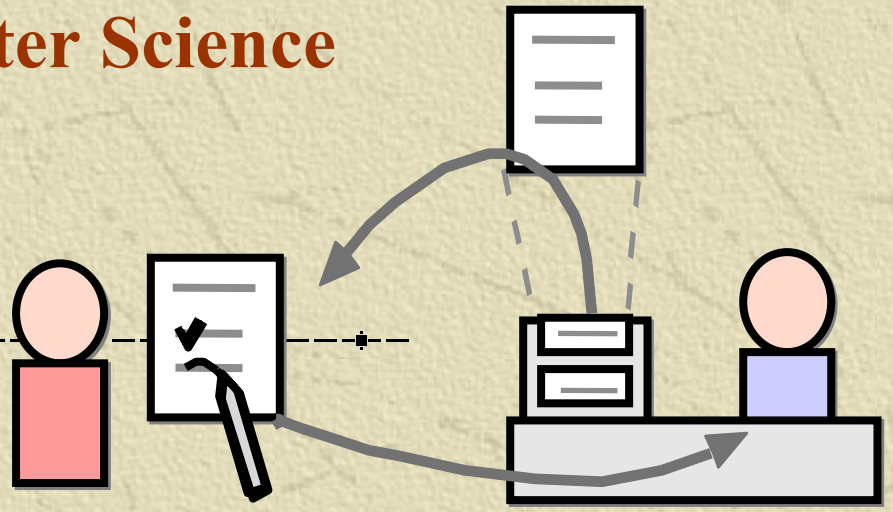
A Form-based Approach for Web Services by Enduser-Initiative Application Development

Takeshi CHUSHO, Katsuya FUJIWARA,
Hisashi ISHIGURE and Kei SHIMADA
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**
  - **Primary target domain : local areas**
    C2D (Client to Department)
    D2D (Department to Department)
  - **Applied technologies:**
    * **Component-Based Software Engineering**
    * **Multi-agent systems**

Basic Concepts for Web Services

◆ Rapid development and continuous variation :

・Business on Internet, is rapidly changed,

・The period from defining a new business model through releasing new services, must be short.

