Mobile Content Transformation using XSLT and its Evaluation

Hideharu Suzuki, Norihiro Ishikawa, Hidetoshi Ueno, and
NTT DoCoMo, Multimedia Laboratories
3-5, Hikarinooka, Yokosuka,
Kanagawa, 239-8536, JAPAN
{hideharu, ishikawa, hueno} @mml.yrp.nttdocomo.co.jp

Tetsuya Gotoh
TOSHIBA CORPORATION
iVALUE CREATION COMPANY
1-12-7, Shiba, Minato-ku,
Tokyo, 105-0014, JAPAN
Tetsuya.gotou@toshiba.co.jp

ABSTRACT
For the standardization of Wireless Application Protocol (WAP) 2.0, we proposed the functions specific to i-mode HyperText Markup Language (HTML) to the WAP Forum. As a result, we were able to assure the functional compatibility between the WAP 2.0 markup language and i-mode HTML. In order to technically verify the functional compatibility of the WAP 2.0 with i-mode HTML, we conducted the experiment on an automatic content transformation between both markup languages, using XSLT. This paper discusses the experiment results.

Categories and Subject Descriptors
D.3.3 [Programming Languages]: Language Classifications – design languages, extensible languages.

General Terms
Measurement, Performance, Experimentation, Standardization, Languages, Verification.

Keywords
XML, XSLT, Mobile Content, Transformation.

1. INTRODUCTION
The WAP Forum has been established to standardize protocols and application environments primarily for the purpose of enabling users to access Internet content from mobile phones and other mobile terminals. The WAP Forum initially specified the Wireless Markup Language (WML) compliant to the eXtensible Markup Language (XML) as the markup language for WAP clients.

The latest version, WML 1.3, is designed to be used with mobile terminals, which have a small screen and limited processing power. WML 1.3 is not compatible with HyperText Markup Language (HTML), which is widely used on the Internet. On the other hand, NTT DoCoMo i-mode adopts HTML as the markup language. i-mode HTML consists of the subset of HTML 2.0, 3.2 and 4.0, and its latest version is 3.0 [1]. The number of i-mode users has steadily increased since the service was launched in February 1999, and exceeded 3 million users as of October 2002. i-mode is rapidly penetrating the market as a social infrastructure, and its rapid diffusion is believed to be attributable to the fact that it adopted HTML, which is widely used in the Internet, rather than WML 1.3, which is a markup language specific to the wireless environment. For the further evolution of the mobile Internet, WAP specifications should be more tightly integrated with standards widely accepted in the Internet such as TCP/IP and HTML.

In consideration of the above, we proposed to the WAP Forum the standardization of next-generation WAP (WAP 2.x) based on the specifications standardized by the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C). This proposal was widely accepted at the WAP Forum, and the final specifications of the first version of next-generation WAP (WAP 2.0) were published in January 2002 [2] [3]. In June 2002, The WAP Forum was renamed as the Open Mobile Alliance (OMA) and reorganized for further standardization of the mobile Internet, by integrating it with affiliated organizations such as SyncML.

As the markup language, WAP 2.0 adopted the eXtensible HyperText Markup Language (XHTML), which is the next-generation markup language standardized by W3C. In the WAP 2.0 standardization process, we proposed the functions (e.g. marquee) specific to i-mode HTML to the WAP Forum. Consequently, all our proposals were accepted, and the functional compatibility of the WAP 2.0 markup language with i-mode HTML was assured thereby.

When the both markup languages would come into wide use in the mobile Internet, it is desirable for the WAP 2.0 users to display the existing i-mode HTML content. It is important to automatically transform i-mode HTML content into the WAP 2.0 markup language content. In order to technically verify the functional compatibility of the WAP 2.0 markup language with i-mode HTML, we conducted the experiment to automatically transform i-mode HTML content into the WAP 2.0 markup language content. To begin with, we did the experiment on a static content transformation using i-mode HTML sample content, and confirmed that the transformation is possible.
In order to verify the feasibility of content transformation technology based on XSLT, it is necessary to do the experiment on a dynamic content transformation with respect to dynamic content generated by the Common Gateway Interface (CGI), which is actually being provided to i-mode users. To this end, we did the experiment on a dynamic content transformation using Eki-mae Tanken Club, which is a flagship utility content of i-mode, with the cooperation of Toshiba iValue Creation Company. This paper discusses the results of the experiment on the automatic transformation of i-mode HTML content into the WAP 2.0 markup language content. In the third section, we introduce the WAP 2.0 markup language, and compare it to i-mode HTML concerning an example of document source and displaying. In the forth section, we describe a content transformation technology, a document formatting and a validation. In the fifth section, we describe the overview of content transformation tools we have used. In the last section, we give some conclusions and draw some perspectives related to transformation software tools.

