Automatic Filling in a Form by an Agent for Web Applications

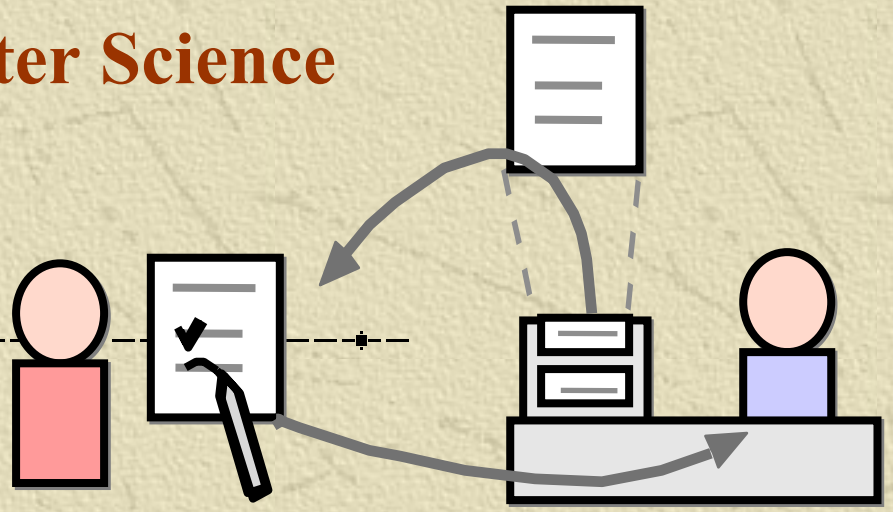
Takeshi CHUSHO, Katsuya FUJIWARA, and Keiji MINAMITANI

Department of Computer Science

Meiji University

Kawasaki, Japan

chusho@cs.meiji.ac.jp



Background of Research (1)

- **Trend**

  The number of end-users using the Internet increases on the inside and outside of offices.

- **Goal**

  Applications for web services should be supported by business professionals because web services must be modified frequently.

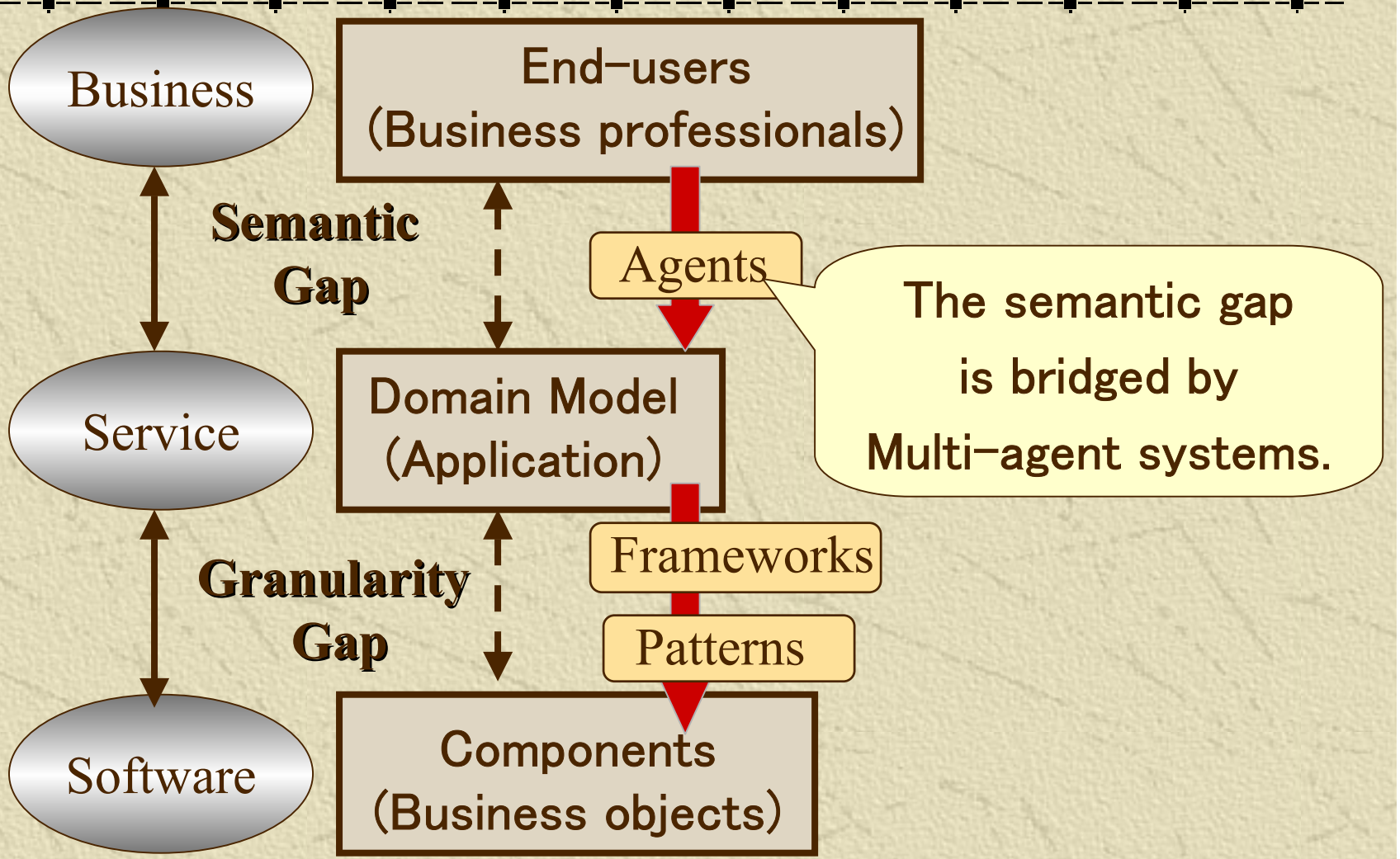
Background of Research (2)

