Technology Reports

An Automatic Form Filling Function on Mobile Terminals

Chie Noda

An Automatic Form Filling function running on a mobile terminal that fills Web forms on the Internet with preset user data has been developed experimentally. This function improves browser usability when entering data in such online forms automatically. This article presents an algorithm suitable for resource-scarce mobile terminals and outlines the prototype system.

1. Introduction

Mobile Internet applications and services have been widely deployed and extensively explored in the last decade. An increasing number of mobile services including Internet shopping, booking accommodation, and membership registration have become available, and often require Web forms[*1] to be filled with users profiles. **Figure 1** shows an example of a Japanese HTML-based form filled with user data. In this example, each line from top to bottom requires entry of the name in kanji (Chinese characters), the name in katakana (Japanese alphabets for loan words) that show how to read, postal code, state, city, other parts of the address, telephone number, and mobile e-mail address. Many Web pages require the entry of such common user data. We have seen some promising data mining solutions and products for mobile terminals such as input prediction and conversion engines, e.g., T9®[*2] and Wnn®[*3] for Japanese, that considerably reduce the burden on the user when manually entering data by using limited user interfaces on a mobile terminal. These products are particularly effective in predicting character strings by matching the initial part of various input data (such as when writing emails) and converting hiragana (another type of Japanese alphabets for native Japanese

*1 Web form: A technology for building Web pages to provide a mechanism enabling the engine on a server to process information entered and selected by the client. In this article, it refers to the text entry input and selection fields included on a Web page.

*2 T9®: T9 and the T9 logo are registered trademarks of Tegic Communications Inc. of the United States.

Figure 1 Web form input (example)

words) into kanji characters. However, even with an input prediction and conversion engine, each individual field still has to be selected and certain data entered manually. These products are therefore unsuitable for entering user data of limited variety.

We therefore propose the Automatic Form Filling function running on a mobile terminal that fills forms in downloaded Web pages with the preset user data. This function can be implemented by specifying special attributes for Web forms. In this case, data is entered automatically into Web forms when these attributes are found. DoCoMo's My Profile function supported on some 903i Series mobile terminals adopts this method, although existing Web forms not utilizing these special attributes are incompatible. The Internet Engineering Task Force (IETF)[*4] has specified input field names for e-commerce use as Electronic Commerce Modeling Language (ECML) [1] however they are rarely used on the Internet, and do not accommodate Japanese.

Moreover, such products as 'gooID memory' provided by NTT Resonant Inc. for PCs and PDAs enable automatic form filling executing locally on a terminal without any particular need to specify attributes. Such products have not yet been implemented on mobile terminals.

Given these circumstances, an Automatic Form Filling function that is executed locally on a mobile terminal without requiring changes to Web forms has been experimentally developed.

Table 1 Various Name attributes in Web sites

Concept name	Web site A	Web site B	Web site C
First Name	shippingAddress.firstname	name	firstName
Last Name	shippingAddress.lastname	name	lastName
Address 1	shippingAddress.address1	address1	address1
Address 2	shippingAddress.address2	address2	address2
City	shippingAddress.city	address3	City
State	shippingAddress.state	address4	state
Zip	shippingAddress.zip	postcode	postalCode
Country	shippingAddress.countryCode	country	country
Phone number	shippingAddress.voice	telephone	phoneNumber
E-mail address	–	email	email

This article describes a form filling algorithm, its system overview, and the prototype developed for evaluation.

2. Characteristics of Web Forms and Definitions of Rule Syntax

An analysis of Web forms on the Internet, covering such points as the required types of user data, order of input fields, and context information before and after the input fields, reveals many similarities. Here, 'context information' refers to information within HTML related to the input fields. In practice it indicates information displayed on the browser (referred to as 'labels') and attribute information specified for the input fields within HTML source code. Rules for form filling can be extracted from this information. This chapter explains the characteristics of Web forms used in the generation of such rules, and describes the basic rule syntax and expansion of syntax into Input Group Rules.

2.1 Characteristics of Web Forms

The similarities of Web forms on the Internet requiring user data were analyzed. As an example, **Table 1** shows the types of user data required on three English-language Websites (the

*3 Wnn®: Wnn is a registered trademark of Omron Corporation.
*4 IETF: A standardization organization that develops and promotes standards for Internet technology. The technology specifications formulated here are published as RFCs (Request For Comment).