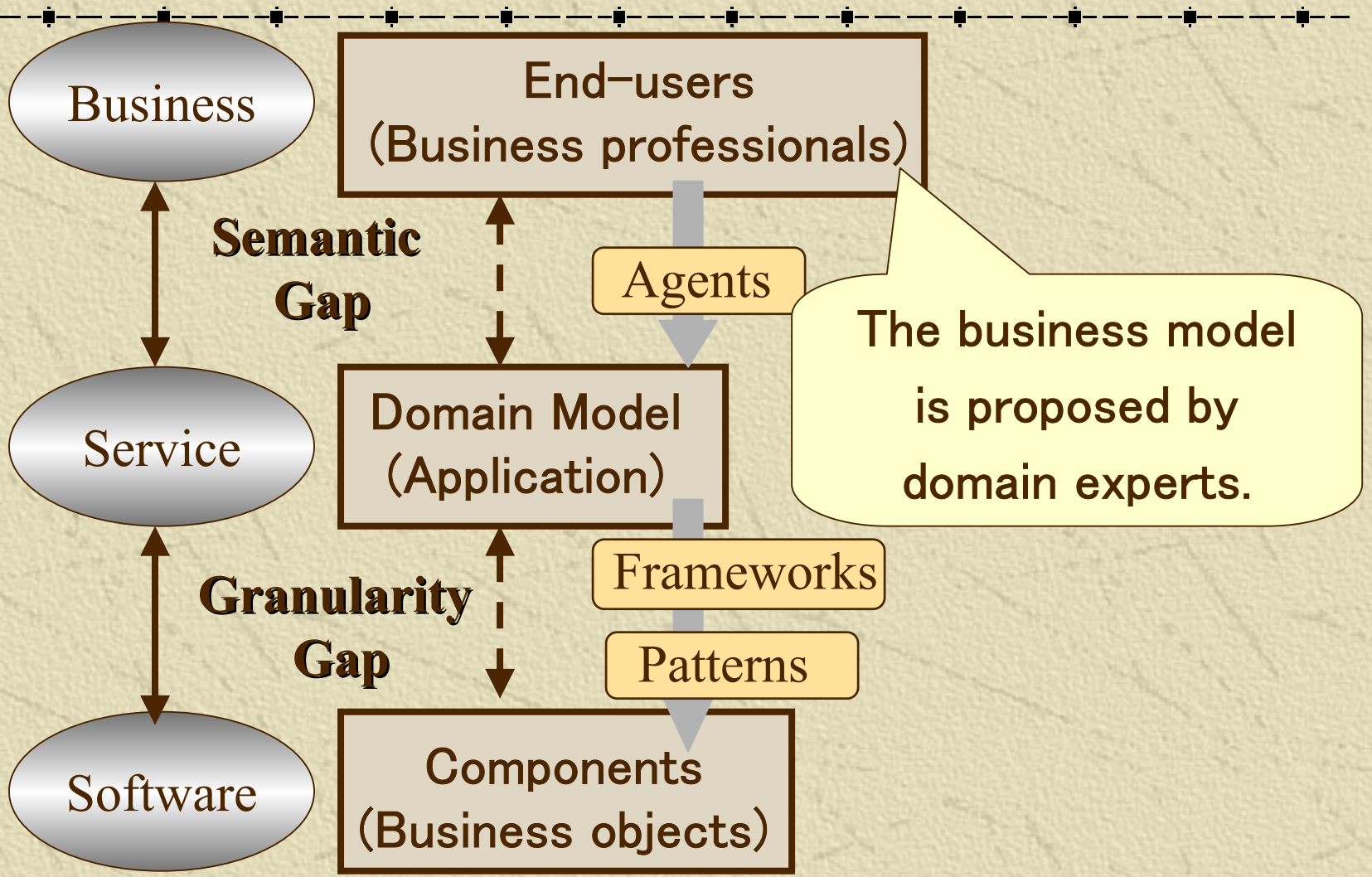
◆ End-user initiative development :

・For “just-in-use”

・Development by system engineers is not suitable.

Enduser-Initiative Approach

- how to make web applications supporting web services -



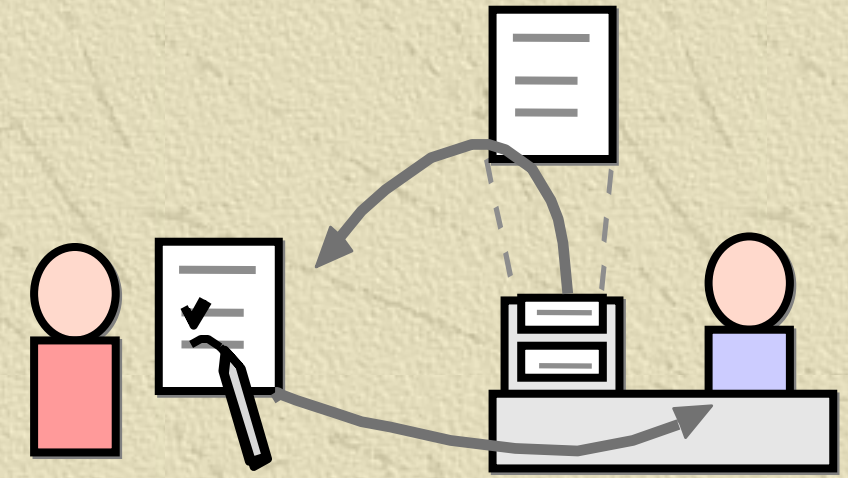
Metaphors for Web Services
- Window work -

◆ 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.