- **Approach**
  - Form-based end-user computing
  - Applied technologies:
    - * Component-Based Software Engineering
    - * Multi-agent systems

- **One of sub-goals**
  - **Automatic filling in a form** by an agent in collaboration with a broker agent

Enduser-Initiative Approach

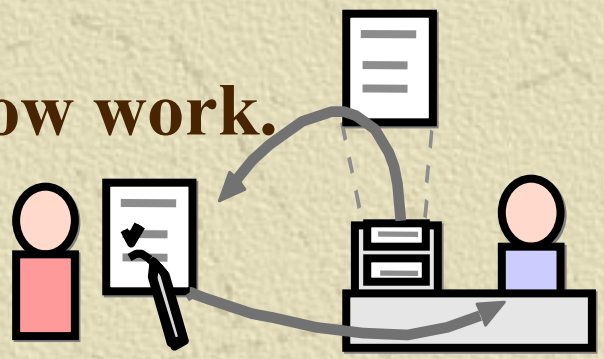
- how to make web applications -



Metaphors for Web Services
- Forms -

◆ Target Domain

- A typical distributed system : window work
- This is not limited to the actual window work.
  (Ex.) SCM can be considered as
  combination of the virtual window work.



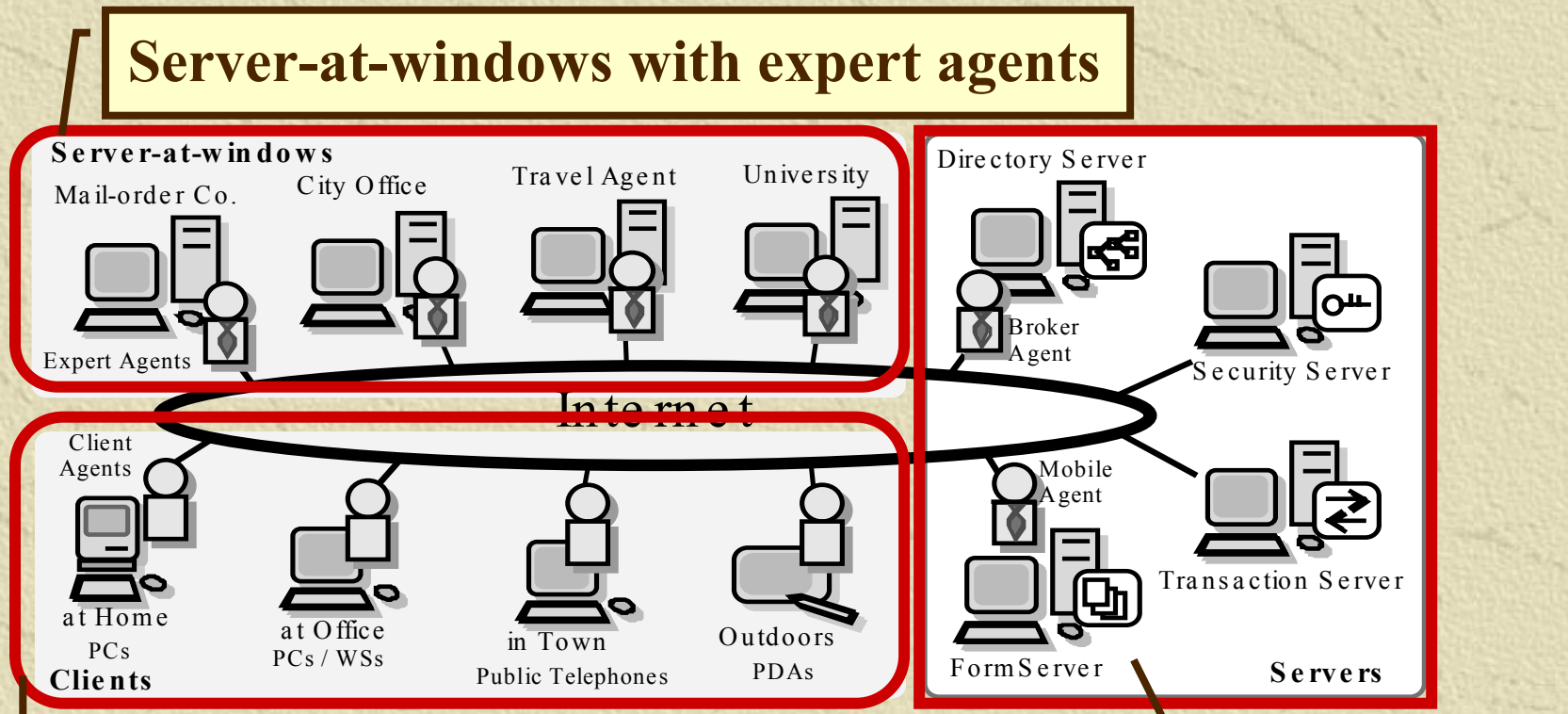
◆ Metaphors

- Window work is considered as service requests between clients and service providers.
- Forms are considered as the interface.