2. RELATED WORKS
Converting Web pages into well-formed XML documents were discussed [4]. One of typical topic was that HTML tagging rules are loosely defined. For instance, empty elements didn’t have the character ‘/’ before closing the tag, attribute values were not enclosed by two similar quotation marks (either ‘ or “), and end tags were optional for some specific non-empty elements. The converting process involved the following operations, (i) syntactic mapping of HTML to XML, (ii) resolving ambiguity introduced by HTML tagging rules, (iii) handling errors that may occur due to improper usage of HTML by the authors. They examined the filter that is responsible for removing any ambiguity or error from web documents by converting them into XML format.

Already existing HTML applications can be transformed into WML applications for use on WAP-enabled devices, yet this process is not as simple as the alteration of the markup tags. A number of problems discovered with the transformation of complex HTML contents into simplified WML contents. The major problems of displaying tabulated data, hyperlinks, navigational aids, and user input are discussed, with possible solutions presented [5]. The other project studied the relevance of numerals and keywords for removing unimportant or meaningless contents. The original contents will be reduced and reorganized to fit the size of mobile phone screens, thus reducing the communication cost and enhancing readability [6]. These problems would be encountered if a HTML content that was fit for a relatively larger size screen of PCs transformed into a WML content that was fit for smaller size screen of mobile devices. We discussed the transformation between i-mode HTML into WAP 2.0 markup language. Therefore, we assumed that the screen size of the users terminal device is same before and after the transformation.

3. i-mode HTML AND THE WAP 2.0 MARKUP LANGUAGE
The markup language of WAP 2.0 can roughly be divided into the following three parts.
3.1 XHTML Mobile Profile

This is based on XHTML Basic [7] standardized by W3C as the subset of XHTML 1.1. In addition to XHTML Basic, XHTML Mobile Profile includes some functions such as the hr tag (the horizontal line that delimits content), which is widely used in i-mode HTML content.

3.2 WAP CSS

WAP CSS defines the subset for mobile terminals of the Cascading Style Sheet level 2 (CSS2) [8], which is a W3C standard for defining the font, color, and other styles of content. Moreover, it adds functions widely used in i-mode, such as the marquee (horizontal scroll), access key (selection of link by key entry) and input mode (automatic switching of Japanese Front End Processor (FEP)). Figure 1 shows the relations among i-mode HTML, XHTML Mobile Profile and related specifications. As shown in Figure 1, XHTML 1.0 is the redefinition of HTML 4.01 by using XML, and XHTML 1.1 the modularization of XHTML 1.0.

The key characteristics of the WAP 2.0 markup language in comparison to i-mode HTML are described below.

1) High Extensibility

The WAP 2.0 markup language can easily be combined with other XML languages, and can flexibly accommodate future extensions. For example, if it is combined with XML Signature [9], which is an electronic signature technology for XML standardized by W3C, it is possible to append an electronic signature to the WAP 2.0 markup language content.

2) Strictly-defined Grammar

As the grammar of the WAP 2.0 markup language is strictly defined compared to i-mode HTML, it is easier to develop a WAP 2.0 browser, compared with i-mode HTML browser. On the other hand, content developers are required to have precise content creation skills.

3) Separation of Document Structure and Style Presentation

The WAP 2.0 markup language defines the document structure and the style presentation separately. For the style presentation, it defines WAP CSS based on CSS2. By using WAP CSS, it is possible to specify the document style more flexibly than HTML.

Figure 2-1 and Figure 2-2 show an example of document source written in i-mode HTML and the WAP 2.0 markup language respectively. Figure 3 illustrates how the content is actually displayed. Although the content on the screen appears the same (Figure 3), the descriptions of the content are much different (Figure 2-1, Figure 2-2).
and the generation of a task, sending a HTTP method, etc. This
source anchor for a hyperlink, the assignments of an access key
Module specifies the User Agent Behaviour, the definitions of a
elements or upon timer expiration. The Context & Navigation
generating events when the user interacts with several WML
interaction with the user. The Events Module is capable of
maintain backward compatibility with WML 1.x. It consists of
WAP extensions are unique features specified in WAP to
between
Figure 4. Example of XSLT Stylesheet.
Therefore, it is necessary to transform content written in the
WAP 2.0 markup language into i-mode HTML so that the WAP
2.0 markup language content can be displayed accurately in
browsers those support only i-mode HTML, that are already
deployed in the market. Similarly, it is essential to transform
content written in i-mode HTML into the WAP 2.0 markup
language so that it can be accurately displayed in browsers that
support only the WAP 2.0 markup language. In order to meet
these requirements, we studied the XSLT content transformation
technology that would enable the automatic transformation
between i-mode HTML content and the WAP 2.0 markup
language content.