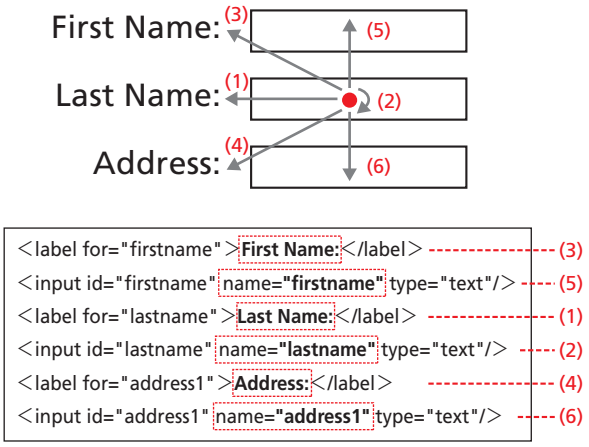


Figure 2 User data input fields and HTML source code (example)

'concept name') and name attribute specified for each input field. As described above, common user data is required. On Website A, name attributes beginning with the identifier 'shipping' are used. If we exclude the pre-identifiers, common sub-strings[*5] can be identified for the same user data type. It therefore becomes necessary to analyze various character strings and to find common sub-strings when extracting rules.

Rules are extracted from the similarities in context information related to the before and the after input fields in addition to the target input field, such as labels displayed on the browser, name attributes specified for each form in source code, and order of appearance of input fields. Based on common sub-strings used in labels and name attributes, the probability with which specific concept names are required must be determined by analyzing a large number of Web forms on the Internet.

2.2 Basic Rule Syntax

Rules specify which context information within HTML is to be used and the probability with which a certain concept name is required, in order to derive the optimum concept names for input fields on each Web form. **Figure 2** shows some of the input fields displayed on the browser, and the corresponding HTML source code. The six positions shown in Fig. 2 are context information used in predicting data entry. The label for the target input field (Fig. 2 (1)), and the name attribute specified for the input field itself (Fig. 2 (2)) are considered together with labels of input fields before and after (or left and right sides of) them (Fig. 2 (3) and (4)) and their name attributes (Fig. 2 (5) and (6)). Note that the definition of a label used here is not restricted to the <label> tag in the HTML source code, but the strings displayed before input fields on the HTML browser.

Rule syntax is defined as shown below in order to use the context information given in the six positions noted above.

Position | Condition | Value | = Concept Name | Probability
(If the information given in the six positions related to the target input field includes, or is the same as, value, the probability with which 'Concept Name *x*' is required is *y*%.)

According to this rule syntax, the rules applied to (2) and (3) in Fig. 2 are shown below as examples.
Example of rule applied to (2):

Current_Name_Attribute | Equals | lastname | = Last Name | 100
(If the name attribute in the target field is equal to 'lastname,' the probability of filling 'Last Name' in the target input field is 100%.)
Example of rule applied to (3):

Upper_Label | Contains | First Name | = Last Name | 70
(If the label of the immediately previous input field includes 'First Name,' the probability of filling 'Last Name' in the target input field is 70%.)

The analysis and generation of rules use a large number of Internet Web forms as source data. They may be automated in software. Rules related to labels and name attributes including these probability values are held locally on the mobile terminal.

When the information in the above-mentioned six positions is analyzed and multiple rules found to match the target input field, probability values are summed for each concept name. The concept name having the highest probability value is selected as the most probable user data to be filled in. This method provides greater accuracy than evaluating the rule syntax from single information.

*5 Sub-string: A part of a string of characters.

2.3 Input Group Rules

In addition to the simple rule syntax described in Section 2.2, a method of expanding rule syntax is explained here. The formats for using Japanese Web forms are more complex than for such European languages as English and German [2]. This complexity is primarily due to the following three reasons. First, there may be more than one expression for the same meaning, and multiple meanings for the same expression. The following shows different expressions for meaning 'name' found in 20 Japanese Web pages.