- Our concept : **"One service = One form"**

Application Architecture

- For Agent-based applications -

A Multiagent-Oriented Office Network (MOON)



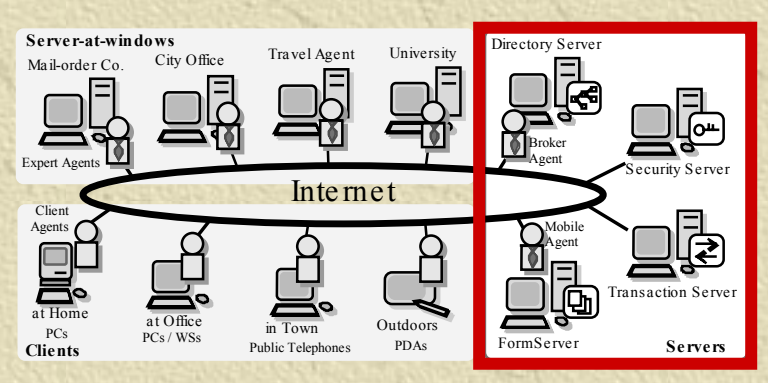
The MOON Servers

(1) **A directory server** with **a broker agent** :
manages service directories of windows.

(2) **A form server** with a mobile agent :
manages forms with help messages.

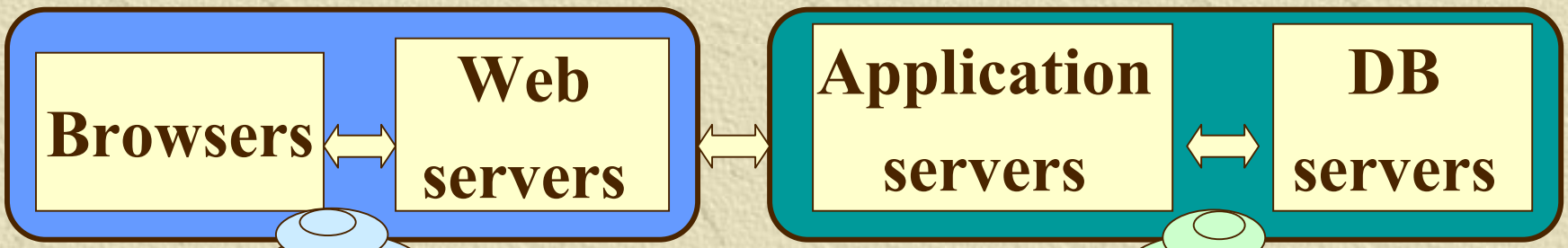
(3) **A transaction server** :
manages written applications
with ID numbers.

(4) **A security server** :
controls access rights.



Our Experiences of Prototyping

- The actual system configuration is the 4-tier architecture

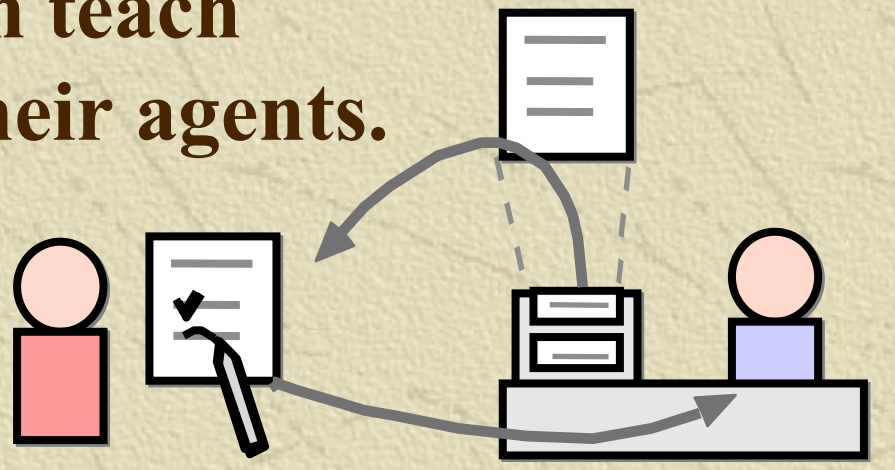


The front end is supported by application framework and multi-agents.

