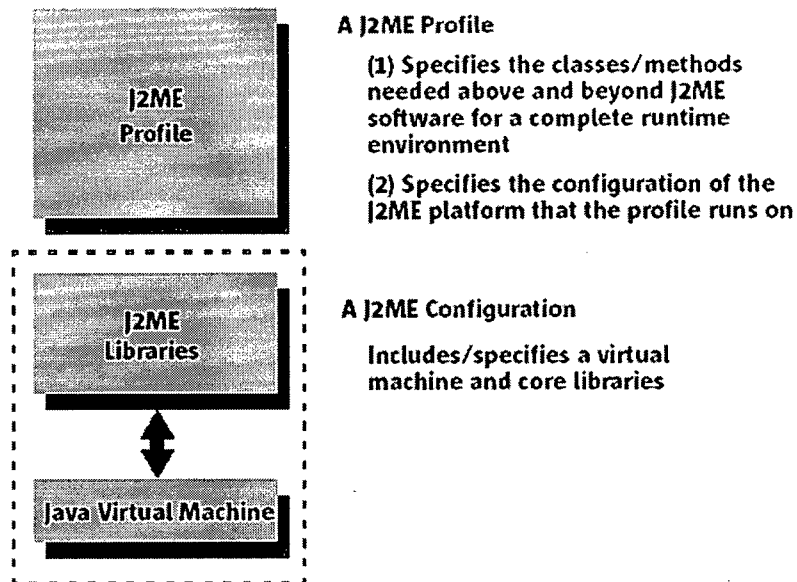


Figure II-11: J2ME Configuration and Profile [73]

There are a number of profiles in J2ME: Mobile Information Device Profile²¹ (MIDP), Personal Profile²², Personal Basic Profile²³, and so on. The profiles set hardware

²¹ Mobile Information Device Profile (MIDP), Apr. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/midp/>>.

²² J2ME Personal Profile, Jul. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/personalprofile/>>.

²³ J2ME Personal Basis Profile, May 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/personalbasis/>>.

requirements related to display, memory, networking, and user input. MIDP, for example, is targeted at mobile information devices which include, but not restricted to, cell phones, two-way pagers, and wireless enabled PDAs. The latest specification of MIDP is “MIDP 2.0 on CLDC 1.0.” This version sets the hardware requirements for MIDP 2.0 compatible Mobile Information Devices as demonstrated in the table below [73].

Display:	Screen-size: 96×54
	Display depth: 1-bit
	Pixel shape (aspect ratio): approx. 1:1
Input:	One or more of the following user-input mechanisms: one-handed keyboard, two-handed keyboard, touch screen.
Memory:	256KB of non-volatile memory for MIDP implementation, beyond what is required for CLDC ²⁴ .
	8 KB of non-volatile memory for application-created persistent data.
	128KB of volatile memory for the Java runtime (e.g. Java heap).
Networking:	Two-way, wireless, possibly intermittent, with limited BW (e.g. BW \leq 9600bps) connectivity to some type of network.
Sound:	The ability to play tones, via dedicated hardware, or software algorithm.

Table II-4: MIDP 2.0 on CLDC 1.0 [73]

²⁴ CLDC 1.0 requires: 128–512KB of total memory with \leq 256KB ROM/ Flash and \leq 256KB RAM. In most cases devices have more ROM than RAM or flash memory.

The use of J2ME Profile to classify the content-browsing capability of a wireless terminal provides several benefits. First, the profiles are distinguished from each other by hardware capabilities that are most relevant to content-browsing applications such as display size, networking capability, and memory. Second, the number of J2ME profiles is limited. Even though it is expected that there will be a few more profiles to be introduced, compared with the “full list of name/value pairs” approach, J2ME profile still helps us keep categories under control. Last, but not least, J2ME is generally recognized as a long-term reliable platform. The tremendous momentum of the J2ME technology has already placed it as the dominant terminal platform in the wireless sector. According to Zelop Group, “J2ME will be found in over 450 million handsets sold in 2007, corresponding to 74% of all wireless phones that ship that year.” In addition, competing technologies, such as Qualcomm's Binary Runtime Environment for Wireless²⁵ (BREW) and a specialized version of Microsoft Windows, are not even contenders [74]. Considering the above advantages of J2ME, we shall include in the HTTP request header a name/value pair that indicates the J2ME profile of the device, which will serve as the basis of our device detection.

2.8 Summary

In this chapter, we have introduced a number of the most recent technologies which have potential applications to the creation of information architecture for the mobile Internet. Based on our studies of these technologies, we will use XML to store the Web content,

²⁵ Qualcomm BREW Home Page, Apr 2003, Qualcomm Inc., 16 July 2003

<<http://www.qualcomm.com/brew/>>.

XSLT to transform it, and XHTML to present it. We will build our architecture based on the Web/WAP 2.0 programming model and code it in Java. We will apply our channel detection and device detection proposals to acquire the necessary input parameters for content scaling. With an elaboration of our proposed data structure for content encoding and information architecture in the next two chapters, the applications of the technologies mentioned above will become readily evident.

Chapter III: A New Data Structure for Content Encoding

3.1 Overview of Approach

The goal of this thesis is to create a novel information architecture for scalable and dynamic content generation for the mobile Internet. This means that a single copy of original Internet content must be automatically and dynamically adapted into multiple presentations for a wide variety of environment conditions. These conditions include the diverse capabilities of Web-accessing clients, various conditions of the communication channel, and possibly different preferences of end-users. Other important requirements for information architecture concerning the growing proliferation of multimedia content over the Web include: 1) a high scalability over media types, 2) minimal information loss during the adaptation, and 3) the potential of providing QoS support.

We approached the problem from a unique angle. We focused on encoding the source content in an elegant data structure so as to facilitate content adaptation for a dynamic and multi-dimensional environment. This data structure must form the basis of a wide range of scalabilities and support the use of XML and related technologies in order to realize the “one content, multiple presentations” principle. Upon this data structure, we built a new information architecture to perform dynamic adaptation on the content based on knowledge of the environment encompassing device, channel, and personalization specifics. Information of the environment needs to be collected immediately prior to the adaptation process.

3.2 A New Data Structure for Content Encoding

The basic function of a data structure is to contain data. Since the data structure that we want to design is expected to form the basis of various content scalabilities, it must

contain not only the original content, but the scaled contents as well. In addition, it must suggest for which kind of environment a particular scaled content is suitable. Sections 3.2.1 through 3.2.4 describe in detail the steps of how this data structure is conceived to meet the design purpose.

3.2.1 Decomposition of Web Content

Having defined the objectives that the data structure needs to achieve, we started the design process. To begin with, as a data structure is meant for storing data, we must first make a decision on the object that is to be contained within. For example, should the Web page be treated as an integral entity to be held in the data structure? Or should it be partitioned into parts? As we explain in the next paragraph, the multimedia nature of current and future Web content suggests that the latter is a more suitable choice.

When producing good content, it is every content provider's inclination to embellish a Web page with rich information instead of inadequacy. However, due to the restrictions of the wireless communication channel and the display capability of thin clients, it is often the case that a well-cultivated content document is not satisfactorily delivered to the client and/or rendered on the device display. Often a Web page with bulky multimedia information fails to reach a mobile handset because of timeout. Another difficulty is a long narrative that preoccupies the end-user with navigation instead of attention to reading. Thus, it is clear that, in order to achieve a good user experience, we need to adapt the multimedia content for various channel conditions and different terminal devices. However, we must first determine whether the content in question needs to be adapted or not. Thus, the Web server must ask the question, "Can this Web page be efficiently transmitted over the communication channel and properly rendered on the client?" If the answer is "yes," the Web page will be shipped

right away. If not, it will be adapted before the delivery. Yet it is unfeasible and unwise to adjust a Web page as a whole because it is a collection of many content elements of various data types such as texts, lists, tables, links, images, audio files, and video files. Different data types generally require different conversion schemes. Even for elements of the same data type, the degree of scaling can be different. Therefore, it is necessary to break one content document into its basic building blocks, i.e. content elements, and then perform the appropriate adaptation on each individual element.

The decomposition of the Web page is not difficult as each content element in a Web page is an independent entity enclosed in markup tags. This is especially true with Web pages encoded in XML because XML requires the content author to strictly follow the “well-formedness” principle and, therefore content elements are easy to identify as leaves of the XML tree. As shown in Figure II-2, the “Movie.xml” tree depicts individual content elements as leaf nodes such as “movieTitle,” “movieDescription,” “leadingActorName,” and so on. These nodes contain the real information that this XML Web page wants to convey to end-users. A table of all the content elements in “Movie.xml” and their corresponding “real information” is presented below. Once we have identified the individual content elements, we can put each of them into our data structure and move on to the next step which deals with the content adaptation.

Content Element	Real Information of the Content Element
<movieTitle>	Titanic
<movieDescription>	James Cameron's "Titanic" is an epic, action-packed romance set against the ill-fated maiden voyage of the R.M.S.
<leadingActorName>	Leonardo DiCaprio
<leadingActorLink>	www.leonardodicaprio.com
<leadingActorPhoto>	leodPicGif.gif
<leadingActorBio>	Leonardo Dicaprio is ...

Table III-1: Content Elements of "Movie.xml"

3.2.2 Alternate Content Element

Knowing the objects to be worked on, we must find an adaptation scheme for these objects and then contemplate how the data structure can assist with this scheme. A common adaptation method used in multimedia technologies is to convert an original file into a "simpler version." For instance, a twenty-word summary is a simpler version of a two thousand-word text, an 8-bit gray scale image is a simpler version of a 24-bit true-color one, and a video that is encoded in a low bit rate such as 12kbps is a simpler version of one encoded in a high bit rate such as 32kbps. Reduced in one or more file attributes requiring high computing resources from the environment, these simpler terms place less constraint on the channel conditions as well as the device capabilities. Therefore, the simpler-version scheme is a suitable one for our content adaptation for a multi-dimensional environment. Since a Web page is made up of various multimedia content elements, we can apply the simpler-version scheme to each of these elements. We call the simpler versions of an original content element its "alternate content elements." It is obvious that, in general, an

original content element can have more than one alternate content element. To give three examples, a two thousand-word text can be summarized in an unlimited number of ways, a large, color JPEG image can be converted to many versions with different “lossiness,” and audio files as well as video files can be encoded with numerous bit rates resulting in different audio and visual qualities.

These alternate elements can either be created manually or automatically generated by multimedia file conversion tools. The recent development in multimedia technologies has resulted in many software tools to automate conversion in images, audios, videos, et cetera. Taking image as an example, format conversion, size decrease, color reduction, and many other automatic functions can be found in many software products, from well-known ones such as CorelDraw, Adobe Photoshop, and Macromedia Fireworks to less famous, yet high-quality ones such as Web Image Guru, ImageMagick, and so on [75]. Tools that perform format conversion of audio files and video files in order to achieve better compatibility with the delivery and presentation are widespread on the Internet [77]. In most cases, the “automatic” mechanism is certainly more favorable because it can dynamically produce elements suitable for changing environment conditions. Occasionally, however, the “manual” way is preferred as, for instance, in the case of text summarization, where machine generated summaries may not maintain the correct meaning of the original text [78]. Text summarization, one of the technologies to solve the information overload problem caused by the proliferation of the Internet, has been under active research for approximately ten years. Although many theories as well as practical tools have flourished, there remain, however, many problems to be solved [79]. Nevertheless, it is hopeful that satisfactory text summaries will be generated automatically with the development in this research area.

Naturally, it is not enough for our data structure to simply provide alternate content elements. To help us achieve the scalable content creation with ease, these alternate content elements must be properly linked with their originals. Two important design considerations are involved at this step: First, we want our data structure to be a tree because we intend to map the contents in the data structure to an XML document, which has a strict tree structure. (The reason we want to use XML is because an XML document can be transformed into many presentations to suit different devices.) Second, the large variation in channel conditions and device capabilities demand many alternate content elements for each single original element. Considering these two factors, a tree-form data structure for an original content element is designed as shown in Figure III-1 below. It can be seen that an original content element, denoted as "Original Element," has many children: "Alt 1," "Alt 2," until "Alt n," which are the alternate content elements. In addition, Original Element has a child "Data" which is used to contain the raw information of Original Element. For instance, if Original Element is an image element that represents a JPEG picture named "image_original.jpg," Alt 1 can be a text element which serves as a text explanation of the image, Alt 2 can be another image element which represents a small-sized version of the original, while Data contains the image file itself, i.e. "image_original.jpg." It is always advantageous to include in the data structure an alternate element that contains only a short text because such a simple version can be accommodated by any kind of environment.

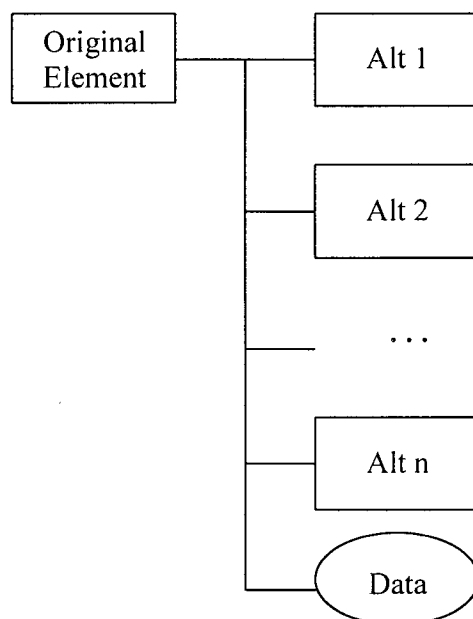


Figure III-1: Many Alternate Elements for an Original Element

Although we have differentiated the original and alternate content elements in our discussions above, these two types of elements are in fact the same in the sense that they are both containers of multimedia information. Therefore, an alternate content element can have its own alternate elements and, thus, has the same structure as its original. This idea is depicted in Figure III-2 below. In this figure, all the alternate content elements, denoted as a series of “Alt”s, have the same form as Original Element. Each of them has its own “Alt”s and Data. To differentiate all the content elements, we call Original Content Element “Level 0” element, the immediate children of Original Element “level-1” elements, the immediate children of level-1 elements “level-2” elements, and so on. A level-1 element is the result of performing adaptation to Original Element once, while a level-2 element twice. The idea of the levels means that an original content element can be simplified in multiple steps, a mechanism to adapt content elements to meet more than one environment condition. For example, in the figure, Original Element represents a color JPEG image with a large file size

named “image_original.jpg.” Alt 2 at Level 1 is an image element that represents a smaller-sized picture, “image_small.jpg,” the result of reducing the file size of “image_original.jpg” by a factor of 10 due to a channel with narrow BW. Alt 1 at Level 2, say “image_small_bw.jpg,” is a black-and-white version of “image_small.jpg” because the terminal device lacks color capability. It can be sent that this “multi-step” adaptation is in line with the multi-dimensional environment factors that we must consider when we perform content adaptation.

Curves in Figure III-2 indicate that elements can share the same information. Therefore, to save space, we designate one of them as the primary element, keeping it unmodified, and change the other element into a link pointing to the primary element. Data nodes are always chosen as primary elements if there are other elements sharing information with them. The decision to link such elements together results in a directed acyclic graph (DAG) for an original content element. In implementation, the links can be easily constructed by using common tags such as “ID” and/or “IDREF.” The “ID” tag serves as a unique identity for an element while “IDREF” refers to an element defined by other attributes such as a descriptive name [77]. Either of these tags can be used to identify the primary element, and the other “identical” elements can simply be defined as an anchor element because the anchor tag, “<a>,” is used to create a “bookmark,” or “link,” inside a Web page by using the “ID” or “IDREF” as attributes. For example, clicking on an anchor element such as will lead to an element identified by “idref.” Therefore, by using these simple tags, it is extremely easy to realize our DAG.

as a replacement for Adaptable Element. Data, however, has no AR link, because it contains the raw information of Adaptable Element. The algorithm that performs the adaptation is straightforward:

- 1) Whenever the Web server encounters an adaptable content element, it starts the adaptation process by evaluating the first adaptation rule in the DAG, AR 1. If the evaluation result is “yes,” Adaptable Element is replaced by Alt 1; if “no,” the server moves down to AR 2 and evaluation this rule .
- 2) The evaluation of adaptation rules goes on until a particular AR is evaluated to be true. Then, its corresponding Alt is chosen to replace the Adaptable Element.
- 3) If none of the ARs is satisfied, Adaptable Element remains un-adapted and the content in Data is used as the final representation.

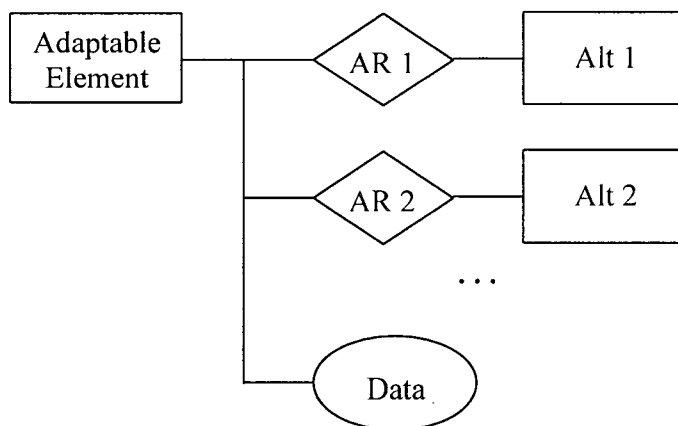


Figure III-3: Adaptation Rules Built into Data Structure

It can be seen that with the adaptation rules built in, our data structure is still a tree which can be easily “mapped” to XML: First, all the content elements, whether original or alternate, can be encoded as XML elements. Second, as for the ARs, since they are closely related to alternate elements but are not containers of content information, it is appropriate to encode them as attributes into their corresponding Alt elements. Finally, looking at the

content document as a whole, because all the alternate content elements are XML elements, they will be properly enclosed in XML tags, therefore, we can introduce a set of special tags to identify each alternate element which can serve as a means to determine the processing order of the adaptation rules. Knowing which rule to evaluate first and which next, our adaptation algorithm will work.