3.3 WAP extensions
WAP extensions are unique features specified in WAP to
maintain backward compatibility with WML 1.x. It consists of
the wml:card element on the Structure Module, the Events
Module and the Context & Navigation Module. The wml:card
element specifies a fragment of the document body. Each
wml:card element represents an individual presentation and/or
interaction with the user. The Events Module is capable of
generating events when the user interacts with several WML
elements or upon timer expiration. The Context & Navigation
Module specifies the User Agent Behaviour, the definitions of a
source anchor for a hyperlink, the assignments of an access key
and the generation of a task, sending a HTTP method, etc. This
paper didn’t use the WAP extensions because of its
incompatibility with the i-mode HTML.

4. CONTENT TRANSFORMATION TECHNOLOGY
Content written in HTML and XHTML can be presented in
terms of the parent-child relationship between tags, and can be
modeled based on a tree structure as a whole. Accordingly, by
considering such a content structure, it is possible to execute a
content transformation efficiently. We focused on the eXtensible
Stylesheet Language Transformations (XSLT) [10] as a content
transformation technology, because it considers the content
structure when transforming XML content. XSLT was originally
a part of eXtensible Stylesheet Language (XSL), which is a
stylesheet specification that offers functions similar to CSS. As
its content transformation function is highly versatile and
extremely powerful, it was defined as an independent
specification, as XSLT. We adopted XSLT as the technology of
transforming i-mode HTML content and the WAP 2.0 markup
language content for these reasons.

4.1 XSLT
Figure 4 shows an example of the XSLT stylesheet used when
transforming i-mode HTML content into the WAP 2.0 markup
language content. This example shows how the root tag, the title
tag and the font tag in i-mode HTML content are transformed
into the WAP 2.0 markup language content. The XSLT stylesheet is used to designate the part of XML contents to be transformed using the match attribute of the template tag, and specify the actual changes, which is described within the template tag.

4.2 Document Formatting and Validation

As explained above, due to the loosely defined grammar of HTML, many browsers are capable of displaying the HTML content even if its syntax is ambiguous (e.g., lack of end tag). However, HTML content needs to be written in a grammatically correct form, because its content structure is strictly parsed when transforming the HTML content using XSLT. For example, the attribute must be enclosed by double or single quotation marks as shown in Figure 2-2 (b), and the parent-child relationship between the tags must be properly described as shown in Figure 2-2 (c). The task of correcting the grammatical error of content in such a manner is called formatting, and HTML content that has been corrected through formatting is referred to as well-formed content. The correction of simple grammatical errors, such as adding a missing end tag, must be performed at this stage. XHTML content does not require document formatting because it is written according to its strictly defined grammar.

The definition of the grammar of the content, such as the hierarchy among elements and the type of attributes in the content, is referred to as Document Type Definition (DTD), whereas the task of checking the grammar as to whether the content is written in compliance with the definition of DTD is called validation.

Validation should be performed in order to ascertain that the content resulting from automatic content transformation has no grammatical errors.

5. OVERVIEW OF CONTENT TRANSFORMATION TOOLS

Since the WAP 2.0 markup language was specified to be functionally compatible with the i-mode HTML, it is theoretically possible to transform existing i-mode HTML content into the WAP 2.0 markup language content. However, it is important to prepare software tools necessary for content transformation and verification, to encourage users and content providers to migrate into new markup language environments such as the WAP 2.0 markup language content transformation technology is the key technology for such migration. It is extremely useful for preparing content generation and verification environments for a new markup language, based on the existing content development environment. The prototype software environment is described below, which is developed to verify the feasibility of content transformation using XSLT.