The back end is supported by domain modeling and business objects.

Features of Agent-based Applications as Front End (1)

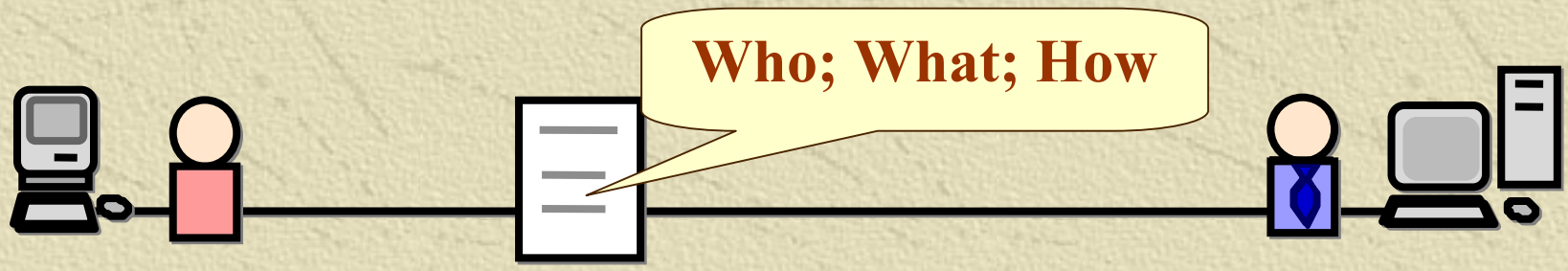
- **Form processing is navigated by agents :**
  - Clients can teach the fixed operations such as their names and addresses to their agents.
  - Domain experts can teach their expertise to their agents.



Features of Agent-based Applications as Front End (2)

- **Standardization of ACL**
  for communication among agents.

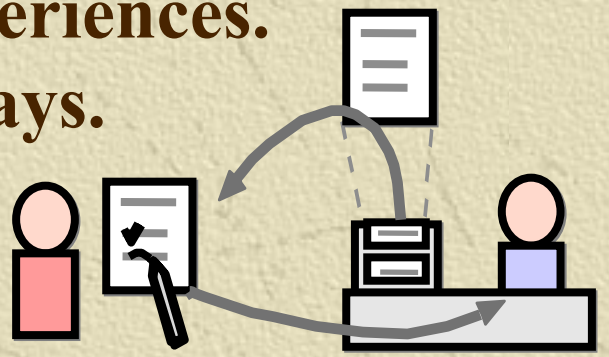
ACL : Agent Communication Language
FACL : Form-based ACL



Automatic Filling in a Form
- Conventional approaches -

◆ Approaches

- Predefined rules for the input fields
  → limitations of the number of rules.
- The auto-complete feature by showing the candidates based on past experiences.
  → Sometimes useful but not always.



◆ Common method

- The value of the name attribute in the input field of the HTML document, is checked.
  → This value is not always believable.

Automatic Filling in a Form
- Basic Problems -

◆ **Using knowledge on the owner itself**

- Ex. a name, an address, a phone number, etc.
- This is independent of each form.

◆ **Solution for different expressions of the same meaning**

- Ex. "Phone" and "TEL"
- Concept names are introduced.
  Ex. @name, @address, @phone, etc.

Cultural Problems of Japanese (1)

◆ **Many different expressions of the same meaning, which are used as label names for input fields.**

Ex. As a part of <u>different expressions</u> for the name, twelve examples are shown in this figure.

氏名	名前	申込者
ご氏名	お名前	申請者
御氏名	お名前（全角）	担当者名
本名	お名前（もしくは法人名）	ネーム

Cultural Problems of Japanese (2)

- **Many types of input data.**
  - Chinese characters
  - Japanese cursive syllabary(hiragana)
  - The square form of hiragana(katakana)
  - English letters

名前
なまえ
ナマエ
NAME

- **Two kinds of character codes**

	8 bits code	16 bits code
Katakana	ﾅﾏｴ	ナマエ
English letters	NAME	ＮＡＭＥ
Arabic numerals	12345	１２３４５

EX. Form for IPSJ Membership

http://www.ipsj.or.jp/mousikomi/m-nyukai.html

正会員･学生会員･準会員入会申込書

氏名（漢　　字）：

例）情報　花子（姓名の間に全角スペース）

氏名（カ　　ナ）：

例）ジョウホウ　ハナコ（姓名の間に全角スペース）

氏名（ローマ字）：

例）JOHO HANAKO（姓名の間に半角スペース

生年月日(yyyy/mm/dd)：

発信元 E-mail：

Kinds of rules

◆ **Target : HTML, not XML**

- **HTML documents have a critical defect of lack of semantic information, but are used mainly.**