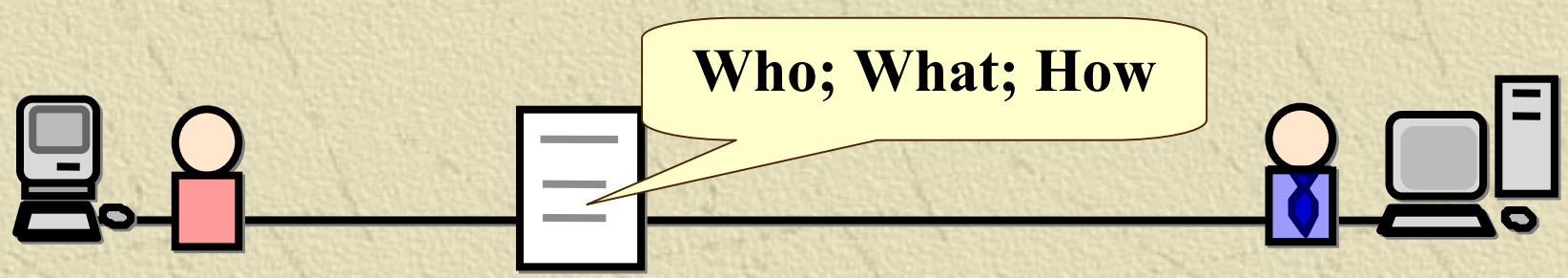
◆ Problems

- Conventional systems
  by IT professionals
  are expensive.
- It is difficult
  to modify software timely.