The processing order of adaptation rules is important to the performance of scalable content creation and, thus, needs to be designed carefully. A general principle is that adaptation rules dealing with device capabilities should be processed before those dealing with channel conditions. For example, if a device does not support image, the “support image?” rule should be evaluated before “enough bandwidth?” because such an order can save the trouble of adapting images to smaller sizes, yet at the end realizing that the device does not support images at all. Due to time constraints, we leave the design of a processing order for future work.

3.2.4 A New Data Structure for Content Encoding

So far, we have gone through the stages of developing a new data structure: We decomposed a Web page into individual content elements, proposed an alternate element mechanism to provide various simpler versions, designed a DAG to connect all the elements in a multi-level fashion to address the multi-dimensional consideration of environment factors, and embedded adaptation rules as a means to determine which element must be selected under certain environment conditions. An overall consideration of the above steps results in a new data structure for content encoding, as shown in Figure III-4 below. Being able to store a large collection of content elements and providing means to locate a proper one to match a set of environment conditions, this data structure has achieved our design goal.

Moreover, as a tree, this data structure can be easily converted into XML, an ideal format of Web content documents.

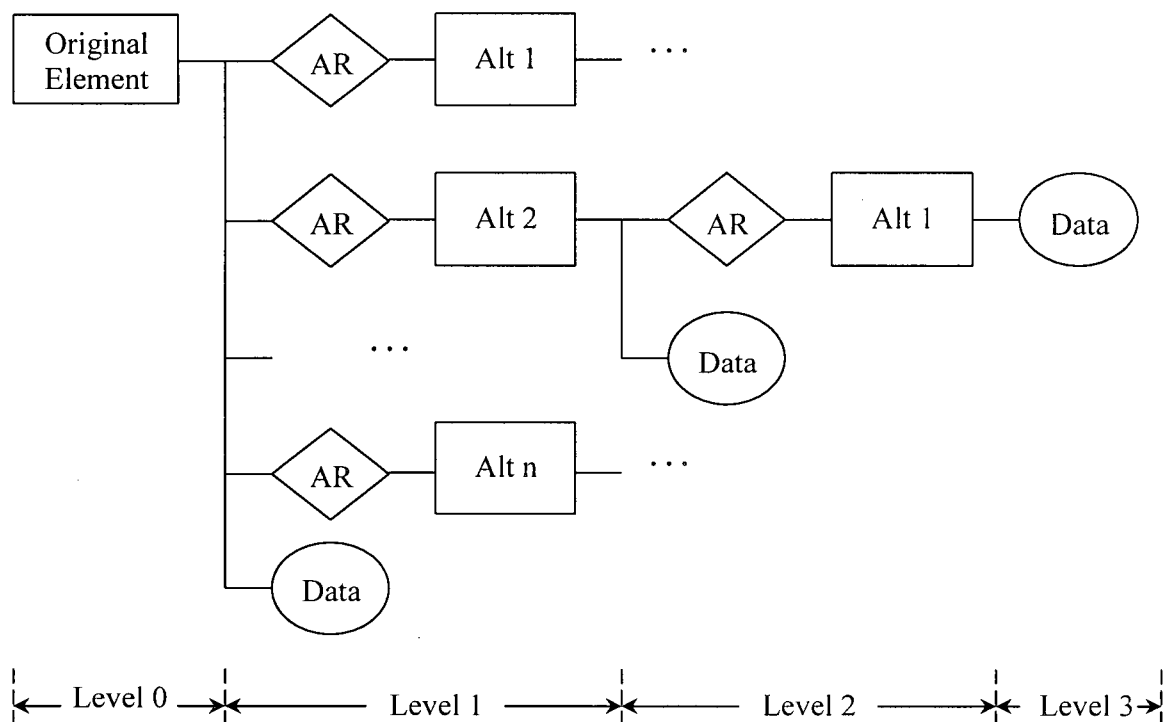


Figure III-4: A New Data Structure for Content Encoding

3.2.5 An “Image” Example

In this section, we demonstrate the use of the data structure that we created above with a concrete example. We chose image for our illustration because image is one of the most common content components of Web pages and it is supported by almost all the devices. Thus, an image example is practical as well as persuasive. We designed this example as shown in the Figures III-5 and III-6 below:

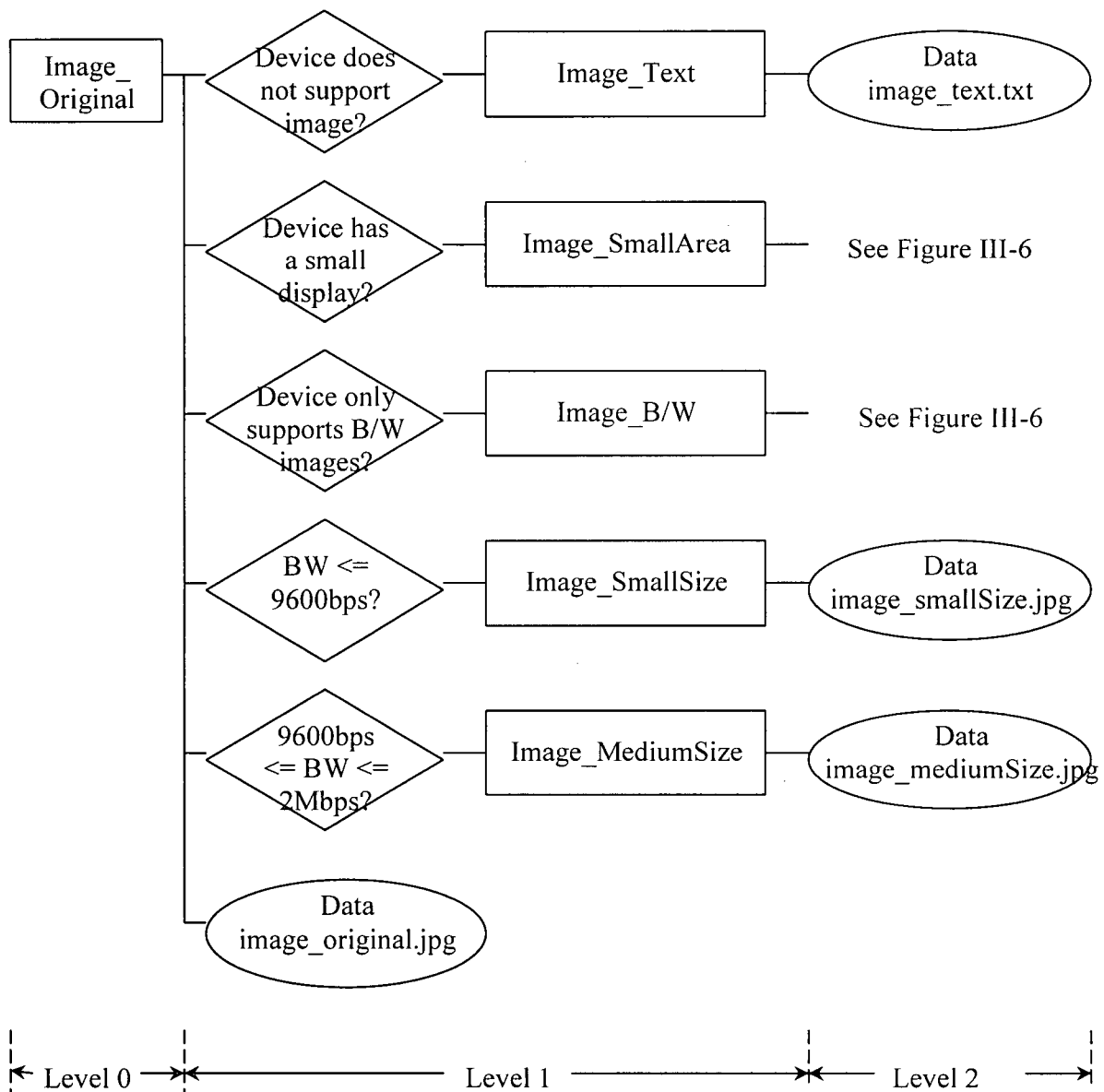


Figure III-5: An Image Example (Part 1, Level-1 Processing)

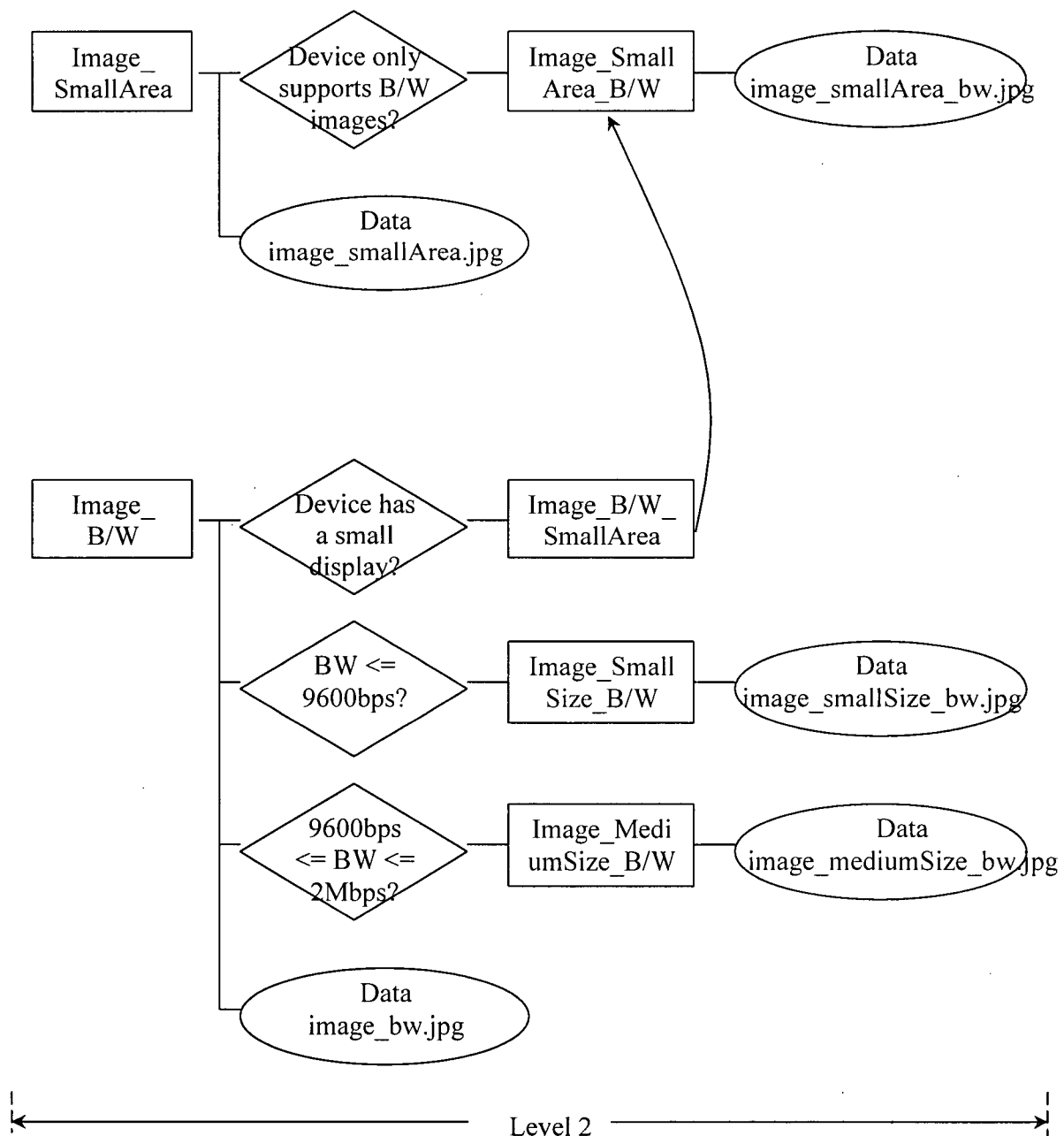


Figure III-6: An Image Example (Part 2, Leve-2 Processing)

As seen from the two figures above, the original image element, with its alternate elements and corresponding adaptation rules, is contained in our designed DAG structure. The root, Original Element, is named "Image_Original," whose raw information is contained within one of its children, Data. This raw information is an image file named "image_original.jpg," which is a 24-bit color picture in JPEG format, has a size of 2665KB, and requires a display area of 5.74 inches \times 3.96 inches. This image is most suitable for display on a device with a large display and color capability, and transmission under a channel with a high bandwidth such as over 10Mbps due to its large file size. When the environment is not ideal, Image_Original will be adapted. The adaptation process is executed as follows:

First, as shown in Figure III-5, level-1 processing is performed: Image_Original has five alternate elements as its level-1 Alts, namely, "Image_Text," "Image_SmallArea," "Image_B/W," "Image_SmallSize," and "Image_MediumSize." These Alts are linked with five adaptation rules respectively: "Device does not support image?" "Device has a small display?" "Device only supports black and white (B/W) images?" "BW \leq 9600bps?" and "9600bps \leq BW \leq 2Mbps?" These adaptation rules are evaluated as below:

- 1) To begin with, the first adaptation rule, "Device does not support image?" is checked. If it is evaluated to be true, an indication of an extremely simple device, Image_Original needs to be replaced by its simplest version, Image_Text, a short text explanation of the picture. If the evaluation result is false, meaning that the device does support image, we do not choose this text version, but move to the second adaptation rule.

2) The server examines the second rule “Device has a small display?” If the answer is “yes,” Image_Original has to be converted to an image occupying a small display area, Image_SmallArea. If “no,” this route is ignored and the next adaptation rule is checked.

3) The server inspects the third rule “Device only supports black and white (B/W) images?” If this rule evaluates “true,” Image_Original will be adapted from a color picture to a black and white one, i.e. Image_B/W. Or else, the server goes down to the fourth rule.

4) At the fourth adaptation rule, the server encounters “BW \leq 9600bps?” If true, Image_Original must be substituted by an image with a small file size, say, Image_SmallSize. If false, the server neglects this path and proceeds with the next adaptation rule.

5) The server checks the fifth and last rule in our example, “9600bps \leq BW \leq 2Mbps?” (We consider the range between 9600bps and 2Mbps a medium BW range.) If this rule is evaluated true, Image_Original is adapted to a medium file size, Image_MediumSize. Otherwise, the server moves toward the next as well as the last child of Image_Original, Data.

6) Arriving at Data means that all the level-1 adaptation rules have evaluated to be “false.” This indicates that no adaptation is necessary. Therefore, Data is chosen as the un-adapted version of Image_Original, and its content, “image_original.jpg” is to be delivered and displayed.

If, at any point of level-1 adaptation, a certain rule is evaluated to be true, its corresponding Alt will be chosen as a substitute for Image_Original and the server will

follow the corresponding path to come to the level-2 adaptation. To explain the adaptation process at this stage, we choose two possible occurrences of level-2 processing, as shown in Figure III-6, one corresponding to the route following Alt Image_SmallArea and the other the path following Image_B/W.

1) Suppose Image_SmallArea is chosen as the replacement for Image_Original, the server needs to check the adaptation rule for Image:SmallArea, “Device only supports black and white (B/W) images?” If this evaluates true, Image_SmallArea is adapted to Image_SmallArea_B/W, a black and white version of the already-reduced image file. Otherwise, it means the device can support color and, therefore, Image_SmallArea needs not to be adapted anymore. Thus, its Data, image_smallArea.jpg is used as the final representation of the original image.

2) Suppose Image_B/W is chosen as the substitute for Image_Original, the server needs to test the three adaptation rules linked with Image_B/W, i.e. “Device has a small display?” “BW \leq 9600bps?” and “9600bps \leq BW \leq 2Mbps?” If the first rule, “Device has a small display” is evaluated true, Image_B/W is adapted to Image_B/W_SmallArea in order to fit to a small display. Interestingly, this result is the same as the black and white replacement for Image_SmallArea, one sibling of Image_BW. We identify this equivalence with an arrow, as shown in the figure. In real implementation, when the server comes to Image_B/W_SmallArea, it will follow the link to Image_SmallArea_B/W and use the Data there.

3) If the first adaptation rule of Image_B/W evaluates false, “BW \leq 9600bps?” and possibly “9600bps \leq BW \leq 2Mbps?” will be the next to check. The processing of

these adaptation rules is similar to their counterparts at the level-1 stage and, therefore, we will not repeat it here.

Although we have only included two levels of alternate elements and a limited number of adaptation rules in our example, it can be shown clearly that the result of the process is a particular Data chosen to fit a group of environment conditions expressed by multi-stage adaptation rules. For instance, in our image example, if the environment is perfectly good, i.e. a device with large display and color capability and a channel with BW over 2Mbps, the original information, `image_original.jpg` will be delivered and displayed. Another example, if the device can only support black and white images and the channel BW is less than 9600bps, the original image will be adapted to `image_smallSize_B/W.jpg`, a JPEG file with a smaller file size and in black and white. In this way, the resultant image can be satisfactorily delivered over the channel and properly rendered on the device.

To summarize, the image example discussed above demonstrates how to use our invented data structure to encode a source content element for a Web page and how it is processed in order to produce an adapted version to suit a multi-dimensional environment. This data structure, forming the basis of various content scalabilities, can serve as the essential part of our information architecture. To illustrate the practical implementation of our data structure, we will, in the next section, make some simplifications to our theory and, in Chapter V, build a prototype Web application to show that our design can achieve highly satisfactory results.

3.3 A Simplified Data Structure as a Proof of Concept

We have developed a theory of a fully functional data structure for Web content encoding in order to create scalable content for the mobile Internet. To prove the feasibility

and power of our theory, we will design a simplified data structure in this section and implement a prototype Web application in Chapter V. We will show that even a simplified model of our data structure will yield satisfactory results.