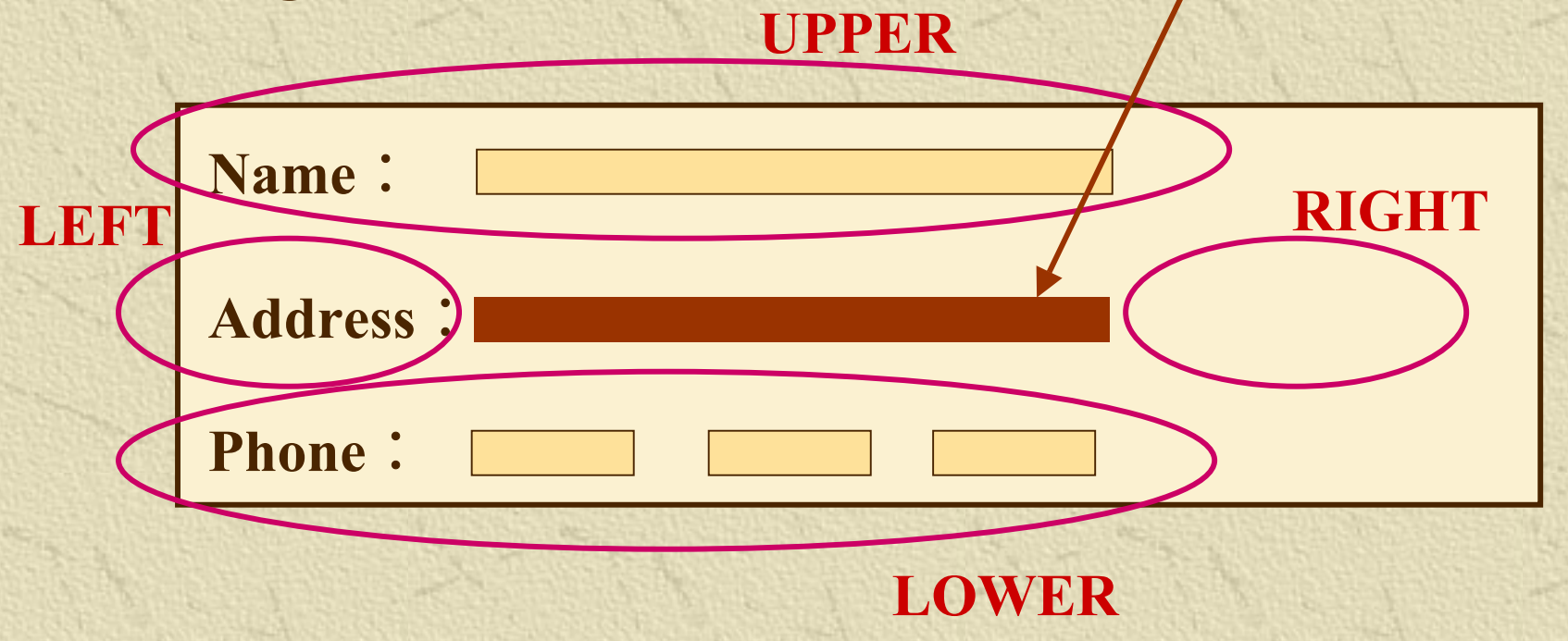
◆ **Two kinds of rules for automatic filling in HTML**

- **Cognitive rules**
  based on cognitive information of displayed forms
- **Experiential rules**
  based on experiences of other users' past behavior