Metaphors for Web Services Protocol - Forms -

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

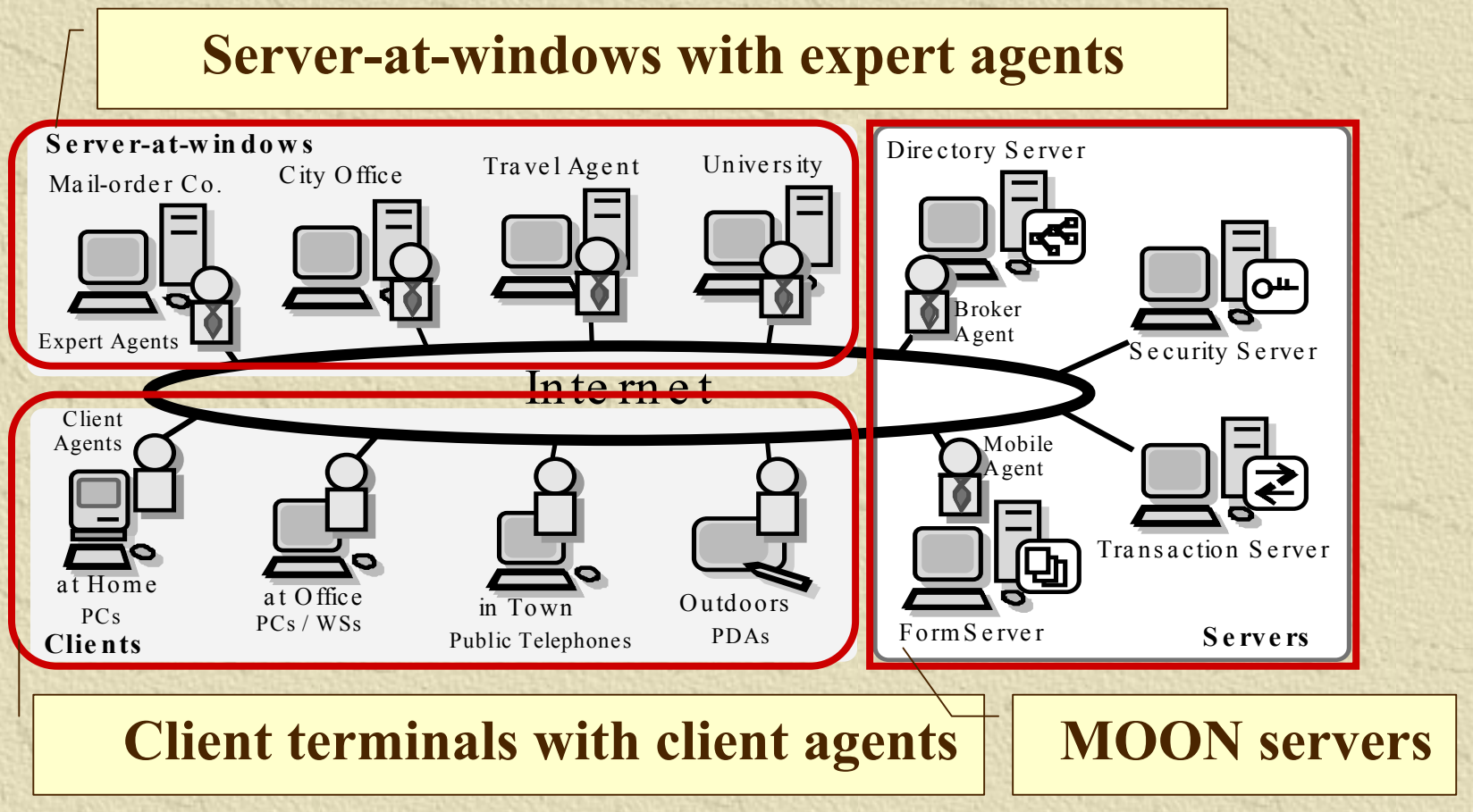


Our actual XML-base protocol of { who, what, how } corresponds to
the UDDI protocol of { white, yellow, green } pages.

Application Architecture

- For Agent-based applications -

A multiagent-oriented office network (MOON)



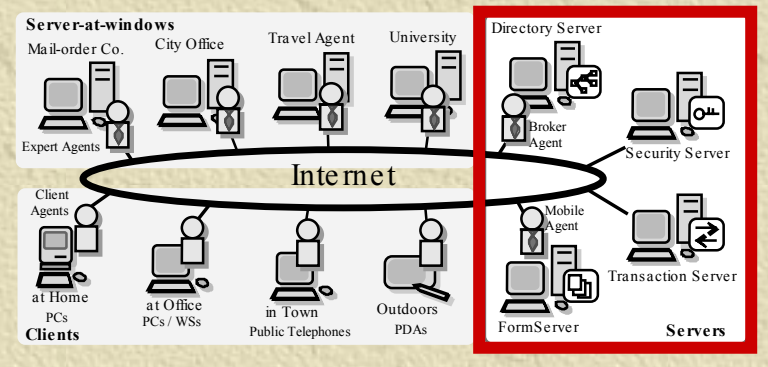
The MOON Servers

(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.