3.3.1 A Simplified Data Structure

The goal of our verification is to see whether or not the original content element can be adapted to a proper “simpler version” based on some environment conditions. Therefore, a simplification can be made to ease the implementation of a prototype: Instead of numerous alternate elements for an adaptable element, only one is provided. This alternate element can be regarded as a representative of all the potentially different alternate elements.

Correspondingly, only one adaptation rule needs to be made for each alternate element. The simplified data structure is shown in Figure III-7 and is accompanied by Figure III-8, which shows an implementation of the adaptation rule. As illustrated in Figure III-7, the root of this tree structure is “Original Element,” which is identified by one of its children, “ID.” The adaptation rule, “AR,” as depicted in Figure III-8, is implemented by the combination of two elements, “Element Type” and “Element Attributes,” which are to be converted to a certain set of environment conditions. (The conversion technique is discussed in Section 3.3.2.) The AR basically states that if the content element cannot be supported by the environment, its Alt element at the next level will be used; otherwise, “Data,” which contains the raw information of this particular content element, will be chosen without adaptation. Original Element is considered a “level-0” element. Its immediate alternate element, Alt 1, is a level-1 element. Similarly, the immediate alternate element of Alt 1, which is not shown in the figure, is a level-2 element, and so on. The higher the level is, the more stages of conversion are applied to Original Element in order to make it “simpler.”

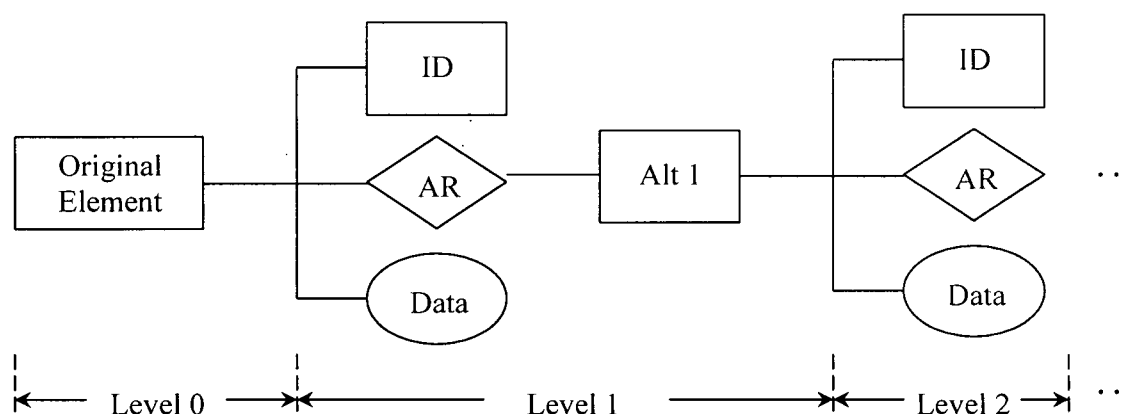


Figure III-7: A Simplified Data Structure for Web Content Element

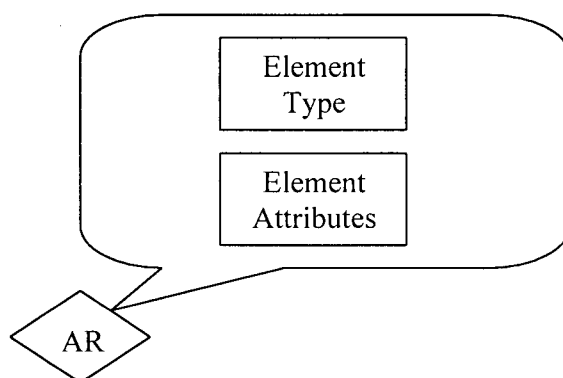


Figure III-8: Adaptation Rule Implemented by Element Type and Element Attributes

3.3.2 Adaptation to the Environment

Since we are adapting our content to match the channel conditions and end-device capabilities, the “simplicity” of a content element reflects the amount of resources that this element needs from the network and the device. The more complex a content element is, better channel conditions and more powerful terminal devices are required for efficient delivery and appropriate display. To implement the adaptation, we need a mechanism to quantify this complexity. For a piece of information, the file type and attributes such as file size, display area required, and encoding scheme are the determining factors of the computing resource needed. Thus, we classified a content element based on its file type and

attributes and then labeled the result with a “level” tag. The “Content Element-to-Level Conversion” table below is a simple implementation of this classification.

ContentType	FileSize	DisplayArea	ContentLevel
text	big	big	1
text	small	small	5
...
image	big	medium	2
image	small	small	4
...
audio	big	medium	1
audio	small	small	2
...
video	big	big	0
video	small	small	2
...
streaming	small	small	0

Table III-2: Content Element-to-Level Conversion (Partial)

The ContentType field in the table corresponds to Element Type in our simplified data structure. FileSize and DisplayArea are examples of Element Attributes. A table with only these two content attributes is sufficient for explaining the concept, although it is easy to include more attributes in the table. We categorize content elements into five levels. The most complex content element, such as streaming video, is defined to be at level 0, while the simplest content element, the short text version, is level 5. The lower the level, the “closer” the element is to Original Element in the data structure, and more complex this element is.

The assignment of levels to content elements is based on common user experience. The human “feel,” although vague, can provide two general guidelines for level allocation:

- 1) Higher levels are assigned to content elements with less complex file types. For example, generally speaking, images are more complex than texts and, therefore, tend to have lower

levels than texts; Videos are more complex than images and, thus, usually have even lower levels. 2) Among content elements sharing a same file type, if the file attributes of a certain element call for better environment, this element is assigned with a lower level. For instance, due to small file size, texts are simple content elements which do not require super environment conditions and, consequently, are often assigned with high levels such as 5. However, some texts are lengthy, thus requiring a big display area. This demand can only be met by devices with large screens. Since such devices are generally powerful terminal such as PCs, these long texts are considered elements that need good environment support and, therefore, a low level, such as 1, is assigned. Another example is image elements, which are basically assigned with relatively high levels, such as 4. This is because that most devices nowadays, whether thin or thick, support images, and it indicates that images do not pose a high demand on the device capabilities. Nevertheless, when the image is big in file size, it needs a relatively high channel BW for the efficient delivery. As a result, a relatively low level, 2, is assigned.

Since we have classified each content element and its alternate elements into “levels,” which serve as an indicator of the computing resources required, if only we can also classify the environment into “levels,” then we can use these “levels” to relate the content element to the environment. For instance, a cell phone with a display size of 10 cm × 20 cm and on a Wireless LAN connection of a BW of 11 Mbps is classified as “level 4,” which is a satisfactory setting for any level-4 content, like a small, still image, to be efficiently delivered and properly rendered. Similarly, a personal computer connecting to the Internet on a high-speed LAN, can be classified as “level 0,” suitable for the delivery and display of level-0

contents such as long text, color images, live videos, and so on. The “Environment-to-Level Conversion” table below maps environment conditions onto levels.

Device	BW	Delay/Jitter	SupportedLevel
MIDP	big	small	4
MIDP	big	medium	4
MIDP	medium	small	5
...
PC	big	small	0
PC	medium	small	2
PC	medium	big	3
...
PersonalProfile	big	medium	2
PersonalProfile	medium	small	3
PersonalProfile	small	small	4

Table III-3: Environment-to-Level Conversion (Partial)

3.3.3 The Processing of the Simplified Data Structure

Based on the general processing algorithm described in Section 3.2.3, we design a specific one to work with our simplified data structure. The basic principle is: Starting from Original Element, the server analyzes the Element Type and Element Attributes and decides if the analysis result meets the adaptation rule, AR, which leads to a simpler version at a higher level. If the AR is met, the server goes to the next level and performs the identical element analysis and AR evaluation. If the AR is not met, a phenomenon which means that the current element is supported by the environment, the server simply chooses the Data node at this level as the information needed for the final presentation of Original Element. A more detailed description of this process is as follows:

- 1) The Web server runs server-side APIs to obtain the condition of the channel (BW, delay, and jitter) and the capability of the device (processing power, display size, resolution, sound capability, color capability, et cetera.) Based on these parameters,

the server decides on the level that can be supported by the current environment. This level is called the “environment supported level.”

2) Each content element in a Web page has been packaged into our designed tree structure. The Web server starts with one tree and traverses it starting from the tree root, Original Element. The information of the Element Type and Element Attributes is used to match against the Content-to-Level Conversion table stored in the Web server. Consequently, a “level” is computed for this particular content element. This level is called the “computed level.”

3) The adaptation rule states that if the computed level of the content element is lower than the environment supported level, this element must be adapted. So, the server moves to the alternate element at the next level and checks the adaptation rule here. This recursion continues until the server reaches a particular alternate element whose computed level is equal to or higher than the environment supported level. Then, this alternate element is returned. The inclusion of a level-5 short-text alternate element in the data structure guarantees the return value is not “null.”

4) The server extracts “Data” from the returned element, stops traversing this tree, and starts with the next.

5) The processing ends after all the content elements in the Web page have been processed.

The result of this recursive processing is a collection of scaled content elements.

Each element is suited for delivery on the channel and presentation on the terminal display.

Therefore, this data structure, although simple, can achieve the design goals of our source content encoding and serve as the essential part of our information architecture. This

simplified data structure will be implemented as a testing prototype in Chapter V to prove that it can achieve satisfactory results in scalable content creation.

3.4 Novelty and Advantages

As pointed out previously, most common solutions to dynamic content creation focus on deriving adaptation functions and inventing new markup languages. We solved this problem by using a delicate structure to organize the source content in order to facilitate the data adaptation to suit multi-dimensional environment conditions. A distinctive advantage of our data structure stems from its elegant tree-form structure which makes it easy to map the Web content into XML so as to take advantage of the features of XML and its related technologies. Another virtue of our data structure is that, theoretically, it can chain together an unlimited number of alternate content elements and, thus, has the capability of supporting any level of content complexity in any type of media. Therefore, this data structure can guarantee minimal information loss during the content adaptation. Last, but not least, our data structure is easy to implement because tree and DAG are widely-used structures for data storage and processing and the various conversion tools for multimedia can help construct the tree by producing all the alternate content elements almost completely automatically. The straightforward implementation in turn facilitates extension of our solution to other Web applications. In fact, our data structure is independent from content browsing and, therefore, it can be applied to any Web application that needs a scalable creation and presentation of information. Chapter VI will discuss more on this issue.

3.5 Summary

In this chapter, we have presented an original approach to the problem of scalable content creation for the mobile Internet. We have invented a data structure, one that can

provide the scaled contents suitable for different environments, to encode the source content. We have demonstrated the stages of creating this novel data structure: 1) the decomposition of a Web page into a set of original content elements due to conversion requirements; 2) the recursive structure of alternate elements as the key to achieve the content scalability; and 3) the inclusion of adaptation rules into the data structure to determine a path leading the appropriate content element for delivery and presentation. Fulfilling the design requirements, the invented data structure is the most essential component in our information architecture. The rest of the work is to define necessary functions to utilize the outputs of the data structure to complete the scalable content creation process. This remaining work will be fully accomplished in the next chapter.

Chapter IV: A New Information Architecture for Content Creation

4.1 High-Level Design of Information Architecture

In addition to the source-encoding scheme that is the core of scalable content creation, both the dynamic adaptation and presentation transformation are indispensable elements of our information architecture. Their major functions consist of collecting the environment parameters, organizing the scaled content elements into a Web page written in XML, and transforming the XML document into XHTML for final presentation.

Since almost all Web-content applications are built upon the client/server Web programming model, we constructed our information architecture on this platform as well. We proposed a high-level design for our information architecture as shown in Figure IV-1. This is a typical Web/WAP 2.0 programming model incorporating all the components we need for scalable content creation centered around our simplified data structure. Figures IV-2 through IV-7 are block diagrams giving a detailed explanation of the overall high-level design. As illustrated, first, the client initializes an HTTP request and the server receives it; second, the “Device Detection” and “Channel Detection” modules collect the environment parameters from the channel and device; these parameters are in turn processed by “Level Processing” to determine the environment supported level; next, “Data Structure Processing” selects the appropriate scaled content elements; then, the output of “Data Structure Processing” is fed into “XML Production” in order to be constructed into an XML document; after that, “XSLT Transformation” converts the XML document into XHTML format; and, finally, the XHTML document is delivered to the client in the HTTP reply and is displayed.