Since the content transformation prototype software was developed for the purpose of verifying the functional compatibility of the WAP 2.0 markup language with the existing i-mode HTML, the scope of transformation is limited to the subset of the WAP 2.0 markup language that functionally corresponds to the i-mode HTML. Figure 5 shows the software configuration for transforming i-mode HTML content into the WAP 2.0 markup language content. The transformation process consists mainly of three steps: (a) formatting, (b) XSLT transformation, and (c) validation. As the prototype software, a number of existing software was used to reduce the development time costs. Table 1 shows the existing software used in the prototype software. The rules for transforming i-mode HTML content into the WAP 2.0 markup language content (i.e., XSLT stylesheet) was newly defined using the standard functions of XSLT based on the specifications of i-mode HTML and the WAP 2.0 markup language.

<table>
<thead>
<tr>
<th>Table 1. List of Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Formatting</td>
</tr>
<tr>
<td>XSLT Engine</td>
</tr>
<tr>
<td>Validation</td>
</tr>
</tbody>
</table>

The transformation steps are as follows. Firstly, the content is formatted after converting the full-width kana characters and half-width kana characters based on Shift Japan Industrial Standard (SJIS) in the i-mode HTML content into 10-decimal character codes, because the formatting tool (i.e., HTML Tidy) does not support SJIS. Secondly, after formatting i-mode HTML content using HTML Tidy, the full-width kana characters and the half-width kana characters shown in the form of characters codes in the formatted i-mode HTML content are converted back into SJIS, and an XML declaration is added, which is necessary for the XSLT engine process. Finally, the formatted i-mode HTML content is transformed into the WAP 2.0 markup language content using the XSLT engine, and the transformed WAP 2.0 markup language content is verified. The above-mentioned SJIS conversion steps are not needed in case of English characters.

For the reverse-transformation of content, that is, the transformation of the WAP 2.0 markup language content into i-mode HTML content, a separate content transformation rule (i.e., XSLT Stylesheet) was defined in a similar manner. The reverse transformation involves only the XSLT transformation process. As the WAP 2.0 markup language is inherently in compliance with XML, formatting is not required. Validation is also
unnecessary because *i-mode* HTML is not in compliance with XML and therefore has no DTD.

6. CONTENT TRANSFORMATION EXPERIMENT AND ANALYSIS

6.1 Overview of Content Transformation Experiment

The experiment of transforming *i-mode* HTML content into the WAP 2.0 markup language content was performed with respect to two types of content: static content, which is already stored as a file in the content server; and dynamic content, which is generated by CGI and Server Side Include (SSI) in response to the user’s request. As part of the experiment using static content, a test was performed with respect to reverse transformation i.e. the re-transformation of the WAP 2.0 markup language content that had resulted from transformation, back into *i-mode* HTML content.

The transformation experiment using static content involved the measurement of: the success rate of the content transformation process itself; the success rate of transformation relating to tags and attributes; changes in the content size before and after transformation; and the time consumed at each step in the transformation process. The transformation experiment using dynamic content involved: the measurement of the transformation processing time by dividing it into the time consumed in acquiring the content and the time consumed in transforming the content; and the analysis of trends in the correlation between the content size and the transformation processing time.

As the content subject to evaluation, *i-mode*’s flagship utility content called Eki-mae Tanken Club (http://ekitan.com/ in Japanese for PC browser) was used. Eki-mae Tanken Club is one of the official sites of *i-mode*, which gives information about station in Japan. We selected several contents of Eki-mae Tanken Club in the vicinity of the Tokyo metropolitan area (about 10 stations).

Table 2 shows the contents used in the experiment that are classified into 5 service types: 1) guidance on how to transfer from station to station, 2) when the last train leaves the station, 3) the train schedule, 4) the weather at the station and 5) the directions to certain places in the vicinity. There are 249 contents that include the dynamically generated contents and 7 static contents (i.e. the top pages of each service). The content of Eki-mae Tanken Club also omitted some end tags to reduce the size of the content. Therefore, these contents are not always in compliance with the *i-mode* HTML specification strictly, but *i-mode* phones can display these contents properly.

6.2 Configuration of Experiment System

The experiment using static content was performed in a laptop PC in which the content transformation software was installed. The content used in the experiment was stored as a file in the local disk. A prototype WAP 2.0 browser was used to confirm how the WAP 2.0 markup language content is displayed on the screen.

Figure 6 illustrates the system configuration used for the experiment on the transformation of dynamic content.

The system consists of a laptop PC as a client in which the WAP 2.0 browser software was installed, the transformation server in which the content transformation software was installed, and the content server of Eki-mae Tanken Club. Each machine is connected with the same segment of Ethernet. The content server generates a HTTP response to the user’s request by using CGI programs and sends it to the transformation server. The transformation server relays an HTTP request sent from a client to the content server, passes the content received from the content server to the content transformation software in order to have it processed, and executes content transformation. Then, it transmits the transformed content to the client.