名前、お名前、なまえ、おなまえ、氏名、ご氏名、漢字氏名、カタカナ氏名、ローマ字氏名、姓名、セイメイ、宿泊者氏名

Secondly, the Japanese language uses multiple character sets–kanji, hiragana, katakana, and romaji (roman alphabets)–along with single-byte and double-byte characters for the latter two. It is often the case that entry of specific character sets is specified for Japanese input fields. In the example shown in Fig. 1, the name is required in kanji and katakana in the first and second lines respectively. Although katakana is not explicitly specified here for the second line, it is obvious here to input katakana since 'フリガナ' (providing information on how to read the kanji) after '氏名' (name) is written in katakana. In the '郵便番号' (postal code) and '電話番号' (phone number) input fields for the third and seventh lines in Fig. 1, the labels to the right of the input fields specify 'single-byte numerals' and provide examples. Thirdly, input fields are often split into multiple sub-fields on Japanese Web pages for further details in input data. In Fig. 1, the first, second, third, and seventh lines for name in kanji and katakana, postal code, and phone number are filled in multiple fields in the same row.

For the reasons described above, Japanese rules must specify not only the label immediately before, but also the multiple labels in the vicinity of the target input field, e.g., the multiple-labels on the left and right sides, and the relationships between the labels.

In the Japanese context, the existence of more than one expression for the same meaning, and the utilization of the combined multiple labels, pose difficulties in the generation of rules with high probability using common sub-strings. The Input Group Rule has been introduced to overcome these difficulties. Multiple input fields and related labels can be grouped based on their positional relationships. A concept name is selected for the entire group according to the Input Group Rule, with the concept name for each input field subsequently determined based on the Inner Group Rule. An input group may be detected, for example, by considering information in the same row in the table as a group based on the <tr> tags that indicate rows in the HTML table, or information located in the vicinity as a group based on coordinate information. The Input Group Rule is used to specify which labels appear within the group, although the order of labels and positional relationships with the target input field are not specified. Conversely, the Input Group Rule is used to predict the concept name to be entered in the input field based on the order in which the labels appear, or the number of input fields.

```
<tr>
   <td>
      <font size="-1"><b>氏名（漢字）</b></font>【必須】</b></font>
   </td>
   <td>
      <font size="2">（姓）</font> & nbsp;<input type="text" name="lname"
            maxlength="85" size="14" value=""> & nbsp; & nbsp;
   <font size="2">（名）</font> & nbsp;<input type="text" name="fname"
            maxlength="85" size="14" value="">
   </td>
</tr>
```

Figure 3 HTML source code (example)

Figure 3 shows an example of the HTML source code for the first row in Fig. 1. The section between the <tr> and </tr> tags is recognized as an input group. Since this input group includes '氏名' and '漢字' (kanji) labels, the following Input Group Rule is used to predict that 'Name_Kanji group' is a concept name of the whole.

Input Group | CONTAINS | 氏名 | Input Group | CONTAINS | 漢字 | = Name_Kanji group | 100

(When '氏名' and '漢字' labels appear in the group, the probability of the concept name being 'Name_Kanji group' is 100%.)

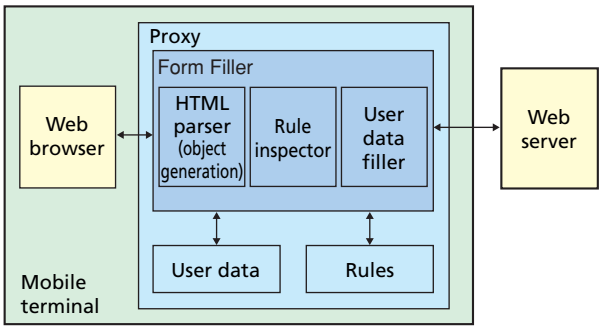


Figure 4 System architecture for Automatic Form Filling (example)

Since the order in which the labels appear is '姓' (last name) and '名' (first name), the following Input Group Rule is used to predict that a concept name for each input field is 'LastName_Kanji' and 'FirstName_Kanji,' respectively.

姓 | 名 | = LastName_Kanji, FirstName_Kanji
(When the labels appear in the order '姓' and '名,' the data is entered in the two input fields in the order LastName_Kanji, FirstName_Kanji.)