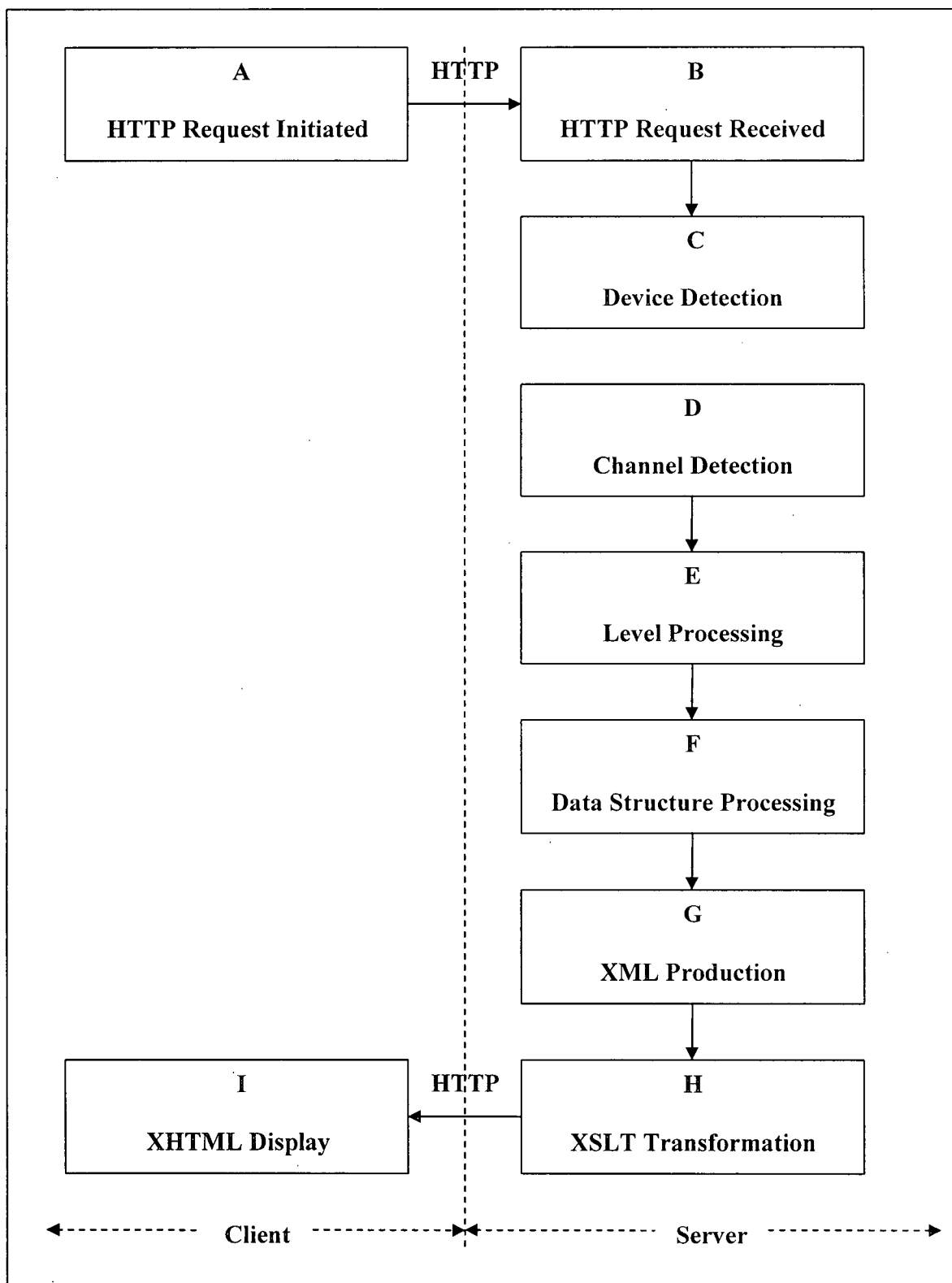


Figure IV-1: High-Level Design for the Information Architecture

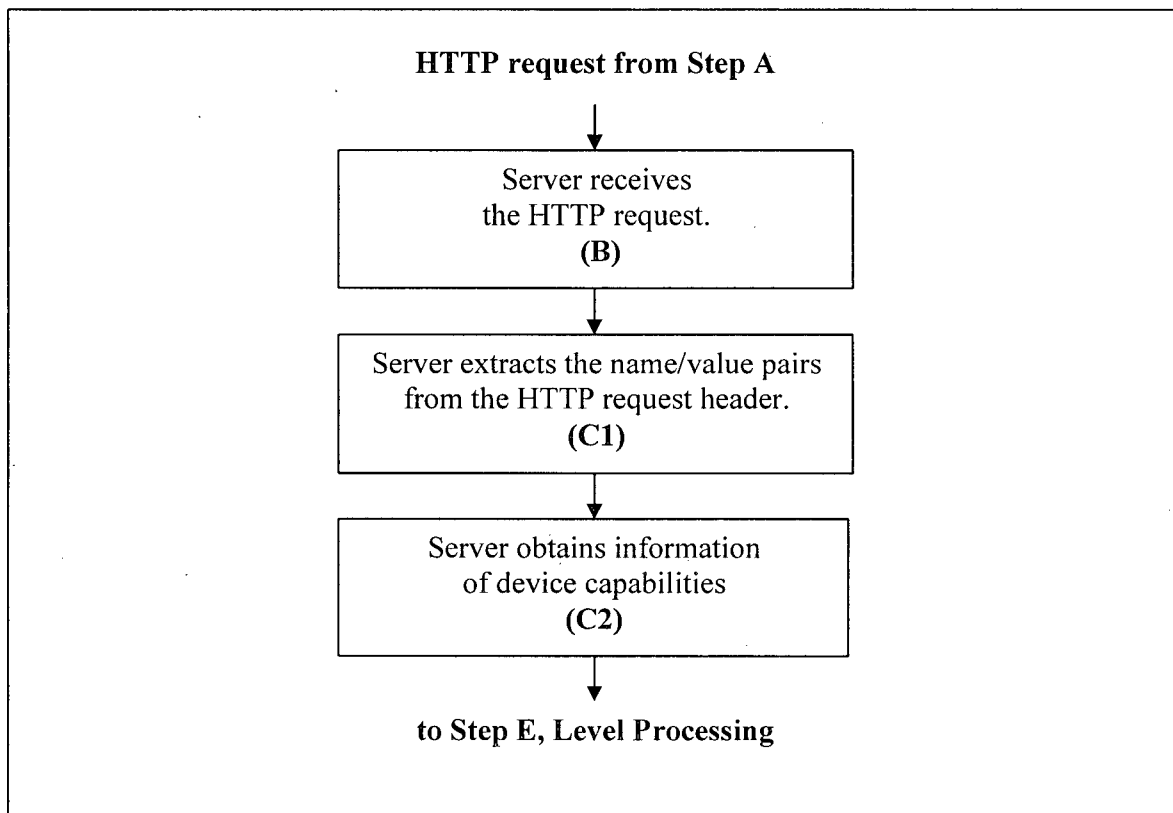


Figure IV-2: Step C, Device Detection

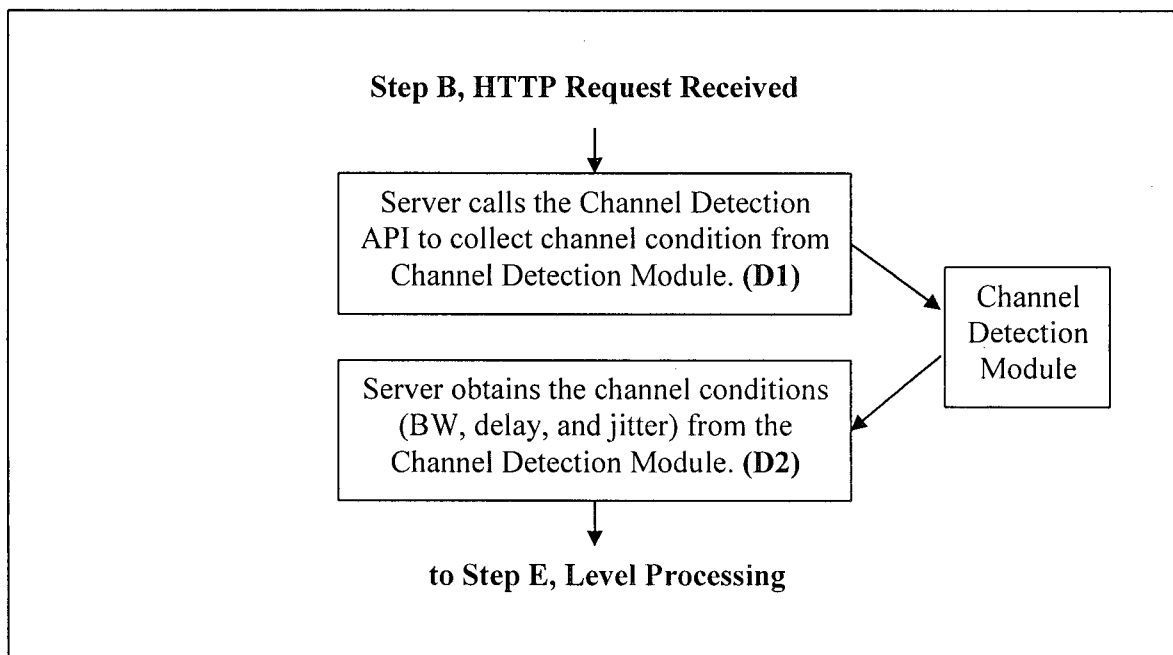


Figure IV-3: Step D, Channel Detection

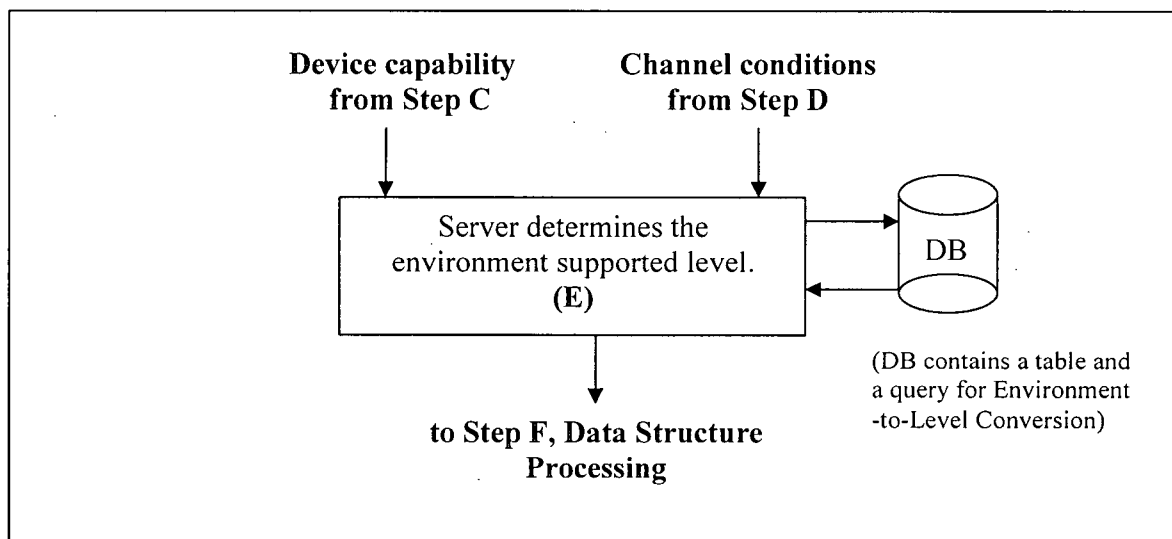


Figure IV-4: Step E, Level Processing

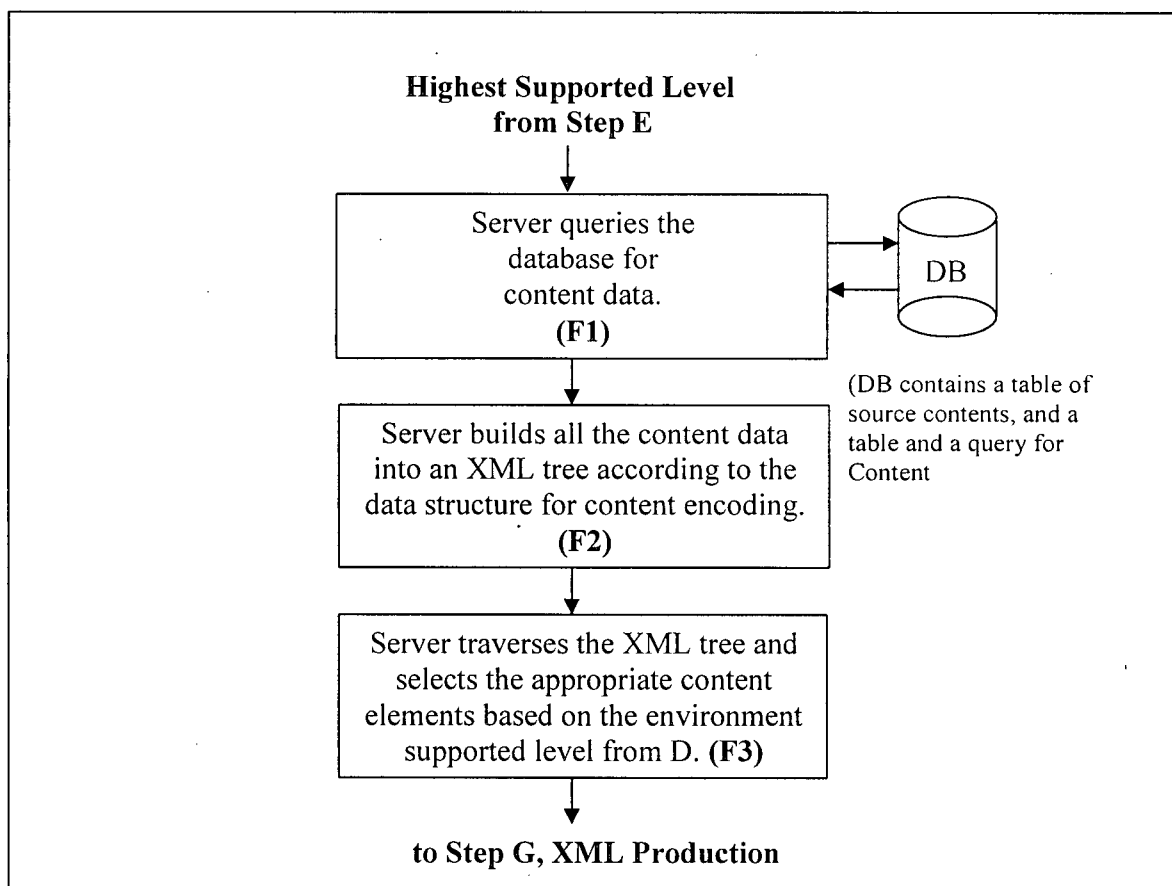


Figure IV-5: Step F, Data Structure Processing

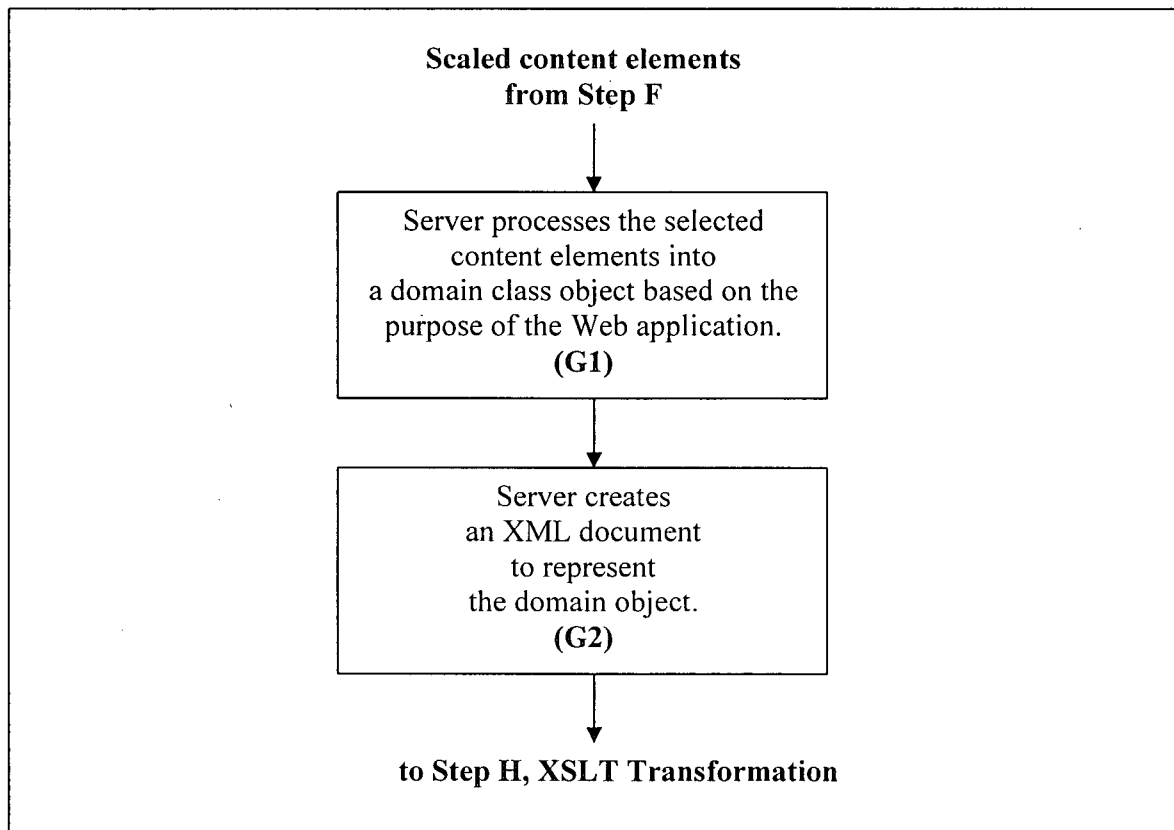


Figure IV-6: Step G, XML Production

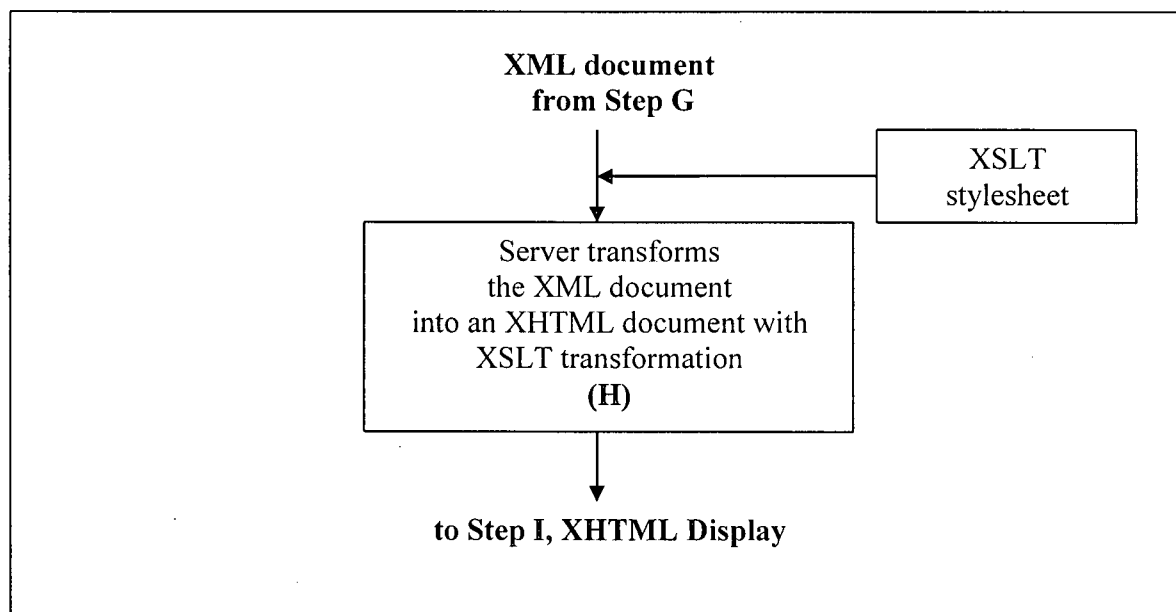


Figure IV-7: Step H, XSLT Transformation

A description of the entire work flow is given below: Initially, the database stores the original content elements and their alternate elements and the Content-to-Level Conversion table. With them, the Web server can associate each original element and its alternate elements with the appropriate levels. The database also contains the Environment-to-Level Conversion table which will be used later in the Channel Detection and Device Detection. Then, as the block diagrams demonstrate, starting from A, the client initiates an HTTP request for a certain Web page (the HTTP request header contains the device capability information). Arriving at Step B, the request is received by the Web server. Then, at Step C, the Web server extracts the device capability information from the HTTP request header. At the fourth step, D, the Web server obtains the channel conditions such as BW, delay, and jitter from the Channel Detection Module. Using the environment information from steps C and D, as well as the Environment-to-Level Conversion table, Step E computes the “environment supported level” which is relayed to the next step, F, where the Web server processes the data structure for each original content element to output scaled elements which are suitable for the current environment. At Step G, the selected elements are assembled into an XML document according to the logical structure of the Web page. Then, at Step H, the Web server applies XSLT transformation to convert the XML document into XHTML. Finally, at Step I, the XHTML document is delivered to the client as an HTTP response. The client then renders the XHTML document on its display. Through this work flow, the source content document in the Web server is scaled in accordance with the environment and is delivered over the channel and displayed at the client.

4.2 Extension to Address Personalization Issues

Our information architecture allows room for some personalization of the final presentation. By “personalization” we mean that the end-user can have his or her own preference of what type of media information to receive. For example, a person on his or her way to work wants to check the weather information with a PDA. The network connection is good and the PDA has a large screen that supports color graphics. Therefore, by default, the Web server will send a small-sized weather map with some colorful images representing the status of the weather. However, this person is walking rapidly and finds the graphic representation difficult to read. Instead, he or she would prefer a simple text version. How would our information architecture take such personalization into account and provide a truly user-satisfactory presentation?

A simple solution for an example such as this is for the Web server to communicate with the end-user and “learn” the “preferred level” of content. To do this, the Web server needs to inform the end-user of the “environment supported level” once it has been computed from the environment parameters. This is called the “initial environment supported level.” Assuming the user knows the classification of content levels, he or she replies to the Web server with a level that is either higher or equal to the initial environment supported level. This is the “final environment supported level” which will then be applied to the adaptation process. In this way, our information architecture can be extended to support scalability for user preference. Figure IV-8 illustrates this extended design to address the personalization issue.