Cognitive Information

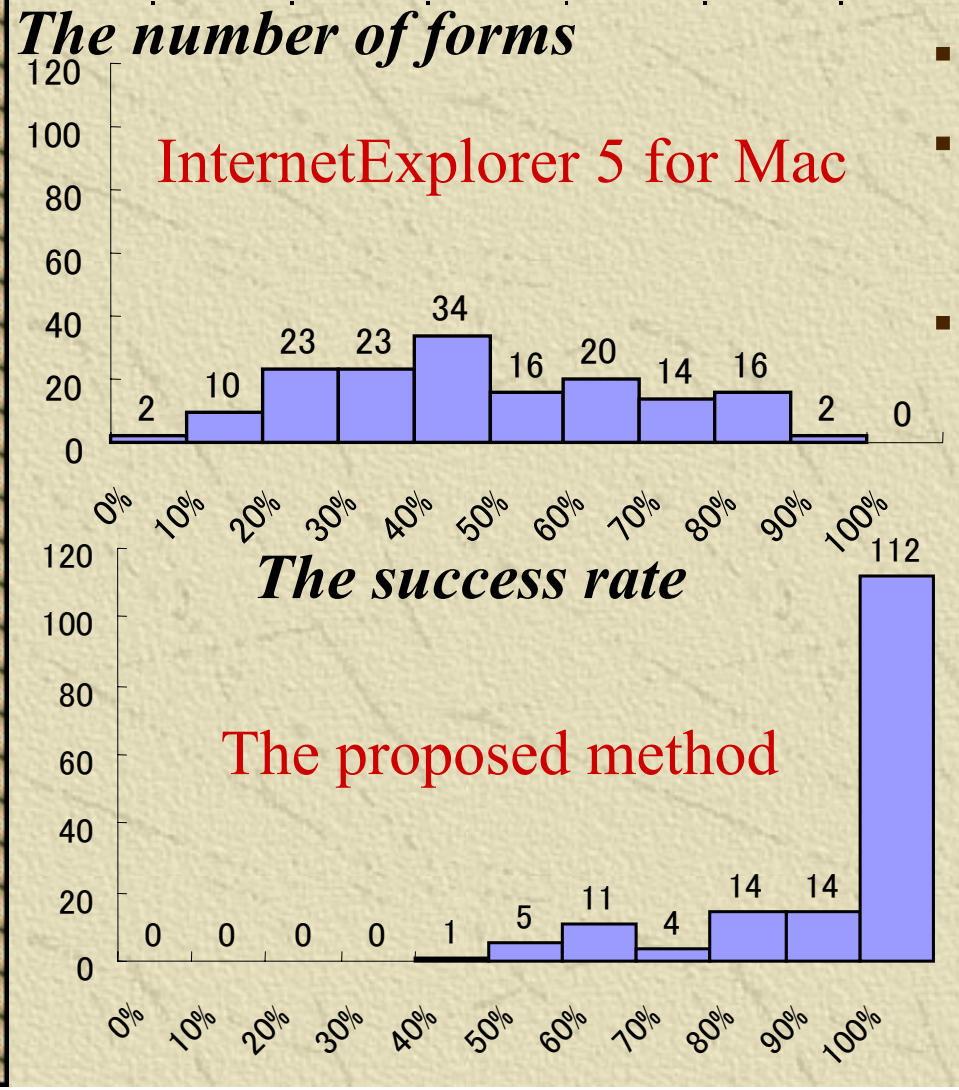
- **Assumption**

  It must be effective to use information about the four sides of the target input field for cognitive rules.



An Experiment

- for confirming this assumption -



- 160 forms to be tested
- 1,914 input fields for personal information
- The average success rate

31%

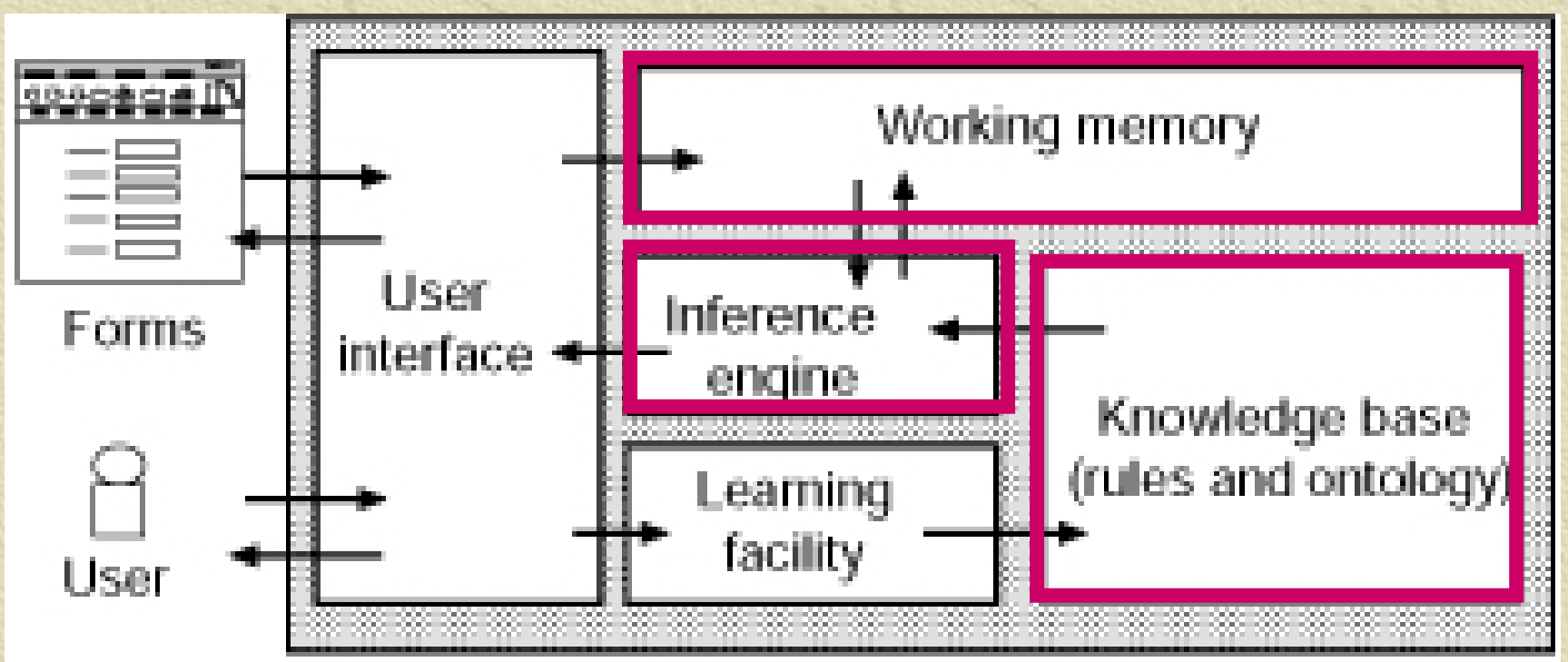
vs.