3. System Overview

Figure 4 shows an example of the system architecture for implementing the Automatic Form Filling function on a mobile terminal. The proxy on the mobile terminal mediates communications between the Web browser and Web servers. It stores user data corresponding to the concept names set by the user in advance, along with a list of rules as described in Chapter 2. When the Web page includes input fields, the Form Filler in the proxy analyzes related multiple context information, selects matching rules from the rule list held on the mobile terminal, generates a dynamic rule matching the context information, and derives the concept name with the highest probability. The result of filling the most probable user data in the input fields is then sent to the Web browser. This may also be introduced as a plug-in function for the Web browser instead of the proxy architecture shown in Fig. 4.

As shown in Fig. 4, the Form Filler is composed of three functional blocks. The HTML parser analyzes the syntax of the downloaded Web page, and then generates an object[*6] structure including the input fields, related context information, and positional information. The rule inspector function selects all rules matching the object structure, and generates a dynamic rule. The probability values specified in the selected rules are summed for each concept name and the concept name with the highest probability value is selected. Priority is given in applying the Input Group Rules to the input fields considered as input groups based on <tr> tags as described in Section 2.3. When an input group is found, the filled user data may be overwritten with its concept name. The user data filler function fills user data corresponding to the selected concept name. **Figure 5** shows an example of a sequence between a Web browser, a proxy for automatic form filling in Web forms, and a Web server, and proxy implementation procedures.

The advantages of this system are described below.

1) The proxy architecture allows the use of existing Web browsers and Web servers.
2) The proxy for form filling creates a dynamic rule in an optimized manner suitable for resource-scarce mobile terminals, based on a given set of context information related to the target input field on currently accessed Web pages and a list of pre-defined rules (as a result of the analysis of a large number of Web forms).
3) User data is stored and used locally on the mobile terminal to ensure privacy.
4) Automatically filled Web forms are displayed on the mobile terminal, enabling deletion and edition of data by user manually before being sent to the Web server.

4. Prototype Development and Evaluation

An English version of the prototype was developed as a Web browser proxy function using the Java 2 Micro Edition (J2ME™)[*7] on the Nokia 60 Series. **Figure 6** shows examples of a screen display when manually setting user data as a user profile to be stored on a mobile terminal and an automatically filled downloaded Web form. The algorithm has been verified as running on a resource-scarce mobile terminal with this prototype.

Furthermore, the prototype supporting both English and

*6 Object: An expression of something existing as an entity or concept in the real world in a form able to be handled in a program. Expressed as a combination of data describing its attributes, and manipulation of the entity.

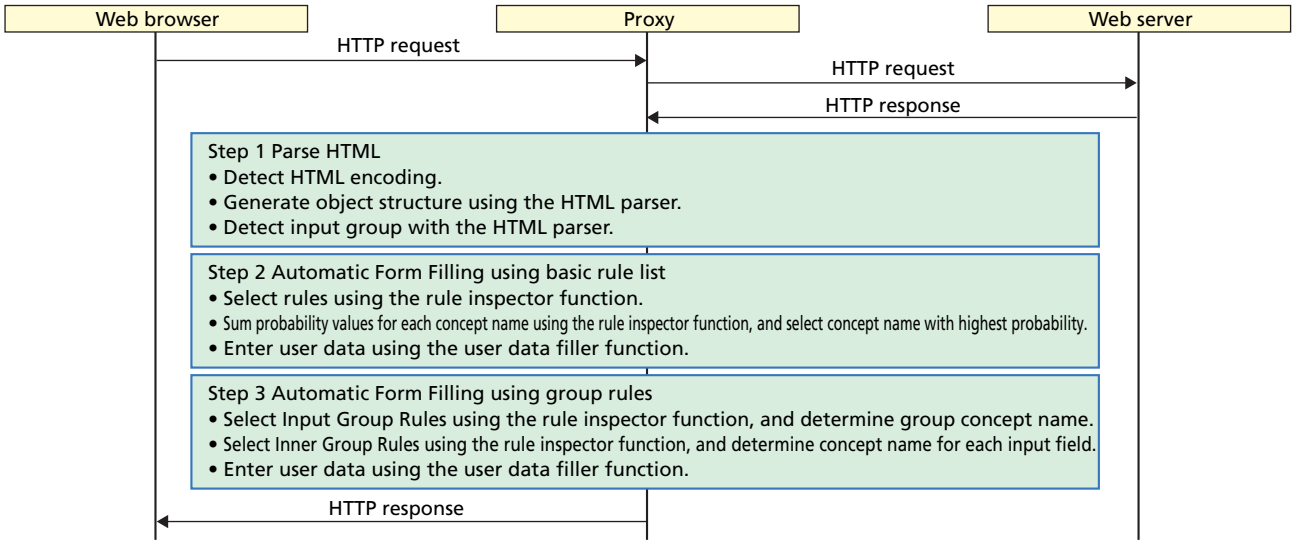
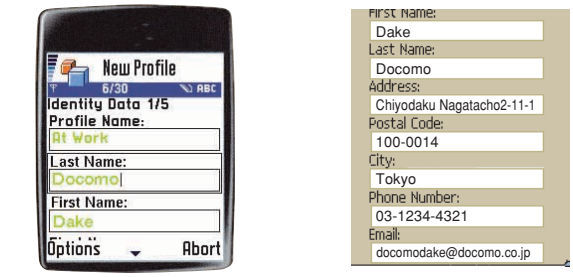