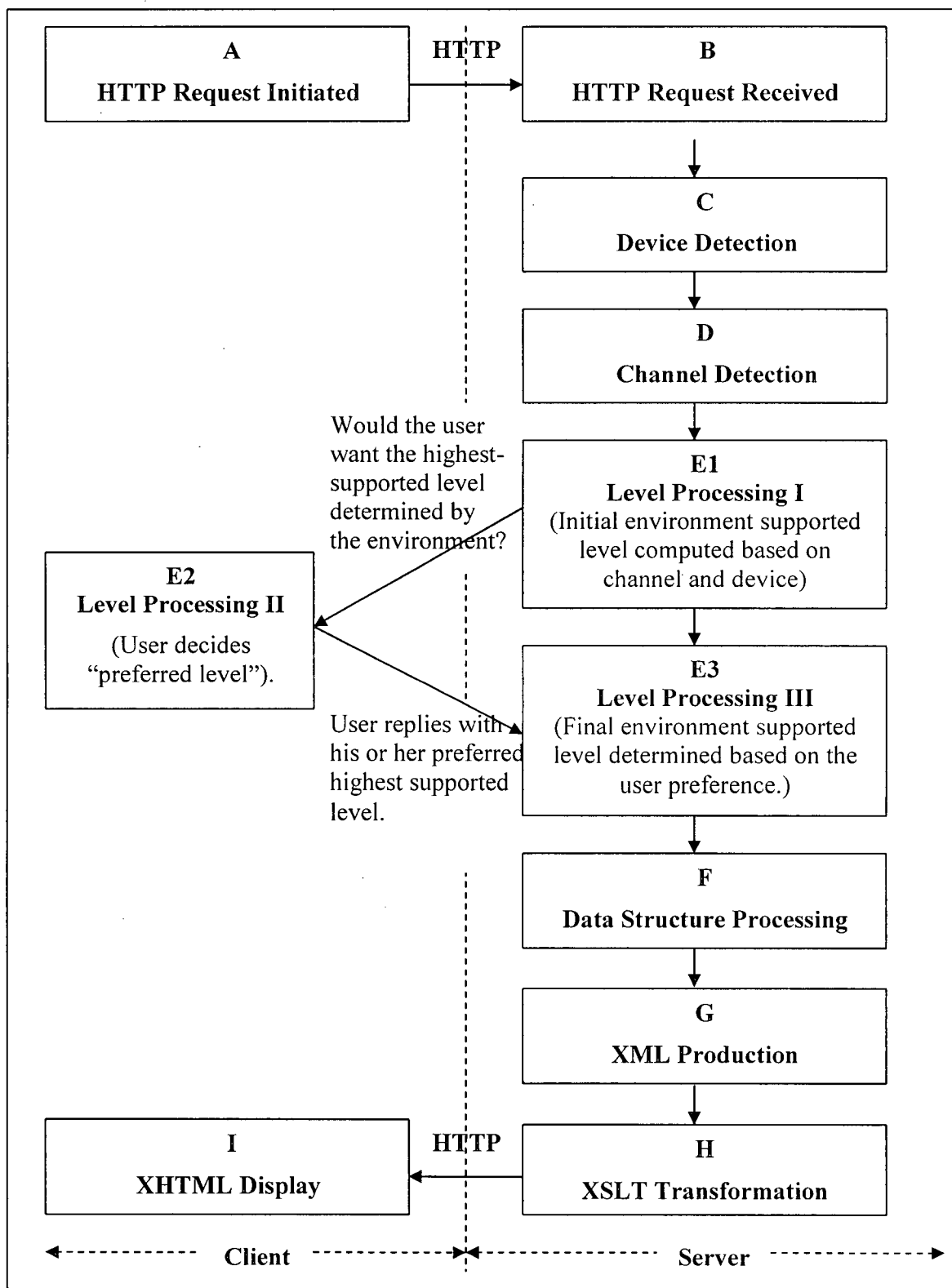


Figure IV-8: Information Architecture, with Personalization Scalability

4.3 Novelty and Advantages

As we have emphasized throughout this thesis, the data structure that we invented for the source content encoding is the essence of our overall information architecture. The novelty of the data structure gives our information architecture the advantage of achieving a high level of scalability. Compared to previous solutions, our information architecture is significantly more scalable because it can adapt source contents according to a wide range of environment factors: particularly, channel conditions, device capabilities, and user preferences, while the previous solutions were basically only capable of handling adaptation for a very limited number of device categories.

Another significant attribute of our information architecture is that it employs various emerging and open technologies relevant to scalable content creation. We have researched and utilized some of the latest technologies that give rise to content scalability: primarily, XML and XSLT in data adaptation, XHTML Modularization in encoding of the final presentation, and a unified Web/WAP programming model. All of these technologies are popular and non-proprietary, bestowing on our design a high potential of rapid commercialization in a variety of heterogeneous computing environments. In addition, these technologies are leading forces in the development of the mobile Internet, thus indicating that our information architecture has a strong viability both now and in the future.

4.4 Summary

We have achieved scalable content creation for the mobile Internet by constructing a novel information architecture that benefits from a unique data structure for the storage of original Web content. The tree hierarchy of the data structure empowers the information architecture to use XML and related technologies to obtain content scalability. Under this

framework, we can generate Web content that is scalable over communication channels, device capabilities, personalization issues, and media types. Minimal information loss and QoS of multimedia content are guaranteed as well. Additional advantages of our information architecture include the utilization of open technologies and ease of implementation. In the next chapter, we shall introduce a prototype of a Web-browsing application to demonstrate how this architecture is put into practice.

Chapter V: Empirical Evaluation

This chapter illustrates the use of the information architecture proposed in Chapter IV to achieve scalable content creation for the mobile Internet with a three-tier Web-browsing application. First, we introduce the three-tier application model which serves as the platform of our experimental evaluation. Second, we describe the configuration of the testbed network. Next, we explain the details of the implementation of the information architecture, including how our simplified data structure is represented in the database and how the overall work flow of the information architecture is realized through the use of Java Servlet. Then, we present the test cases and analyze their results. Finally, we discuss the possible optimizations and extensions that can be made to our information architecture.

To demonstrate how our information architecture achieves dynamic adaptation of multimedia Web content based on the changes in the environment, a prototype Web-browsing application called “Movie” was implemented and tests were conducted. The purpose of the “Movie” application was to present a Web page for the movie Titanic. A simple but common testing scenario was used: After receiving the request from the client, the Web server returns a multimedia Web page that describes the movie Titanic. Detailed information includes the movie title, an explanation of the story, information about the leading actor, and some multimedia contents such as a still image, a soundtrack, and an animated movie clip. We simulated the changes in channel conditions and device capabilities with some random functions and it will be shown that with these changes, the “Movie” page returned to the end-user changes dynamically, creating a suitable user experience for the particular environment under which the adaptation was made.

5.1 Three-tier Web Application Model

A three-tier application is “an application that is organized into three parts, each of which distributed to a different place or places in a network.” The three parts are: 1) the presentation interface at the client, 2) the business logic at the server, and 3) the database and the database management system. The presentation interface provides the user with the application-specific entry functionality with a graphical user interface (GUI), and application-specific entry forms or interactive windows. The business logic, usually located on a local area network (LAN) server that serves for client requests, determines what data is needed by the client and where the data is located. The third tier, including the database and the database management system, provides the source data needed for the application [81].

The advantage of the three-tier application model is the extensive scalability over operating systems, hardware and software, and programming languages used. The development, modification, and evolution of the application are therefore very convenient. This advantage can explain why the three-tier application model is the most commonly used model in Web applications. Therefore, we chose to develop our prototype Web application upon this model.

5.2 “Movie” Web-Browsing Application

5.2.1 Network Configuration

We built our “Movie” application on the three-tier model. The three components in our application are: the Web client, the Web server, and the database. The client makes a request for the Web content from the Internet through either a wired or wireless network, the Web server processes the request and performs the scalable content creation and returns the resultant page to the client for display, and the database contains the source content elements

of the Web page. Below is a diagram showing the configuration of these components, followed by a list of the development environment for each component. The implementation is discussed in detail in Sections 5.2.2 and 5.2.3.

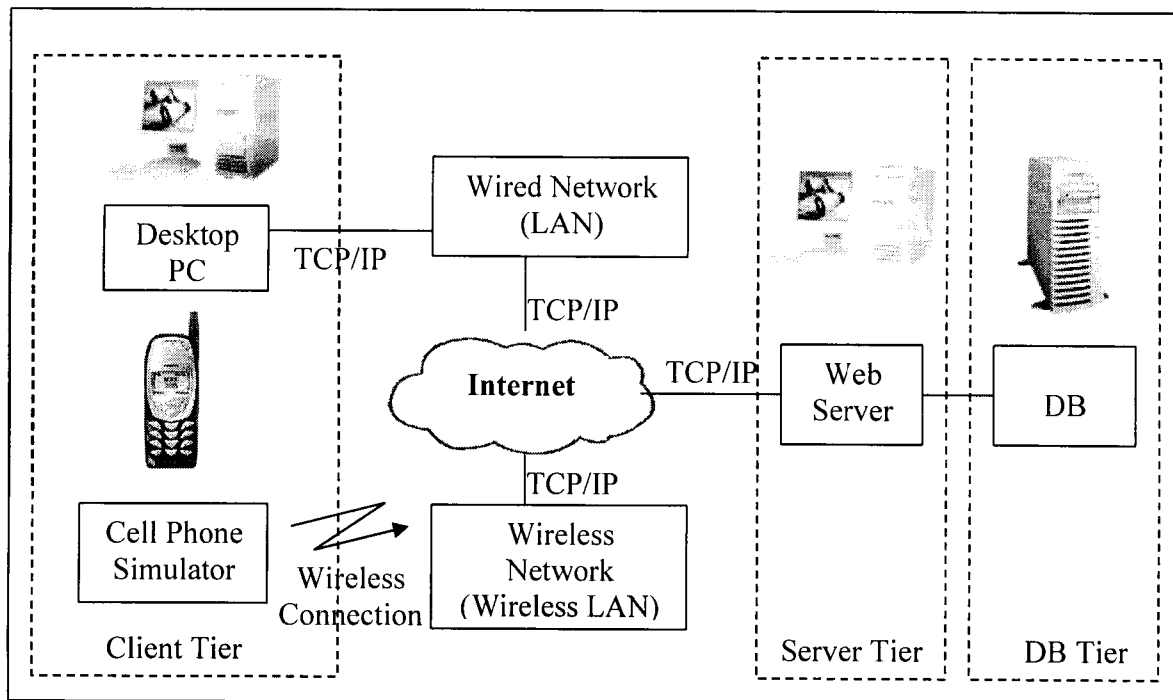


Figure V-1: Three-tier Application and Network Configuration of “Movie” Application

- The Web client was a laptop computer, playing a double role as both a thick and thin client.

As a thick client, the following application environment was set up:

- 1) Operating system: Windows XP Home.
- 2) Web browsers: Internet Explorer 6.0 and Mozilla 1.3.
- 3) Connection to the Internet: TCP/IP connection to the Department LAN of ECE, which has TCP/IP connection to Internet.

As a thin client, the application environment was:

- 1) Operating system: Windows XP Home.

- 2) Web browser: Nokia Mobile Internet Toolkit 3.1, which is a WAP 2.0 standard cell phone simulator acting as a wireless thin client with XHTML browsing capability.
 - 3) Java Runtime Environment (required by the Nokia Mobile Internet Toolkit 3.1): JRE 1.3.
 - 4) Connection to the Internet: Wireless connection to UBC Wireless LAN which has TCP/IP connection to the Internet.
- The Web server:
 - 1) Pentium desktop PC.
 - 2) Operating system: Windows XP Professional.
 - 3) Java development environment: J2SE 1.3.
 - 4) Web server platform: Apache Tomcat Web Server 4.0.6.
 - 5) Connection to the Internet: TCP/IP connection to the Department LAN (of the Electrical and Computer Engineering Department (ECE) of the University of British Columbia (UBC)), which has TCP/IP connection to Internet.
 - The database was a Microsoft Access 2002 Database residing on the same PC as the Web server, with JDBC as the interface between the Java class library and the physical database. (Although the database of the "Movie" Web-browsing application is a Microsoft Access Database which has several limitations, it is sufficient for our prototype development. In fact, our application can be applied to most relational databases including MySQL and Oracle).

5.2.2 Database Design

The database of the “Movie” Web-browsing application serves two major purposes: One is to use tables to store the static source content, pre-defined conversion relations, and dynamic environment parameters. The other is to run queries to output the levels of the content elements and the environment supported level, both of which are used by the content adaptation. All together, there are four tables and two queries to realize these functionalities. A detailed design of the tables and queries are depicted in Tables V-1 through V-6, each preceded by an explanation.

1. Table “Movie”

Corresponding to the function needed in Step F, Data Structure Processing, in the high-level design of our information architecture as shown in Figure IV-5, the “Movie” table stores the source content, including the original content elements and their alternate content elements. A partial view of this table is shown in Table V-1 below. Each record in this table is either an original content element or an alternate one and has an “ID” to identify itself within in the whole database and a “Data” to hold the real information of the element. There are four columns that describe the attributes of the element: 1) “ContentType” specifies the file type, 2) “FileSize” gives the size of the file, 3) “Encoding” indicates the encoding scheme of the file, and 4) “DisplayArea” means the least amount of area needed for a satisfactory rendering of that file. There are some other columns such as “HasAlternateObj,” “IDofAlternateObj,” “IDofImmediateParent,” and so on, which are used to help to “interleave” the elements so as to form our simplified data structure. To save time, we simplified some of the dynamic functions of the “Movie” application at the prototype stage. For example, the real information of all the original content elements and alternate content

elements was inserted into this table manually. In the future, certain scaling functions can be developed to create the alternate content elements dynamically. Another example, the file size, encoding scheme, and display areas were manually classified into “big,” “medium,” and “small” categories. Later, their exact values can be calculated by certain Java APIs so that the categorization will be more precise.

We present four content elements in the “Movie” table below. The first entry, ID 1, is “Titanic,” the name of the movie. Since it is an original content element, it has no immediate parent, thus its “IDofImmediateParent” is set to be “-1,” an indicator of non-existence. In addition, since “Titanic” is a small-sized ASCII text file requiring a small display, it is treated as an element at the “default level,” and, thus, there is no need for us to provide any alternate element for it. As a result, its “HasAlternateObj” is set as “no” and “IDofAlternateObj” set as “-1.” The second entry is “TitanicBanner.gif,” which is a small-sized image file requiring a big display area. Therefore, we need to provide a simpler version of this image, namely, the third entry “TitanicBannerSmall.gif,” which requires a small display area. The “IDofAlternateObj” of “TitanicBanner.gif” is “3,” which is the ID of “TitanicBannerSmall.gif”; the “IDofImmedParent” of “TitanicBannerSmall.gif” is “2,” which is the original content element “TitanicBanner.gif.” This cross-reference provides the basis for the interleaving of the original content elements and the alternate content elements. Finally, since a “default level” element is always needed for every original content element, the fourth entry, “TitanicBanner.txt,” is provided as a small-sized textual representation of the image “TitanicBanner.gif.”

ID	Data	IDofImmediateParent	HasAlternateObj	IDofAlternateObj	Content Type	File Size	Encoding	DisplayArea
1	Titanic	-1	no	-1	text	small	ASCII	small
2	TitanicBanner.gif	-1	yes	3	image	small	gif	big
3	TitanicBannerSmall.gif	2	yes	4	image	small	gif	small
4	TitanicBanner.txt	3	no	-1	text	small	ASCII	small
...

Table V-1: "Movie" Table (Partial)

2. Table "Content_Level_Conversion"

The next table in the database is "Content_Level_Conversion" which is predefined by the content provider. Corresponding to the function needed in Step F, Data Structure Processing in the high-level design of our information architecture as shown in Figure IV-5, this table associates the attributes of a content element with an appropriate level. A partial view of this table is in Table V-2 and it is self-explanatory. For example, if a content element is a text file of a "big" file size requiring a large display area, "level 1" is assigned to it. This level is a relatively high because it needs a device with a "big" display. On the other hand, if a text file has just a few characters calling for a small area for rendering, then "level 5" is assigned. In fact, level 5 content element is the short-text version which is supported under any environment. Worth noticing are the entries for "streaming" which is a highly resource-demanding media type. Thus, even when the file size and the display area needed are both small, the content level of this media type is set to be "0," which is the most complex content element in our implementation.

ContentType	FileSize	DisplayArea	ContentLevel
Text	big	big	1
Text
Text	small	small	5
Image	big	medium	2
Image
Image	small	small	4
Audio	big	medium	1
Audio
Audio	small	small	2
Video	big	big	0
Video
Video	small	small	2
Streaming	big	big	0
Streaming
Streaming	small	small	0
...

Table V-2: "Content_Level_Conversion" Table (Partial)

3. Query "CLC"

"CLC" stands for "Content Level Conversion." It is a query that contains the result of joining the "Movie" table with the "Content_Level_Conversion" table. Stored in this query are the content levels of all the content elements. The level information is used to determine whether a content element (either an original content element or an alternate content element) can be accommodated by the current environment. It corresponds to Step F, Data Structure Processing, in the high-level design of our information architecture as shown in Figure IV-5. Table V-3 below is a partial view of the CLC query in the "Movie" database. Corresponding to the "Movie" table shown in Table V-1, it can be seen that the movie title "Titanic" is assigned with a content level of "5," the movie image "TitanicBanner.gif" with "3," "TitanicBannerSmall.gif" with "4," and "TitanicBanner.txt" with "5."

ID	Data	ContentLevel
1	Titanic	5
2	TitanicBanner.gif	3
3	TitanicBannerSmall.gif	4
4	TitanicBanner.txt	5
10	leodPicGif.gif	2
11	leodPicGif_small.gif	4
12	Lenoardo DiCaprio	5
16	TitanicPic0.gif	1
17	TitanicPic0_medium.gif	2
18	TitanicPic0_small.gif	4
19	A picture from the movie	5
20	Titanic.mp3	2
21	TitanicPicForAV_medium.gif	2
22	TitanicPicForAV_small.gif	4
23	Titanic_mp3.txt	5
24	TitanicMovieClip8.mpeg	0
25	TitanicMovieClip1.mpeg	1
26	TitanicPicForAV_medium.gif	2
27	TitanicPicForAV_small.gif	4
...