87%

↓

It is effective to use the four sides of the target input field.

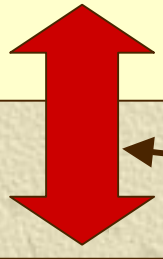
System Architecture
- A Rule-base System -



Knowledge Representation(1)

- The primitive case rule : IF #case THEN #action
- An example

```
IF (UPPER: ("Address"  TEXTFIELD),
      LEFT: "Phone", RIGHT: NONE,
      LOWER: ("Email" TEXTFIELD) )
  THEN @phone
```



Matching

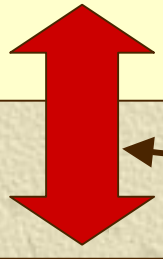
- The working memory

Attributes	*Values*
UPPER	"Address" , TEXTFIELD
LEFT	"Phone"
RIGHT	
LOWER	"Email" , TEXTFIELD

Knowledge Representation(2)

- The abstract case rule → Concept names is used.
- An example

```
IF (UPPER: (@address  TEXTFIELD),
      LEFT: @phone, RIGHT: NONE,
      LOWER: (@email TEXTFIELD) )
  THEN @phone
```



Matching

- The working memory

Attributes	*Values*
UPPER	@address , TEXTFIELD
LEFT	@phone
RIGHT	
LOWER	@email , TEXTFIELD

Feasibility Studies

- The extraction of abstract case rules
  - 160 forms mentioned before
  - 293 input fields of the name
  - 240 abstract case rules
    with the action of the @name

↓

- The application for automatic filling
  - 139 forms other than the above 160 forms
  - 239 input fields of the name
  - 63 fields (26%) were successful.
    ** Not enough **

Extension of Reasoning (1)

◆ Complete matching → incomplete matching

- The reasoning of similarity on the case part

(1) The matching is performed for each one of the four attributes of the case part.

(2) The matching with the same action is counted for each attribute and for each action.

(3) The relative frequency of each action for each attribute is calculated.

$$O_{ij} = \frac{n_{ij}}{\sum_{k=1}^{M} n_{ik}}$$

The attribute i {i=1,2,3,4}

The action j {j=1,2,…,M}

Extension of Reasoning (2)

(4) The average of the relative frequencies for the four attributes is calculated as a certainty factor for the action.

(5) The action with the maximum is selected.

$$CF_j = \frac{\sum_{i=1}^{4} O_{ij}}{4}$$

The attribute i {i=1,2,3,4}

The action j {j=1,2,…,M}

- The application for automatic filling
  - 176 fields (74%) were successful.
  - 62 fields of 63 unsuccessful fields were filled in with wrong values.

Analysis of Unsuccessful Fields

◆ **The first experiment : 176 unsuccessful fields**

- **Lack of rules : 168**
- **Lack of keywords in the ontology : 8**

↓

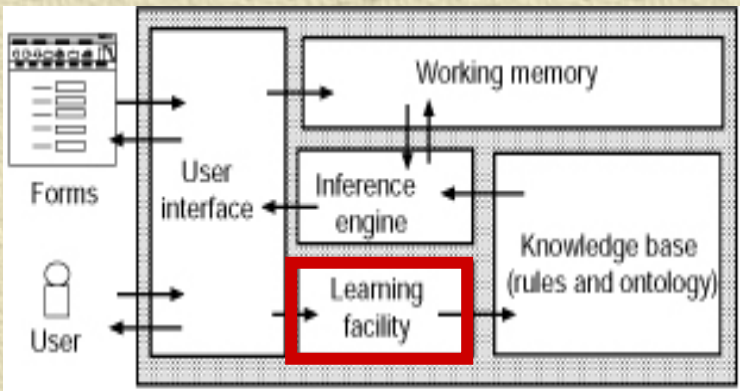
◆ **The second ex. : 63 unsuccessful fields**