The dynamic content transformation process is as follows. At first, the client sends an HTTP request to the transformation server. The transformation server relays the HTTP request to the content server to get the *i-mode* HTML content (This process is called “fetch”).

---

<table>
<thead>
<tr>
<th>Service name</th>
<th>Number of contents for request</th>
<th>Number of contents for reply</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Transfer Guidance</td>
<td>34</td>
</tr>
<tr>
<td>2</td>
<td>Last-train Guidance</td>
<td>34</td>
</tr>
<tr>
<td>3</td>
<td>Train Schedule at Station</td>
<td>21</td>
</tr>
<tr>
<td>4</td>
<td>Weather at station</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>Maps for directions</td>
<td>Menu contents for station select: 3</td>
</tr>
</tbody>
</table>
The transformation server writes the received content into a temporary file (This process is called “input”). The transformation server invokes the transformation software and executes the transformation from the i-mode HTML content into the WAP 2.0 markup language content (This process is called “transform”). The transformation server sends the transformed content to the client (This is called “output”). The response time taken when the client accesses the content server via the transformation server is the total amount of time of the fetch, input, transform and output.

6.3 Experiment Results and Analysis

6.3.1 Success Rate of Content Transformation
Measurements were taken so as to determine the success rate of the content transformation process itself and the success rate of transformation with regard to individual tags and attributes. As for the content transformation process, transformation was deemed successful when all processes —formatting, XSLT transformation and validation— were carried out. Table 3 shows the result of the content transformation experiment. According to the measurement results of transformation processes performed with respect to 249 contents, the success rate of transformation was 100%, provided that the transformation was within the scope of tags and attributes specified in the i-mode HTML specification.

When transforming 249 contents, the formatting tool issued no error message, but 6775 warning messages were issued. Table 4 shows the nature of the warning messages that were actually issued. The process associated with each warning message is as follows.

(a) The message indicating the replacement of “&” with entity reference form was issued with respect to all contents. It indicates that “&” was included in the value of the href attribute in the A tag and was replaced with “&amp” in compliance with the XML specifications.

(b) The message indicating the correction of HTML structural abnormality shows that there was a block-level tag such as an hr tag in the pre tag even though such tag is not permissible in the HTML specifications, and that an end of the pre tag was inserted before the hr tag and a start of the pre tag was inserted after the hr tag in order to remove the hr tag from the inside of the pre tag.

(c) When the message indicating the addition of an unspecified mandatory attribute is issued, an empty value of the attribute is added.

(d) When the message indicating the finding of an undefined HTML attribute is issued, the attribute and its value are deleted.

The 100% success rate was also confirmed with respect to the reverse transformation, that is, the re-transformation of the WAP 2.0 markup language content that had been transformed from i-mode HTML content, back into i-mode HTML content.

Table 3. Experimental Result of Content Transformation

<table>
<thead>
<tr>
<th>Total number of contents</th>
<th>Number of successes</th>
<th>Number of failures</th>
<th>Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>249</td>
<td>0</td>
<td>100%</td>
</tr>
</tbody>
</table>

6.3.2 Comparison of Content Size

Figure 7 compares the content size before and after the transformation. The size of the transformed WAP 2.0 markup language content increases at most or less the constant size, regardless of the size of the original i-mode HTML content (approx. 340 bytes on average). There seems to be two reasons of the size increasing. The reason for the constant increasing is, due to the XML declaration (approx. 44 bytes), the DOCTYPE declaration (approx. 67 bytes) and the namespace declaration (approx. 45 bytes). The total amount of the constant increasing is about 156 bytes. The reasons for the variable increasing are due to the tag size increment caused by transforming i-mode HTML to the WAP 2.0 markup language, the addition of end tags and the double quotation marks for enclosing the attribute values, which can be omitted in i-mode HTML content and the replacement of “&,” “<,” “>” and other characters used as reserved characters in the XML specifications with “&amp,” “&lt,” “&gt” and other entity reference forms, respectively.