Figure 5 Sequence (example)

Japanese has been developed as a proxy for Web browsers using Java 2 Standard Edition (J2SE™)[*8] v1.4.2 on a PC. Concept names for name (full name, last name, first name in kanji, hiragana, and katakana), address (postal code, state, city, town, etc.), date of birth, email address (general, mobile), gender, phone number (fixed line, mobile), fax number, credit card details, and professional details are supported. More than 120 Web pages were analyzed for Japanese, and rules accommodating the <input> and <select> tags were generated.

An evaluation was made to determine the accuracy rate for automatic filling when using the prototype on a PC. Fifty Web sites not previously used for the generation of rules were randomly selected from a variety of different areas (e.g., Internet shopping, travel, hotels, membership registration) for both English and Japanese. The correct data was filled in 96.2% of all input fields for English, and in 79.0% for Japanese. These are the most accurate figures among all such commercial automatic form filling tools available for PCs, and represent a marked improvement over competitive products able to accommodate such a large number of concept names. The system is particularly suited to katakana names where input groups are applicable, achieving an improvement of more than 20% in accuracy rate in comparison with competing products.



(a) Screen display when entering user data (example)

(b) Automatically entered Web form (example)

Figure 6 English-language prototype on mobile terminal

5. Conclusion

This article has described a simple algorithm for the Automatic Form Filling function (regarding the generation of rules based on analysis of input fields on Internet Web pages) running on a mobile terminal, and an overview of the system. The algorithm relies on a set of pre-defined rules generated by analyzing a number of internet Web forms, and creates a dynamic rule by analyzing context information in a given Web form and selecting applicable rules. Also presented were the results of evaluating the accuracy rate of the English and Japanese versions using a prototype, which revealed a significant improvement over competitive products developed for PCs.

*7 J2ME™: A function set of the Java language. Reduces the consumption of resources for embedded devices.
J2ME and all Java-related trademarks and logos are trademarks and registered trademarks of Sun Microsystems, Inc. of the United States of America in the USA and other countries.

*8 J2SE™: A function set of the Java language. A collection of standard functions forming a foundation for network client devices (e.g., PCs).
J2SE and all Java-related trademarks and logos are trademarks and registered trademarks of Sun Microsystems, Inc. of the United States of America in the USA and other countries.

Future work will involve the evaluation of usability aspects other than the accuracy rate in order to improve ease of use (such as comparing time and the number of required strokes when using a prediction and conversion engine on a mobile terminal), and the user interface (such as for the visualization of uncertainty [3]).

The Automatic Form Filling function compatible with existing Web forms can be applied as an extension of DoCoMo's 'My Profile' running on some devices in the 903i Series while sharing previously registered user data.

REFERENCES

[1] IETF: ECMA (Electronic Commerce Modeling Languages), http://www.ietf.org/rfc/rfc3106.txt

[2] T. Chusho, K. Fujiwara and K. Minamitani: "Automatic Filling in a Form by an Agent for Web Applications," Asia-Pacific Software Engineering Conference 2002, IEEE Computer Society, pp.239-247, 2002.

[3] E. Rukzio, J. Hamard, C. Noda and A. De Luca: "Visualization of Uncertainty in Context Aware Mobile Applications," 8th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2006), Espoo, Finland, Sep. 2006.