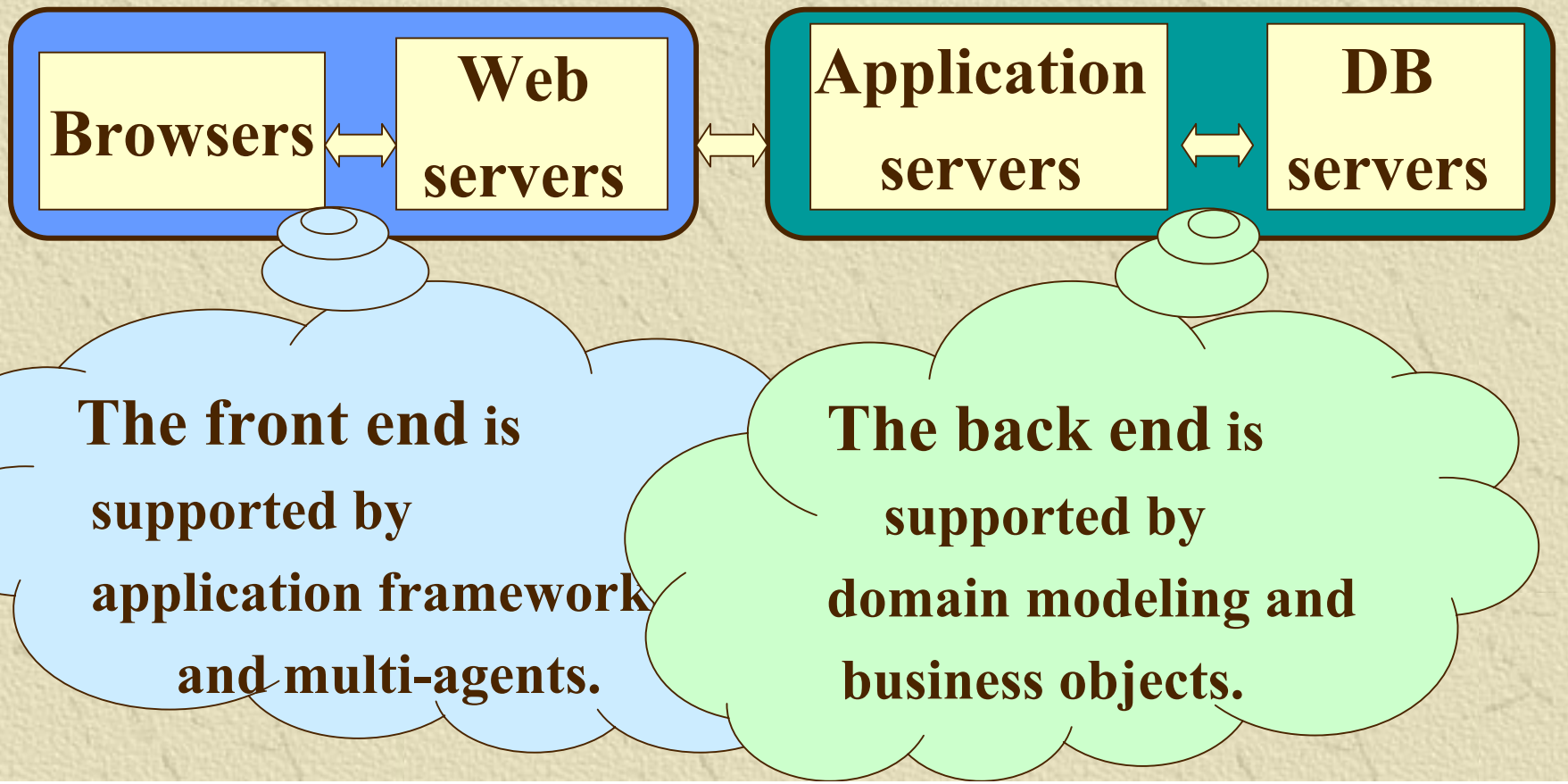
(3) A transaction server :
manages written applications
with ID numbers.

(4) A security server :
controls access rights.