Figure 8 compares the size of tag, which includes the attributes, between i-mode HTML and the WAP 2.0 markup language respectively. The plotting data is reordered by the difference of tag size in the increasing order. As Figure 8 shows, with respect to the difference of tag size, there seems to be several typical characteristics. When a tag size does not have any attributes, the tag size of the WAP 2.0 markup language is equal to that of i-mode HTML. When a tag in the WAP 2.0 markup language has an attribute defined by WAP CSS, which has a large amount of characters, the tag size of the WAP 2.0 markup language is larger than that of i-mode HTML. As compared with the WAP 2.0 markup language against i-mode HTML, the tag size of the WAP 2.0 markup language, includes attributes is about 8 bytes larger than that of i-mode HTML on the average. Equation 1 is a simple approximate expression of the increased size (Y) after the transformation. In this equation, it is assumed that the increased size of content is proportional to the number of tags in content. Where X denotes “the number of tags”.

\[ Y = 8X + 156 \] (1)

Figure 9 compares the increased size of the transformed WAP 2.0 markup language content between the experimental values and the calculated values of Equation 1. As Figure 9 shows, the experimental value is not necessarily to be equal to the calculated values.

Table 4. Warning Message

<table>
<thead>
<tr>
<th>Message</th>
<th>Total Number of Messages</th>
<th>Number of Content Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replacement of “&amp;” with entity reference form</td>
<td>6246</td>
<td>249</td>
</tr>
<tr>
<td>Correction of HTML structural abnormality</td>
<td>537</td>
<td>114</td>
</tr>
<tr>
<td>Addition of an unspecified necessary attribute</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Finding of an undefined HTML attribute</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
6.3.3 Transformation Processing Time of Static Content

Figure 10 shows the time consumed in processing the transformation of static content. The formatting process accounts for about 10% of the total time including the SJIS processing, while the XSLT transformation process and the validation process accounted for approximately 45% each, which is a significant portion. The processing time tended to increase depending on either the content size or the number of tags and attributes in the content.

6.3.4 Transformation Processing Time of Dynamic Content

Figure 11 is the breakdown of the processing time classified in the above mentioned 5 service names. Figure 11 shows that the fetch time, the time taken for the transformation server to acquire the content from the content server, accounts for more over 50% of the response time. The response time depends heavily on the fetch time against the transform time. The fetch time seems to depend on a sort of content because the content was generated by CGI program. For instance, the directions content takes the time to generate map image attachments dynamically, so it takes a long time to perform the fetch process. The other way around, the transfer guidance (Possibilities) content takes very short delay time because its generating process is very simple.

Consequently, it is verified that our transformation software provides sufficient function and performance characteristics in case of the dynamic content and the static content. On the other hand, with respect to the dynamic content, as the fetch time taken to acquire the content accounts for a large portion of the total response time, it seems to be effective to cache the transformed content and reuse it for subsequent requests. And it seems to be important to omit the validation process in the transformation, because the validation process time becomes larger in proportion to the size of content and the number of HTML tags. As we applied the existing commercial server to the experiment, the feasibility of system reconstructions was verified, but the system performance might not be improved. The fetch time that achieved from the experiment might be strongly dependent on the system configuration.
transformation from the imode HTML into the WAP 2.0 markup language. The contradictions are, for instance, the lack of the indispensable elements, no li element in the ol element, the redundancy of the TITLE element that should be only one in a document, a block level element came into an inline element’s section after the transformation had executed, etc. It might be hard for the XSLT to solve the contradictions. It is very important to customize the formatting tool for the purpose of recovery of the contradictions.

8. ACKNOWLEDGMENTS
We would like to thank Dr. Hirotaka Nakano and Mr. Osamu Takahashi, DoCoMo Multimedia Laboratories, for allowing us to put into practice this joint research and experiment. This work was partially supported by TOSHIBA CORPORATION e-SOLUTIONS COMPANY, TOSHIBA IT-SOLUTIONS CORPORATION and DoCoMo i-mode Business Department.

9. REFERENCES

7. CONCLUSION
This paper discussed the results of the automatic content transformation experiment concerning i-mode HTML content and the WAP 2.0 markup language content. We confirmed that 100% of i-mode HTML content could be transformed into the WAP 2.0 markup language content, as far as the content Eki-mae Tanken Club provided by Toshiba iValue Creation Company is concerned. This content transformation experiment is conducted only on the base of 5 types of services for train transport navigation. Therefore, it is necessary the further experiment with much more sample contents to verify the feasibility of content transformation for the general use purpose. In the future, we plan to verify the feasibility of this transformation technology based on the evaluation of a wider range of content, and work on the development of transformation software tools toward its commercial use. Several logical contradictions depended on the ambiguous grammar syntax of the i-mode HTML might be happened after