Table V-3: "CLC" Query (Partial)

4. Table "Env_Level_Conversion"

Another table in our "Movie" database is the "Env_Level_Conversion" table. As the "Content_Level_Conversion" table, it is also predefined by the content provider.

Corresponding to the function needed in Step E, Level Processing of the high-level design of our information architecture as shown in Figure IV-4, "Env_Level_Conversion" table associates the environment with the highest level supported by that environment. For example, an MIDP device, such as a cell phone, can support content elements up to level 4, as long as the channel accommodates a wide BW, small delay, and small jitter. Level-4 contents are basically short texts and small images. On the other hand, given the same

channel conditions, a thick client such as a personal computer, can support content elements up to level 0, which is the most complex element type in our prototype implementation.

Level-0 contents include texts, images, audio files, video files, and streaming multimedia.

Table V-4 below presents a partial view of the table “Env_Level_Conversion.” In this table, the “Device” is either PC or J2ME profiles such as “MIDP” and “PersonalProfile,” all indicators of the device capability. “BW” is the channel bandwidth and “Delay” the end-to-end delay. To simplify the application development, we omitted “jitter,” a channel condition whose effects can be largely eliminated by introducing a setup delay. “SupportedLevel” is the environment supported level resulting from a combined consideration of both the device and channel.

Device	BW	Delay	SupportedLevel
PC	big	small	0
PC	medium	small	2
PC	medium	big	3
...
MIDP	big	small	4
MIDP	big	medium	4
MIDP	medium	small	5
...
PersonalProfile	big	medium	2
PersonalProfile	medium	small	3
PersonalProfile	small	small	4
...

Table V-4: “Env_Level_Conversion” Table (Partial)

5. Table “Env”

Generally, each time a client request is initiated, the environment changes. For example, a request can be initiated from a device with different capabilities, and/or under a communication channel with a different BW and delay. Thus, in our “Movie” database, a

dynamic table is needed to store the changing information of device capabilities and channel conditions. We created “Env” table to do this. Corresponding to the function needed in Step E, Level Processing of the high-level design of our information architecture as shown in Figure IV-4, the “Env” stores the environment parameters dynamically obtained from the Channel Detection Module. Then the Web server joins this table with the “Env_Level_Conversion” table in order to compute the environment supported level corresponding to this particular set of environment parameters. Next, the computed result is stored in the “ELC” query as explained in the next paragraph. Finally, since the information in the “Env” table is no longer needed, it is deleted right away. Table V-5 below is a “snapshot” of the “Env” table. As its content is only needed for an instant, this table is at most times empty.

Device	BW	Delay
--------	----	-------

Table V-5: “Env” Table

6. Query “ELC”

“ELC” stands for “Environment Level Conversion,” but is different from the “Env_Level_Conversion” table. It is a query that contains the result of joining the “Env” table with the “Env_Level_Conversion” table, that is, the environment supported level by a certain environment. Due to the same reason as in the “Env” table, the entry in the “ELC” query is cleared right after the environment supported level is retrieved by the Web server. Therefore, this query is usually empty, as shown in Table V-6. The “ELC” query is used in Step E, Level Processing of the high-level design of our information architecture as shown in Figure IV-4.

Device	BW	Delay	SupportedLevel
--------	----	-------	----------------

Table V-6: “ELC” Query

5.2.3 Java Servlet Design

In the “Movie” Web-browsing application, the information architecture is implemented with a number of Java classes that are driven by a Java servlet. Corresponding to the high-level design as shown in Figure IV-1 on page 74, the servlet is responsible for a series of functions. First, it catches the HTTP request from the client. Then, it drives the server-side functions to detect the channel conditions and the device capabilities. Next, it determines the environment supported level. After that, it interacts with the database to obtain the scaled content. At the next step, it creates an XML “Movie” page and transforms it into XHTML. Finally, it sends the XHTML document as the final response to the client. Figure V-2 on page 95 is the sequence diagram of the “Movie Servlet.” A detailed discussion follows this graph.

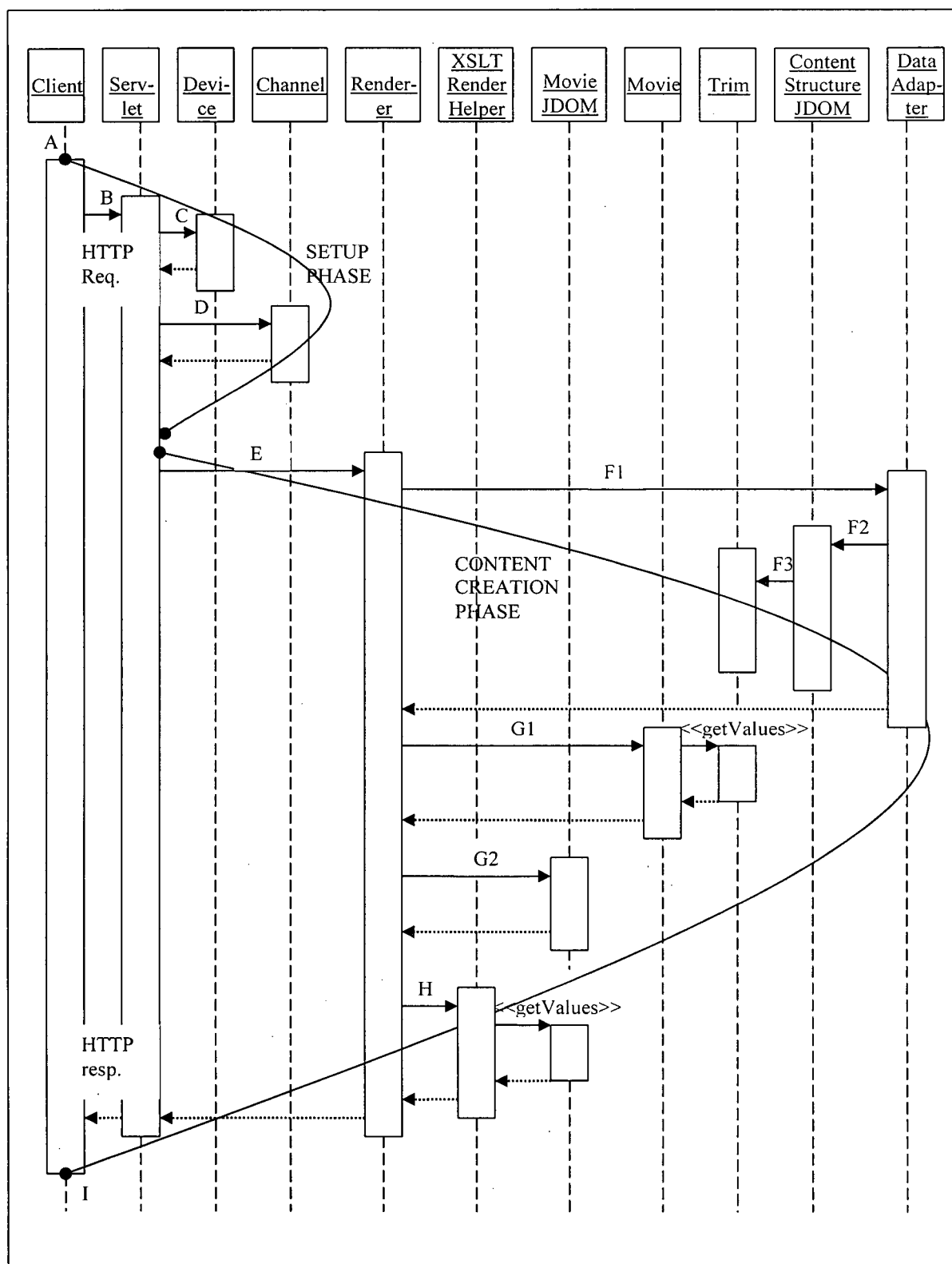


Figure V-2: Sequence Diagram of "Movie Servlet"

The whole servlet sequence is divided into two major phases: the Setup Phase and the Content Creation Phase. In the Setup Phase, the servlet performs the Channel Detection and Device Detection. In the Content Creation Phase, the main theme is to create a “Renderer” instance to handle all the work related to the following major functions in our information architecture: Level Processing, Data Structure Processing, XML Production, and XSLT Transformation. All the capital letters (such as A, B, and so on) in the sequence diagram correspond to those in the block diagrams in Chapter IV (Figures 1 through 7).

In the Setup Phase:

- A) The client initiates an HTTP request for the “Movie” Web page.
- B) The servlet catches the HTTP request and extracts the name/value pair that indicates the device capability from the HTTP request header. The most important index is the J2ME Profile of the device.
- C) The servlet creates a Device object to contain the device capability information.
- D) The servlet creates a Channel object that calls the Channel Detection Module in order to obtain the channel condition information. In our prototype, we simulated the channel with dummy functions in the Channel class. These functions return the random values of end-to-end BW, delay, and jitter, which are then input into the Channel object.

In the Content Creation Phase:

- E) The servlet calculates the environment supported level based on the environment parameters in the Channel and Device objects.
- F) The servlet creates a Renderer instance to handle several tasks, which are broken down into three components:

F1: The Renderer creates a DataAdapter instance to get all the content elements from the database.

F2: Each original content element and its alternate elements are linked by a ContentStructureJDOM class according to our simplified data structure. All these data structures are packaged into an XML document by the JDOM API. The result is an XML document in a tree structure.

F3: This tree is traversed and trimmed based on the environment supported level determined by the environment.

G) Based on the purpose of the “Movie” application, the Renderer creates the “Movie” Web page in XML in two steps.

G1: First, the Renderer creates a Movie object. It holds all the selected elements from the trimming process. The Movie class is application specific but simple.

G2: Second, the Renderer instantiates a MovieJDOM object which is responsible for converting data from the Movie object into a “Movie” document in XML based on the logical structure of the Web page.

H) The Renderer creates an XSLTRenderHelper instance to perform the XSLT transformation. The XSLTRenderHelper loads the XML document from the MovieJDOM, and a ready-made XSLT stylesheet from the file system, and transforms the XML into XHTML.

I) The XHTML “Movie” page is then delivered to the client in an HTTP response to be displayed.

Corresponding to the steps mentioned above, we deployed our “Movie” Web application and conducted browsing tests. The results are analyzed in the next section.

5.3 Results and Analysis


We expected our tests to show that with the changes in channel conditions and/or device capabilities, the “Movie” Web-browsing application automatically adapts the source content to generate Web pages at different scaling levels. To simulate changes in device capabilities, we used different Web browsers to represent the different devices. In our experiments, the Web client was a laptop computer. It acted as a thick client when we used Web browsers such as Internet Explorer and Mozilla. As for the thin client case, the laptop was installed with a popular mobile Internet simulation tool, Nokia Mobile Internet Toolkit 3.1, which conforms to the WAP 2.0 standard. To simulate variations in channel conditions, we first assumed that the channel detection framework we proposed in Figure II-10 was in place, so that it could output the channel condition statistics such as BW, end-to-end delay, and jitter. We used several dummy functions to simulate the output of these channel parameters.

The results of the five test scenarios below illustrate that, with the changes in the channel conditions and device capabilities, the original content is adapted dynamically before it is delivered to the client. For any individual content element that cannot be transmitted efficiently or rendered properly, an alternate element at the suitable level is selected as a replacement.

1. Test Case 1

Test Scenario: The laptop was connected to the ECE LAN. The laptop simulated a thick client with a large display. The channel conditions were high BW and small delay.


Result and Analysis: The environment supported level under this environment is 0. At this level, long texts, large and color images, audio files, and video files can all be delivered and accommodated. Once the Web page was rendered at the client display, the end-user could click on the link to the soundtrack in order to play the mp3 music (Titanic.mp3), or the link to the movie clip to play a large mpeg file (TitanicMovieClip8.mpeg). Figure V-3 below is a partial view of the Web page as seen by the end-user.



Titanic

The Movie

April 10, 1912. Technology had been delivering a steady stream of miracles for the better part of two decades and people were beginning to take this never-ending spiral of progress for granted. What better demonstration of humanity's mastery over nature than the launch of Titanic, the largest and most luxurious moving object ever built by the hand of man. But four-and-a-half days later, the world had changed. The maiden voyage of the "ship of dreams" ended in a nightmare beyond comprehension and mankind's faith in its own indomitable power was forever destroyed by uniquely human shortcomings: arrogance, complacency and greed. James Cameron's "Titanic" is an epic, action-packed romance set against the ill-fated maiden voyage of the R.M.S. Titanic, the pride and joy of the White Star Line and, at the time, the largest moving object ever built. She was the most luxurious liner of her era -- the "ship of dreams" -- which ultimately carried over 1,500 people to their death in the ice cold waters of the North Atlantic in the early hours of April 15, 1912. The journey of "Titanic" begins in the present, at the site of the ship's watery grave, two-and-a-half miles under the ocean surface. An ambitious fortune hunter (Bill Paxton) is determined to plumb the treasures of this once-stately ship, only to bring to the surface a story left untold. The tragic runs melt away to reveal the glittering palace that was Titanic as it prepares to launch on its maiden voyage from England. Amidst the thousands of well-wishers bidding a fond bon voyage, destiny has called two young souls, daring them to nurture a passion that would change their lives forever. Rose DeWitt Bukater (Kate Winslet) is a 17-year-old, upper-class American suffocating under the rigid confines and expectations of Edwardian society who falls for a free-spirited young steerage passenger named Jack Dawson (Leonardo DiCaprio). Once he opens her eyes to the world that lies outside her gilded cage, Rose and Jack's forbidden love begins a powerful mystery that ultimately echoes across the years into the present. Nothing on earth is going to come between them -- not even something as unimaginable as the sinking of Titanic. Also inhabiting this floating microcosm are Cal Hockley, played by Billy Zane, heir to a huge fortune and Rose's fiance and Ruth DeWitt Bukater, Rose's socially driven mother, played by Frances Fisher. Oscar winner Kathy Bates is featured as the ship's most colorful real-life passenger, Molly Brown. Other historic figures include Captain E.J. Smith (Bernard Hill), White Star Line's managing director J. Bruce Ismay (Jonathan Hyde), and master shipbuilder and primary architect of Titanic, Thomas Andrews (Victor Garber). Also participating in this devastating hand of fate dealt to the passengers of Titanic are an Italian emigrant named Fabrizio De Rossi (Danny Nucci), Jack's poor but determined best friend, and Spicer Lovejoy (David Warner), Cal Hockley's pitiless, loyal valet. Paramount Pictures and Twentieth Century Fox



At 23 years of age, LEONARDO DiCAPRIO (Jack Dawson) has developed into one of his generation's most gifted and versatile talents. DiCaprio earned an Academy Award® nomination for Best Supporting Actor at 19 for his portrayal of an ebullient, mentally impaired youngster in "What's Eating Gilbert Grape?" The actor's second film at the time, his performance also garnered awards from the National Board of Review, Chicago Film Critics and the Los Angeles Film Critics, as well as a Golden Globe nomination. DiCaprio made his film debut opposite Robert DeNiro and Ellen Barkin in "This Boy's Life" and has subsequently appeared in a variety of features, from the Western "The Quick and the Dead," with Sharon Stone and Gene Hackman, to "The Basketball Diaries," a harrowing account of a young man's spiral into heroin addiction, to "Total Eclipse," shot in Paris, opposite David Thewlis, in which DiCaprio played the poet Rimbaud. In 1996, he starred with Claire Danes in the critically acclaimed modern adaptation of "William Shakespeare's Romeo + Juliet," as well as the screen adaptation of the successful play "Marvin's Room," opposite Meryl Streep and Diane Keaton. DiCaprio will next co-star with Jeremy Irons, John Malkovich and Gerard Depardieu in "The Man in the Iron Mask," which recently completed filming in France. He has also joined the ensemble cast of Woody Allen's latest project, slated for release in 1998. A native of Los Angeles, DiCaprio began his acting career at age 14, appearing in commercials and educational films. He soon moved into parts on episodic television, including a role on "Growing Pains" in the popular series' last season. He also appeared in the short film "The Foot Shooting Party," released by Touchstone Pictures. Visit the Official Leonardo DiCaprio website at www.leonardodicaprio.com.

Multimedia Experience

Soundtrack

Titanic.mp3

Movie Clip

TitanicMovieClip8.mpeg

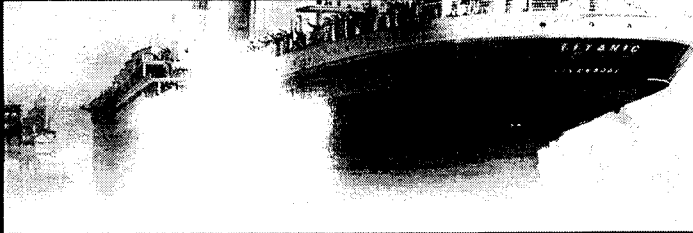
Figure V-3: Test Case 1, Content at Level 0 (Partial View)

2. Test Case 2

Test Scenario: The laptop was connected to the ECE LAN. It simulated a thick client with a medium display. The channel conditions were high BW and medium delay.

Result and Analysis: The environment supported level under this environment is 2. At this level, texts and color images up to medium file size are accommodated. The size of audio and video files, however, must not be too large if they are to be delivered. In this situation, the audio file *Titanic.mp3* is shipped without scaling, but the large video file *TitanicMovieClip8.mpeg* has to be adapted to its level-2 counterpart, a medium-sized color image *TitanicPicForAV_medium.gif*. When the page is rendered at the client display, the end-user can click on the soundtrack link in order to play the mp3 music (*Titanic.mp3*), or the movie clip link, not to see the animation, but a still image representing the movie clip. Figure V-4 on the next page is the Web page displayed at the client, and Figure V-5 is the image that replaces the original mpeg file.