- **Lack of rules : 57**
- **Lack of keywords in the ontology : 6**

↓

◆ **The solution :**

Learning of the agent through the learning facility



Learning of the Agent

- **For improving lack of rules**

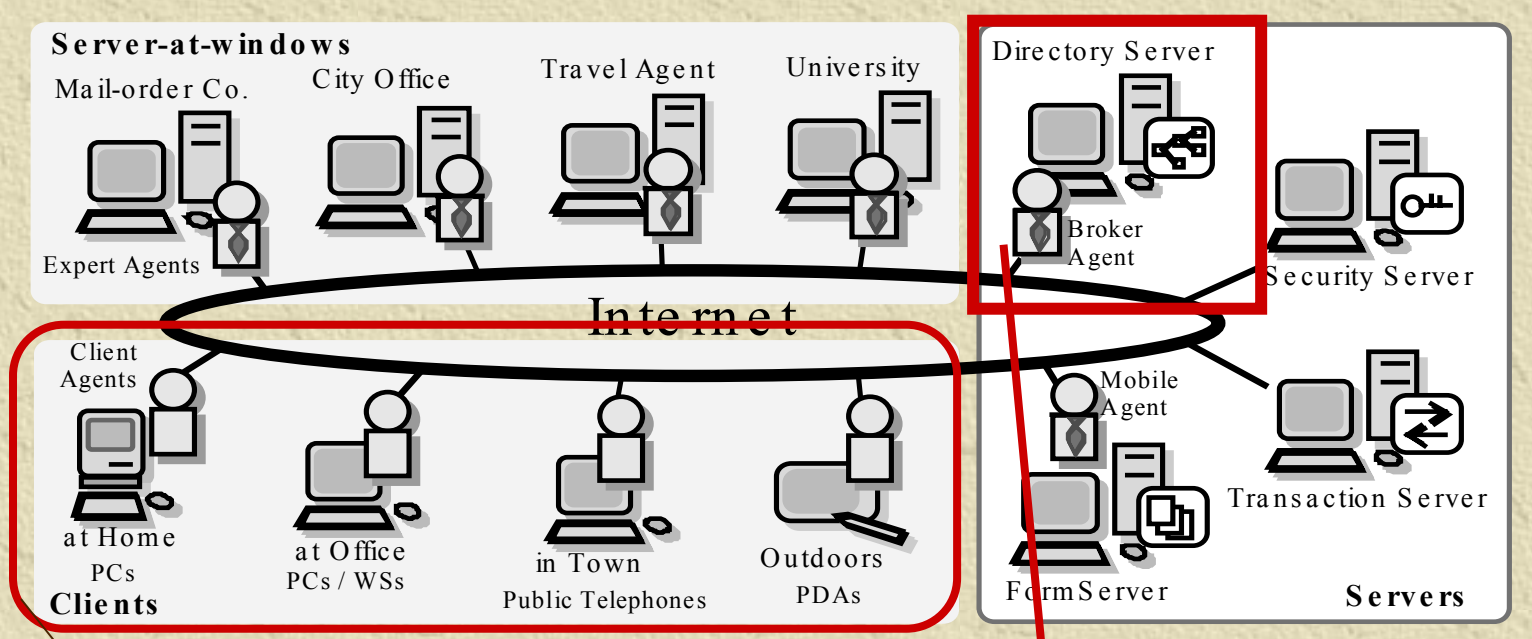
  The agent can acquire new rules by monitoring what the user fills in the target field with, in which the agent could not fill correctly.

- **For improving lack of keywords**

  The agent can acquire new keywords, while it inquires of the user whether the keyword on the left side corresponds to the concept name of the actual value inputted.

Experiential rules

- **Two tasks of a broker agent**
  - **The directory service on forms**
  - **The management of experiential rules**



Automatic Rule Generation

◆The experiential rules are gathered :

(1) A user agent inquires of the broker agent about a necessary form.

(2) The broker agent sends the experiential rules.

(3) The user agent fills in the form automatically.

(4) The user corrects the form if necessary.

(5) The user agent sends the form to the window, and sends the broker agent the information about fields,

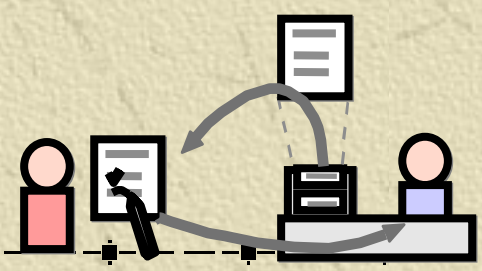
- what values are inputted into the fields modified,
- what values are inputted into the blank fields.

An Example of an Experiential Rule

- XML base -

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=
"http://www.w3.org/1999/02/22-rdf-syntax-ns¥#"
 xmlns:o="http://wwhww.org/schemas/userprofile/1.0/"
 xmlns:="http://wwhww.org/1.0/">
          <FormItem rdf:about="http://www.se.cs.meiji.ac.jp/
library/entry/¥#form[entry].item[name]">
            <history>
              <Profile amount=10>
                <value><o:User.Name.First /></value>
                <separator> </separator>
                <value><o:User.Name.Last /></value>
              </Profile>
            </history>
          </FormItem>
</RDF>
```

Feasibility study



- By the first testee
  - 50 forms with 497 fields for the information :
    a name, an address, a telephone, a fax,
    a birthday and an email address
  - 531 experiential rules were extracted.

↓

- By the second testee
  - 531 fields in automatically (497 are correct)
  - 34 fields were corrected.

↓

- By the third testee
  - 501 fields in automatically (497 are correct)
  - 4 fields were corrected.

Conclusions

- The multi-agent framework was proposed for enduser-initiative application development of web applications.
- The user agents and the broker agent for automatic filling in a form were developed as the front end system of a web application.
- Feasibility studies confirmed the effectiveness of both the abstract case rules of cognitive rules and experiential rules of other users.