Titanic




The Movie

James Cameron's "Titanic" is an epic, action-packed romance set against the ill-fated maiden voyage of the R.M.S. Titanic, the pride and joy of the White Star Line and, at the time, the largest moving object ever built. She was the most luxurious liner of her era -- the "ship of dreams" -- which ultimately carried over 1,500 people to their death in the ice cold waters of the North Atlantic in the early hours of April 15, 1912. The journey of "Titanic" begins in the present, at the site of the ship's watery grave, two-and-a-half miles under the ocean surface. An ambitious fortune hunter (Bill Paxton) is determined to plumb the treasures of this once-stately ship, only to bring to the surface a story left untold. The tragic ruins melt away to reveal the glittering palace that was Titanic as it prepares to launch on its maiden voyage from England. Amidst the thousands of well-wishers bidding a fond bon voyage, destiny has called two young souls, daring them to nurture a passion that would change their lives forever.

Starring

Lenoardo DiCaprio



At 23 years of age, LEONARDO DiCAPRIO (Jack Dawson) has developed into one of his generation's most gifted and versatile talents. DiCaprio earned an Academy Award® nomination for Best Supporting Actor at 19 for his portrayal of an ebullient, mentally impaired youngster in "What's Eating Gilbert Grape?" The actor's second film at the time, his performance also garnered awards from the National Board of Review, Chicago Film Critics and the Los Angeles Film Critics, as well as a Golden Globe nomination.

Multimedia Experience

Soundtrack

Titanic.mp3

Movie Clip

TitanicPicForAV_medium.gif

Figure V-4: Test Case 2, Content at Level 2



Figure V-5: An Image as Alternate Content Element for a Video File

3. Test Case 3

Test Scenario: The laptop was connected to the ECE LAN. It simulated a thick client with a medium display. The channel conditions were medium BW and long delay.

Result and Analysis: The environment supported level under such an environment is 3. At this level, texts of medium size can still be accommodated but the images file should be small, both in file size and display area. Neither audio information (Titanic.mp3) nor video (TitanicMovieClip8.mpeg) can be delivered. Rather, their alternate content element, a level-3 small-sized still image, TitanicPicForAV_small.gif, is delivered and displayed. Figure V-6 below is the resultant Web page displayed at the client, and Figure V-7 is the image file that replaces the original mpeg file as well as the mp3 file.

Titanic


LEONARDO DiCAPRIO **TITANIC** KATE WINSLET LEONARDO DiCAPRIO **TITANIC** KATE WINSLET

The Movie

James Cameron's "Titanic" is an epic, action-packed romance set against the ill-fated maiden voyage of the R.M.S. Titanic, the pride and joy of the White Star Line and, at the time, the largest moving object ever built. She was the most luxurious liner of her era -- the "ship of dreams" -- which ultimately carried over 1,500 people to their death in the ice cold waters of the North Atlantic in the early hours of April 15, 1912. The journey of "Titanic" begins in the present, at the site of the ship's watery grave, two-and-a-half miles under the ocean surface. An ambitious fortune hunter (Bill Paxton) is determined to plumb the treasures of this once-stately ship, only to bring to the surface a story left untold. The tragic ruins melt away to reveal the glittering palace that was Titanic as it prepares to launch on its maiden voyage from England. Amidst the thousands of well wishers bidding a fond bon voyage, destiny has called two young souls, daring them to nurture a passion that would change their lives forever.

Starring

Lenoardo DiCaprio



At 23 years of age, LEONARDO DiCAPRIO (Jack Dawson) has developed into one of his generation's most gifted and versatile talents. DiCaprio earned an Academy Award nomination for Best Supporting Actor at 19 for his portrayal of an ebullient, mentally impaired youngster in "What's Eating Gilbert Grape?" The actor's second film at the time, his performance also garnered awards from the National Board of Review, Chicago Film Critics and the Los Angeles Film Critics, as well as a Golden Globe nomination.

Multimedia Experience

TitanicPicForAV_small.gif

Movie Clip

TitanicPicForAV_small.gif

Figure V-6: Test Case 3, Content at Level 3



Figure V-7: A Small Image as Alternate Content Element for a Video File and an Audio File

4. Test Case 4

Test Scenario: The laptop was connected to the UBC Wireless LAN. Nokia Mobile Internet Toolkit was started on the laptop so that the laptop simulated an MIDP cell phone. The display area was small. The channel conditions were high BW and medium delay.

Result and Analysis: The environment supported level under this environment is 4. At this level, short texts and small images can be delivered and displayed, as shown in Figure V-8. Audio (Titanic.mp3) and video (TitanicMovieClip8.mpeg) contents are not suitable for the environment and are thus not delivered. Instead, their small-image alternatives (TitanicPicForAV_small.gif) are delivered and displayed, as shown in Figure V-9.

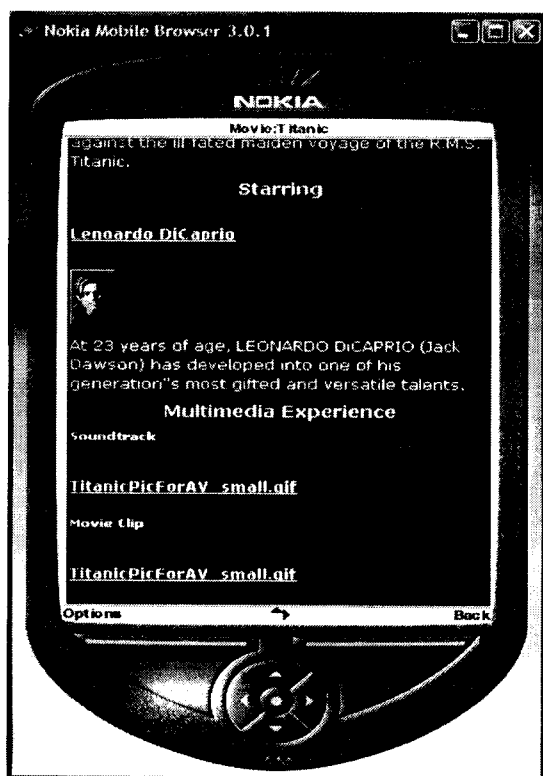


Figure V-8: Test Case 4, Content at Level 4

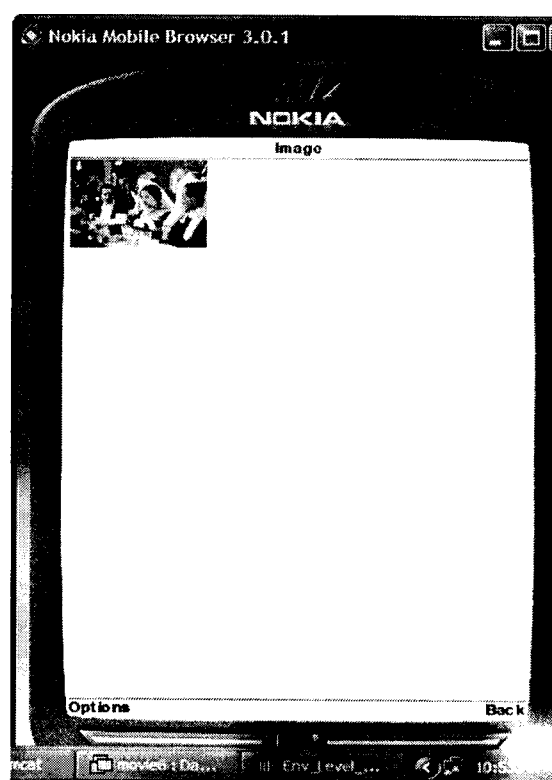


Figure V-9: A Small Image as Alternate Content Element for a Video File

5. Test Case 5

Test Scenario: The laptop was connected to the UBC Wireless LAN. Nokia Mobile Internet Toolkit was started on the laptop so that the laptop simulated an MIDP cell phone. The display area was small. The channel condition was medium BW.

Result and Analysis: The environment supported level under such an environment is 5. At this level, only short texts can be delivered and shown. Still or animated images, audio files, and video files cannot be shown. Instead, their short-text alternatives are used. For example, for the video file TitanicMovieClip8.mpeg, its alternate content element "TitanicMovieClip.txt" is delivered and displayed. Figure V-10 is the Web page displayed at the client and Figure V-11 the text file that replaces the original mpeg file and the mp3 file.

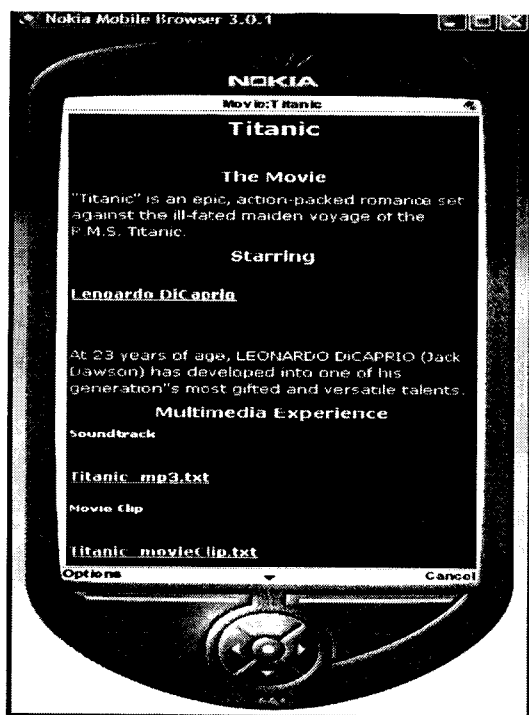


Figure V-10: Test Case 5, Content at Level 5

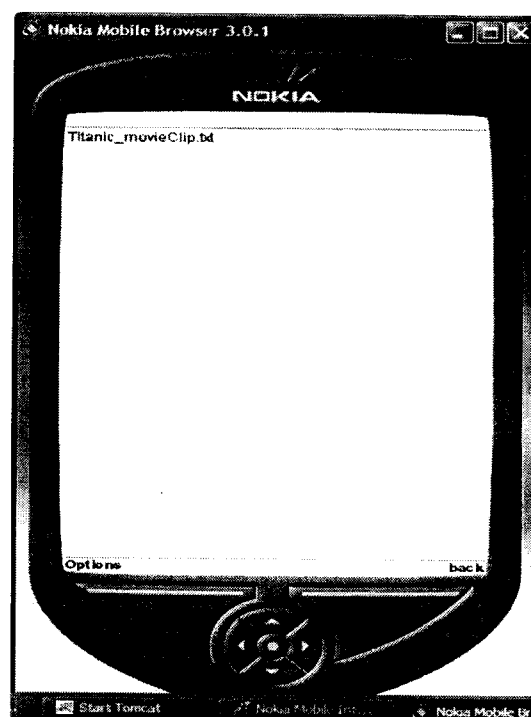


Figure V-11: A Short Text as Alternate Content Element for a Video File

To summarize, the five test cases above are typical scenarios of the adaptive content creation under various environment conditions. Their results clearly show that with dynamic changes in channel conditions and/or device capabilities, the same original content is adapted automatically before it is delivered to the client. Under optimal conditions, the content elements that require ample computing resources from the network and the client are maintained. When the communication channel deteriorates as, for instance, the BW drops and/or the end-to-end delay becomes longer, multimedia content elements such as graphic image, audio, and video are scaled to smaller file sizes or to another simpler media type, which then can be transmitted efficiently. When the device capabilities are less, for example, for a smaller screen, all content items occupying a large terrain are reduced in size so as to be rendered properly on the display. In all cases, the final “Movie” page that is returned to the client maximizes the user experience for that particular environment. The “Movie” Web-browsing application confirms that our information architecture has achieved our design objectives.

5.4 Discussions and Future Work

Our information architecture is capable of creating scalable content for the mobile Internet and has a strong potential to enable other types of scalable mobile Internet applications. The prototype that we deployed for empirical evaluation has revealed, however, that several implementation issues call for improvements. Additionally, a number of extensions to our design would make it even more useful. We will discuss these improvements and extensions in the following two sections.

5.4.1 Improvements to Our Information Architecture

We identify three improvements which can be made to our information architecture:

The first is that more research is needed on how to structure the adaptation rules for our general data structure so as to reach the best performance. In addition, a special set of XML tags should be developed for the data structure in order to link the adaptation rules in a standard way so as to ease the content encoding as well as the processing. To achieve this, further studies on DAG, XML, and how the environment factors can affect the multimedia need to be conducted.

The second improvement involves the employment of several dynamic functions to make our implementation suited for larger and more complex Web applications. When we discussed the “Movie” table in Section 5.2.2 “Database Design,” we pointed out that, due to time restrictions, we used manually-created alternate content elements in our prototype. This is favored for simple Web-browsing applications which need a limited number of content elements. However, for large and complex applications, the manual approach is not a good option because it would be too labor intensive and time-consuming. Content authors would expect some dynamic scaling functions to be built into the information architecture so that an original element could be converted into simpler versions automatically. Many such functions do exist and it is worth implementing them into our framework. Similarly, we used the manual way of determining the file type and file attributes for the content elements. To suit large and complex Web applications, dynamic functions should be used to determine the file size, display area, and encoding method.

The third improvement is to incorporate the content creation techniques proposed by other researchers. For example, the small information display of many terminals makes it

difficult to achieve a satisfactory user experience in content browsing. The proposed “navigation” model of Internet browsing for small devices can ameliorate the situation to a certain degree. If we integrate this model into our information architecture, we will see that, for an element that cannot be accommodated, its alternative will be displayed and in the format of a list of choices such as “email content,” “read content,” “print content,” “fax content,” and so on. This strategy will make our design more attractive to the end-users of very small devices.

5.4.2 Extensions to Our Information Architecture

Our information architecture can be extended in two ways: First, it would be a fascinating project to extend our information architecture from Web browsing to other Web applications. To reiterate, content browsing is the basis for all other Web applications, both in the wired and the wireless domains. To use an application or service, the users always need to locate them first. Additionally, whether they are gaming or e-learning applications, online “infotainment” or business services, Web applications invariably need to display Web content. Therefore, the “display” function is inevitable and is capable of making use of our approach to scalable content creation. For instance, the presentation of a widget for a board game can be very different on a PC screen from that on a cell phone. Using our information architecture, we can choose what image should be sent to the device to represent the widget. Certainly, the extension to other Web applications involve many issues not relevant to Web browsing, such as protocols other than HTTP, different markup languages other than XHTML, peer-to-peer communication instead of the client/server model, to name a few. In theory, since the core contribution of our information architecture is the data structure, which

is independent from communication models or protocols, it is hoped that positive extension results will occur.

Second, there are a few specific extension issues directly related to the implementation: 1) Due to time constraints, our prototype was developed to handle non-concurrent user accesses only. To handle multiple, concurrent accesses, we need to add multi-threading into our servlet and transplant our database from Microsoft Access to other more powerful relational databases such as MySQL or Oracle; 2) With the increase in the number of user accesses, performance can become a critical issue. The bottlenecks in the content creation process could occur when querying a large database and/or obtaining channel detection results. Cookie and caching speed the process, but it seems to need more than these two candidate techniques to reach and maintain a satisfactory performance level; 3) We used the idea of “Channel Detection API” proposed in Section 2.6.3 to obtain the conditions of the channel. Due to the focus of this thesis, as well as time restrictions, we simplified this part with dummy functions in our prototype. Nevertheless, it is important to implement the idea by using either simulation or emulation so that our information architecture can be tested in a “real” environment.

5.5 Summary

In this chapter, we present a prototype Web-browsing application, “Movie,” to demonstrate how our information architecture is put into practice. Test results are exhibited to show that our design can successfully achieve the research objective of scalable content creation. Moreover, we discussed the possible improvements and extensions that can be made to the information architecture to make it more practical and attractive. In summary, the implementation of our information architecture is easy, quick, and effective. It can be

concluded that our solution to scalable content creation for the mobile Internet is a successful one.

Chapter VI: Conclusion

In the rapidly expanding, lucrative business of the mobile Internet, scalabilities of Web content for multiple terminals over wireless connections is crucial to its success.

Previous solutions addressed this issue with XML related technologies and achieved positive results, but lacked a comprehensive consideration of the scalability problems introduced by the wireless communication system.

We have proposed a scalable content creation scheme for the mobile Internet by creating a novel data structure for Web content encoding and constructing an information architecture centered on this structure. We have implemented a prototype Web-browsing application that proves that our solution reaches the objectives, namely scalability over communication channels, devices capabilities, and media types. The most distinctive advantage of our invention arises from the unique tree form of the data structure, which enables our overall solution to leverage many scalable content creation techniques, especially XML. In addition, our proposed data structure is easy to implement and extend, and the information architecture is completely based on open technologies.

With the benefits mentioned above, our solution offers significant commercial appeal for Internet content providers, service developers, and telecommunication network operators, who create scalable multimedia contents for the mobile Internet. Furthermore, through extensions, our work can make a considerable contribution to the development of mobile Internet services and the ultimate value it brings to the general public will be enormous.

References

- [1] Mobile Systems Mobile Internet Home Page, 1 Jul. 2002, Ericsson, 16 July 2003 <http://www.ericsson.com/network_operators/mobilesystems>.
- [2] Introduction to 3G, 2003, [www.3G.co.uk](http://www.3g.co.uk), 16 July 2003 <<http://www.3g.co.uk/All%20About%203G.htm>>.
- [3] What You Can Do with 3G?, 2003, [www.3G.co.uk](http://www.3g.co.uk), 16 July 2003 <<http://www.3g.co.uk/What%20It%20Does.htm>>.
- [4] Charles Arehart, et al., Professional WAP (Wrox P, 2000) 11.
- [5] H. Kim, J. Kim, Y. Lee, M. Chae, and Y. Choi, An Empirical Study of the Use Contexts and Usability Problems in Mobile Internet, 35th Annual Hawaii Intl. Conf. on System Sciences (HICSS'02), Mobile Informatics, Research Concerning Mobile Information Technology Use (ETMIR), Big Island, HI, 2002. Vol. 5: 132.
- [6] Content, www.whatis.com: searchWebservices.com Definitions, 16 July 2003 <http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci211834,00.html>.
- [7] Multimedia, www.whatis.com: searchWebservices.com Definitions, 16 July 2003 <http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci212612,00.html>.
- [8] V. C. M. Leung, "Synopsis," WAP Research Project, Apr. 2001, 16 July 2003, <<http://www.ece.ubc.ca/~vleung/proposal.doc>>.
- [9] "Mobinet 2: A Rude Awakening For WAP Dreamers," Mobinet Study, 2001, A.T. Kearney, 16 July 2003 <http://www.atkearney.com/shared_res/pdf/Mobinet_2_S.pdf>.
- [10] "Mobinet Index #4," Mobinet Study, A.T. Kearney and The Judge Institute of Management at Cambridge U, Feb. 2002, 16 July 2003 <http://www.atkearney.com/shared_res/pdf/Mobinet_4_S.pdf>.

- [11] Scott Goldman, "Status of WAP," Presentation at WAP Australia, WAP Forum, May 2001, 16 July 2003 <<http://www.wapforum.org/new/WAPAustraliaMAY01.pdf>>.
- [12] Xueshan Shan and J. Jenny Li, A Case Study of IP Network Monitoring Using Wireless Mobile Devices, Proc. of 10th Intl. Conf. on Computer Communications and Networks, 2001. 590-593.
- [13] Raymond S. T. Lee, A New Era of Mobile Shopping Based on Intelligent Fuzzy Neuro-Based Shopping Agents, Proc. of Intl. Conf. on Consumer Electronics, 2001. 270-271.
- [14] Henry C. B. Chan, Irene S.K. Ho, and Raymond S. T. Lee, Design and Implementing of a Mobile Agent-Based Auction System Communications, Proc. of IEEE Pacific Rim Conf. on Computers and signal Processing, 2001. Vol. 2: 740-743.
- [15] K. K. Tan and C. Y. Soh, "Internet Home Control System Using Bluetooth Over WAP," Engineering Science and Education Journal, Vol. 11, Issue 4, (2002): 126-132.
- [16] N. M. Sgouros and S. Gerogiannakis, Integrating WAP-based Wireless Devices in Robot Teleoperation Environments, Proc. of IEEE ICRA, Washington DC, 2002. Vol. 2: 1191-1196.
- [17] P. d'Angelo and P. Corke, Using a WAP Phone as Robot Interface, Proc. of IEEE ICRA, Washington DC, 2002, Vol. 2: 1173-1178.
- [18] M. Hagleitner and T. A. Mueck, WAP-G: A Case Study in Mobile Entertainment, Proc. of HICSS, Wailea Maui, HI, 2001. 1115-1124.
- [19] T. Lilja, "Mobile Energy Supervision," Telecommunications Energy Conference, Proc. of INTELEC'00, Phoenix, AZ, pp. 707-712, Sept. 2000.

- [20] J. Parkka, M. Van Gils, T. Tuomisto, R. Lappalainen, and I. Korhonen, A Wireless Wellness Monitor for Personal Weight Management, Information Technology Applications in Biomedicine, Proc. of IEEE EMBS, Hangzhou, China, 2000. 83-88.
- [21] K. Hung and Y. T. Zhang, On the Feasibility of the Usage of WAP Devices in Telemedicine, Information Technology Applications in Biomedicine, Proc. of IEEE EMBS, Hangzhou, China, 2000. 28-31.
- [22] Wireless Markup Language Version 2 Specification, WAP-238-WML-20010911-a, Jul. 2001, WAP Forum, 16 July 2003
<[http://www.wapforum.org/ what/technical.htm](http://www.wapforum.org/what/technical.htm)>.
- [23] Wireless Application Protocol Architecture Specification, WAP-210-WAPArch-20010712-a, Jul. 2001, WAP Forum, 16 July 2003
<[http://www.wapforum.org/ what/technical.htm](http://www.wapforum.org/what/technical.htm)>.
- [24] K. H. Britton, et al., "Transcoding: Extending E-business to New Environments," IBM Systems Journal, Vol. 40, No. 1, (2001): 153-178.
- [25] McHenry, et al., Open Pluggable Edge Services Use Cases and Deployment Scenarios, Internet Draft draft-mchenry-opes-deployment-scenarios-00.txt, Jul. 2001, the Internet Engineering Task Force, 16 July 2003 <<http://standards.nortelnetworks.com/opes/non-wg-doc/draft-mchenry-opes-deployment-scenarios-00.txt>>.
- [26] Frank P. Coyle, "Wireless and XML," Cutter IT Journal, Vol. 14, No. 3, Mar. 2001, 16 July 2003 <<http://www.cutter.com/itjournal/2001toc.html>>.
- [27] Iftikhar Ahmed, "Enabling Web Applications for Wireless Devices," Atlas Software Technologies, Inc. Dec. 2000, 16 July 2003 <<http://www.sun.com/software/xml/developers/iftwireless/?redirect=false>>.

- [28] Luca Passini, "Building WAP Services: XML and ASP Will Set You Free," NewArchitect.com, Mar. 2000, 16 July 2003 <<http://www.webtechniques.com/archives/2000/03/passani/>>.
- [29] Michael A. Blackstock, "Markup for Transformation to Mobile Device Presentation Markups using XML," Master's thesis, School of Computer Science, Simon Fraser U, Vancouver, BC, Can., Nov. 2001.
- [30] Bill N. Schilit, Jonathan Trevor, David M. Hilbert, and Tzu Khiau Koh, "Web Interaction Using Very Small Internet Devices," IEEE Computer Society Digital Library, IEEE Computer, Vol. 35, No. 10: 37-45, Oct. 2002, 16 July 2003 <<http://csdl.computer.org/comp/mags/co/2002/10/rxtoc.htm>>
- [31] George Lawton, "Browsing the Mobile Internet," IEEE Computer Society Digital Library, IEEE Computer, Vol. 34, No. 12: 18-21, Dec. 2001, 16 July 2003 <<http://csdl.computer.org/comp/mags/co/2001/12/rztoc.htm>>.
- [32] SyncML Device Information DTD Version 1.1, 15 Feb. 2002, SyncML Initiative, 16 July 2003 <http://www.syncml.org/docs/syncml_devinf_v11_20020215.pdf>.
- [33] WAP Synchronisation Specification, WAP-234-SYNC-20010530-a, Version 30-May-2001, Jul. 2001, WAP Forum, 16 July 2003 <<http://www.wapforum.org/what/technical.htm>>.
- [34] Standard Generalized Markup Language (SGML), 12 Jul. 2002, Core Standards, the Cover Pages Web Site hosted by OASIS, 16 July 2003 <<http://xml.coverpages.org/sgml.html>>.
- [35] Origin and Goals, Extensible Markup Language (XML) 1.0 (Second Edition), Oct. 2002, W3C, 16 July 2003 <<http://www.w3.org/TR/REC-xml#sec-origin-goals>>.

- [36] Well-Formed XML Documents, Extensible Markup Language (XML) 1.0 (Second Edition), Oct. 2002, W3C, 16 July 2003 <<http://www.w3.org/TR/2000/REC-xml-20001006#sec-well-formed>>.
- [37] Related W3C Work HyperText Markup Language (HTML) Home Page, Jul. 2003, W3C, 16 July 2003 <<http://www.w3.org/MarkUp/>>.
- [38] XML Applications and Initiatives, 12 Jul. 2003, Core Standards, the Cover Pages Web Site, 16 July 2003 <<http://xml.coverpages.org/xmlApplications.html>>.
- [39] Web Style Sheets Home Page, Apr. 2003, W3C, 16 July 2003 <<http://www.w3.org/Style/>>.
- [40] XHTML 1.0 The Extensible HyperText Markup Language (Second Edition), a Reformulation of HTML 4 in XML 1.0, W3C Recommendation, Aug. 2002, W3C, 16 July 2003 <<http://www.w3.org/TR/xhtml1/#xhtml>>.
- [41] XHTML Modularization - an Overview, Apr. 2001, W3C, 16 July 2003 <<http://www.w3.org/MarkUp/modularization>>.
- [42] Modularization of XHTML - XHTML Abstract Modules, Apr. 2001, W3C, 16 July 2003 <http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html>.
- [43] Modularization of XHTML, W3C Recommendation, Apr. 2001, W3C, 16 July 2003 <<http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/Overview.html#toc>>.
- [44] XHTML Basic, W3C Recommendation, Dec. 2000, W3C, 16 July 2003 <<http://www.w3.org/TR/xhtml-basic/>>.

- [45] XHTML Mobile Profile Specification WAP-277-XHTMLMP-20011029-a, Jul.2001, WAP Forum, 16 July 2003 <<http://www.wapforum.org/what/technical.htm>>.
- [46] WAP Forum, 2003, WAP Forum, 16 July 2003 <<http://www.wapforum.org/>>.
- [47] What is WAP? 2002, Open Mobile Alliance, 16 July 2003 <<http://www.wapforum.org/what/index.htm>>.
- [48] WAP Forum Releases WAP 2.0 Specification for Public Review, Aug. 2001, WAP Forum, 16 July 2003 <<http://www.wapforum.org/new/wap2.0.pdf>>.
- [49] Wireless Application Protocol WAP 2.0 Technical White Paper, Jan. 2001, WAP Forum, 16 July 2003 <http://www.wapforum.org/what/WAPWhite_Paper1.pdf>.
- [50] About the Java Technology, the Java Tutorial, May 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html>>.
- [51] What Can Java Technology Do, the Java Tutorial, May 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/docs/books/tutorial/getStarted/intro/cando.html>>.
- [52] Apache Tomcat, The Apache Jakarta Project, 2002, Apache Software Foundation, 16 July 2003 <<http://jakarta.apache.org/tomcat/>>.
- [53] Internet Information Services, Windows Server 2003, 2003, Microsoft Corp., 16 July 2003 <<http://www.microsoft.com/WindowsServer2003/iis/default.msp>>.
- [54] Fundamentals of Java Servlets, March 2003, MageLang Institute, the Java Tutorial, Sun Microsystems, Inc., 16 July 2003 <<http://developer.java.sun.com/developer/onlineTraining/Servlets/Fundamentals/introduction.html>>.
- [55] Java API for XML Processing (JAXP), Jun. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/xml/jaxp/>>.

- [56] Elliotte Rusty Harold, Processing XML with Java, April 2003, 16 July 2003
<<http://cafeconleche.org/books/xmljava/>>.
- [57] Pawan Goyal, Simon S. Lam, and Harrick M. Vin, "Determining End-to-End Delay Bounds in Heterogeneous Networks," Multimedia Systems (1997) 5: 157-163, Springer-Verlag, 1997.
- [58] Dmitri Loguinov and Hayder Radha, Effects of Channel Delays on Underflow Events of Compressed Video over the Internet, IEEE 2002 Intl. Conf. on Image Processing (ICIP), Rochester, NY, Sept. 2002.
- [59] Vern Paxson, "End-to-End Internet Packet Dynamics," IEEE/ACM Transactions on Networking, Vol. 7, No. 3, (1999): 277-292.
- [60] Jean-Chrysostome Bolot, End-to-End Packet Delay and Loss Behavior in the Internet, Proc. of SIGCOMM, 1993: 289-298.
- [61] Robert L. Carter and Mark E. Crovella, Measuring Bottleneck Link Speed in Packet-Switched Networks, Technical Report, TR-96-006, Boston U Computer Science Dept., Boston, MA, Mar. 1996.
- [62] Kevin Lai and Mary Baker, Measuring Bandwidth, Proc. of IEEE INFOCOMM, New York, NY, 1999: 235-245.
- [63] Bob Melander, Mats Björkman, and Per Gunningberg, A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks, IEEE Globecom - Global Internet Symposium, San Francisco, CA, Nov. 2000.
- [64] S. Ninda, et al., "Adaptation Techniques in Wireless Packet Data Services," IEEE Communications Magazine, (2000): 54-64.

[65] Pathchar tool in Network Research Group (NRG) of the Information and Computing Sciences Division (ICSD) at Lawrence Berkeley National Laboratory (LBNL) in Berkeley, California, Sept. 2002, 16 July 2003, <<http://ee.lbl.gov/>>.

[66] Alexander G. Palos, Identification of the Internet End-to-End Delay Dynamics Using Multi-Step Neuro-Predictors, WCCI/IJCNN, 2002, <<http://bioinfo.cpgei.cefetpr.br/anais/WCCI02/IJCNN02/PDFFiles/Papers/1049.pdf>>.

[67] L. Cheng and I. Marsic, Wireless Awareness for Multimedia Applications, Proc. of the IFIP Intl. Conf. on Communication Technologies, Beijing, China, 2000: 1376-1382, <<http://www.ifip.or.at/con2000/icct2000/icct436.pdf>>.

[68] F. Yu and V. C. M. Leung, "Connection Admission Control for PCS-to-Internet Protocol Internetworking," International Journal of Wireless Information Networks, Vol. 9, No. 1: 1-12, Jan. 2002, 16 July 2003 <<http://www.cwins.wpi.edu/journal.html/>>.

[69] Byoung-Jo J. Kim, A Network Service Providing Wireless Channel Information for Adaptive Mobile Applications: Part I: Proposal, Proc. of IEEE Intl. Conf. on Communications (ICC), Helsinki, Jun. 2001.

[70] Java 2 Platform, Micro Edition Data Sheet, Jan. 2001, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/j2me/j2me-ds-0201.pdf>>.

[71] 2 New 3G Handsets, Jul. 2003, www. 3g.co.uk, 16 July 2003 <<http://www.3g.co.uk/PR/July2003/5616.htm>>.

[72] The K Virtual Machine, Apr. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/cldc/ds/>>.

[73] Mobile Information Device Profile (MIDP), Apr. 2003, Sun Microsystems, Inc., 16 July 2003 <<http://java.sun.com/products/midp/>>.

- [74] "Wireless Java Carrier and OEM Support Guarantees Ubiquity," Zelop Group, Feb. 2003, 16 July 2003 <<http://www.zelosgroup.com/reports/JavaReportTOC.pdf>>.
- [75] Image Processing Technologies, Jun. 2003, VIMAS Image Processing, 20 September 2003 <http://www.vimas.com/ve_imm.htm>.
- [76] ImageMagick.org, ImageMagic.org, 20 September 2003 <<http://www.imagemagick.org>>.
- [77] Animation and Video Tools, Sept. 2003, DownLoad32.com, 20 September 2003 <<http://www.downlinux.com/windows/Graphics/AnimationandVideoTools/>>.
- [78] Dragomir R. Radev, "Text Summarization," Tutorial ACM SIGIR, New Orleans, Louisiana, Sept. 2001, 20 September 2003 <<http://www.si.umich.edu/~radev>>.
- [79] J. D. Schlesinger, J. M. Conroy, M. E. Okurowski, and D. P. O'Leary, "Machine and Human Performance for Single and Multidocument Summarization," Intelligent Systems, IEEE, Vol. 18, Issue 1: 46-64, Jan./Feb. 2003.
- [80] Inderjeet Mani, Text Summarization and Question Answering: Recent Developments in Text Summarization, Proc. of the tenth Intl. Conf. on Information and Knowledge Management, Oct. 2001.
- [81] 3-Tier Application, www.whatis.com: searchWebServices.com Definitions, 16 July 2003 <http://searchdatabase.techtarget.com/sDefinition/0,,sid13_gci211500,00.html>.

Appendix

List of Abbreviations and Acronyms

2G	Second Generation (Wireless Communication System)
3G	Third Generation (Wireless Communication System)
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASP	Active Server Page(s)
BER	Bit Error Rate
BREW	Binary Runtime Environment for Wireless
BW	Bandwidth
CC/PP	Composite Capability/Preference Profiles
CCD	Charge-Coupled Device
CD	Compact Disc
CDC	Connected Device Configuration
CDMA	Code Division Multiple Access
CDM	Channel Detection Module
CDPD	Cellular Digital Packet Data
CLDC	Connected Limited Device Configuration
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DAG	Directed Acyclic Graph
DOM	Document Object Model
DB	Database

ECE	Electrical and Computer Engineering Department (of UBC)
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hyper Text Transfer Protocol
IEEE	Institute of Electrical & Electronics Engineers
IP	Internet Protocol
J2ME	Java 2 Mobile Edition
J2SE	Java 2 Standard Edition
JAXP	Java API for XML Processing
Java VM	Java Virtual Machine
JDBC	Java Database Connectivity
JDOM	Java Document Object Model
JPEG	Joint Photographic Experts Group
JRE	Java Runtime Environment
JSR	Java Specification Request
KVM	Kilo Virtual Machine
LAN	Local Area Network
MAC	Media Access Control
MID	Mobile Information Device
MIDP	Mobile Information Device Profile
MP3	MPEG Audio Layer-3

MPEG	Moving Picture Experts Group
PC	Personal Computer
PDA	Personal Digital Assistant
NTT	Nippon Telegraph and Telephone Corporation
PERL	Practical Extraction and Reporting Language
QoS	Quality of Service
RAM	Random-Access Memory
RDF	Resource Description Framework
ROM	Read-Only Memory
SAX	Simple API for XML
SGML	Standard Generalized Markup Language
SyncML	Synchronization Markup Language
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UA	User Agent
UBC	The University of British Columbia
URL	Uniform Resource Locator
VCR	Videocassette Recorder
VoiceML	Voice Markup Language
W3C	World Wide Web Consortium
WAE	Wireless Application Environment
WAP	Wireless Application Protocol
WCI	Wireless Channel Information

WML	Wireless Markup Language
WSDL	Web Services Description Language
WTA	Wireless Telephony Application
XHTML	Extensible HyperText Markup Language
XHTML MP	XHTML Mobile Profile
XML	Extensible Markup Language
XPath	XML Path Language
XSL	eXtensible Stylesheet Language
XSL-FO	XSL Formatting Objects
XSLT	Extensible Stylesheet Language Transformations