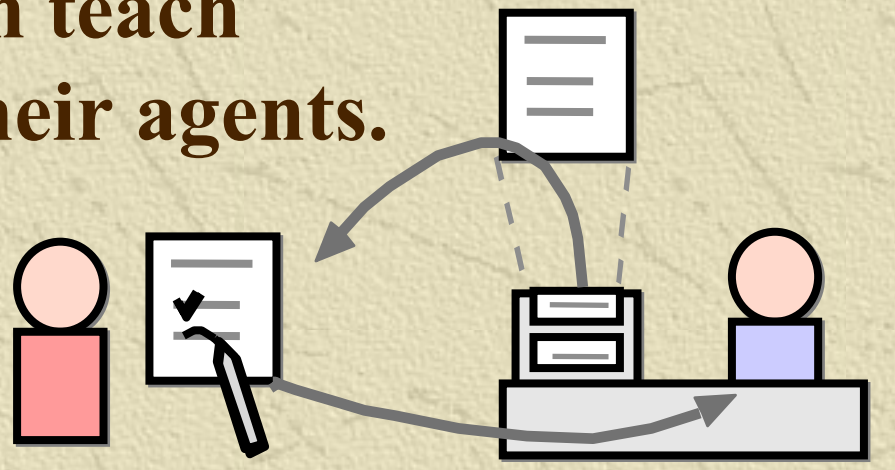
Our Experiences of Prototyping

- The actual system configuration is the 4-tier architecture



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

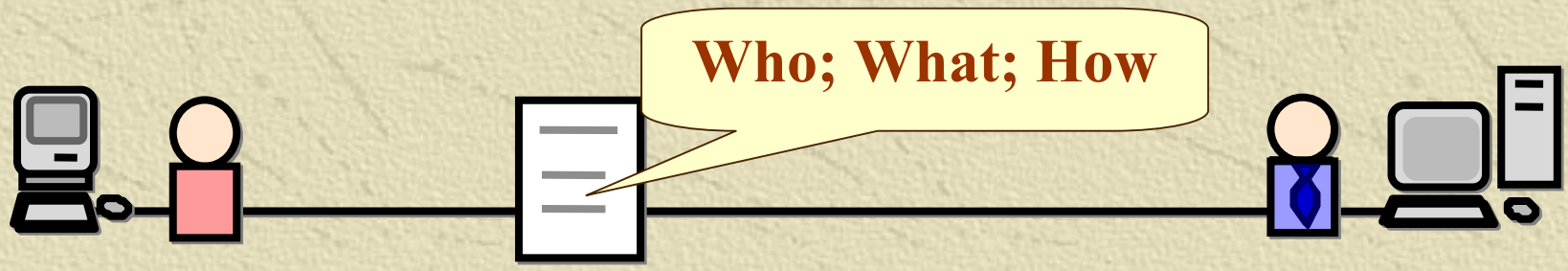
- **Form processing is navigated by agents :**
  - Clients can teach the fixed operations such as their names and addresses to their agents.
  - Domain experts can teach their expertise to their agents.



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

- **Standardization of ACL**
  for communication between client agents and expert agents.

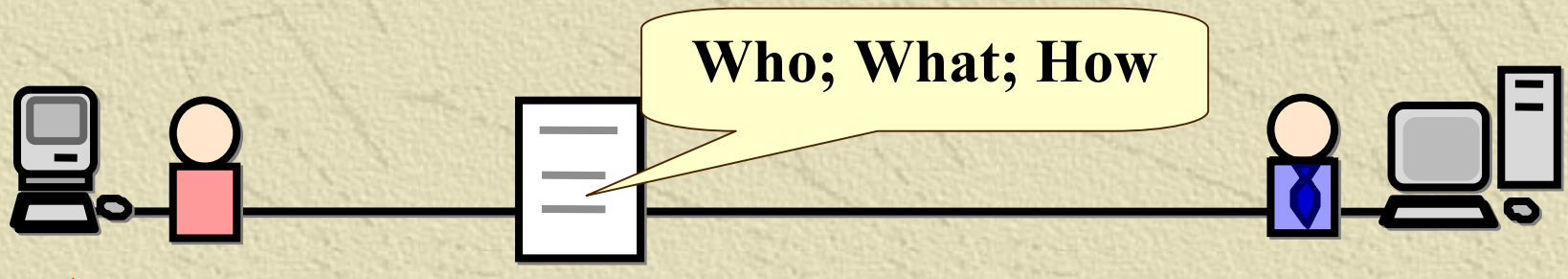
ACL : Agent Communication Language
FACL : Form-based ACL



The Basic Form of Web Service Interface

- **Messages of requests to windows :**
  - Who receives your request ?
  - What do you request to the window ?
  - How do you request it ?
  - Which is your request ?

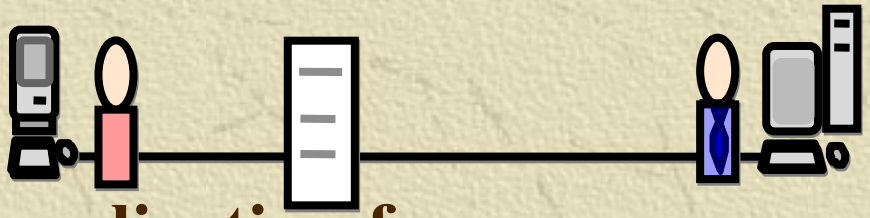
<Note> wwHww : who-what-how with WWW



- **The basic form : (who, what, how, which)**

The Semantics of the Protocol

- **Message passing of OOP**
  - **who : A message receiver object**
  - **what : A method name**
  - **how : Parameters for a method invocation**
  - **which : A message number**
- **FACL**



  - **who : A window**
  - **what : Title of the application form.**
  - **how : Contents of the application form**
  - **which : A receipt number**
- **Difference : In FACL, an unknown value of a parameter implies an inquiry about the parameter.**

End-User Interface (1)

The basic form notation

a, b, ... : Parameters with known values.

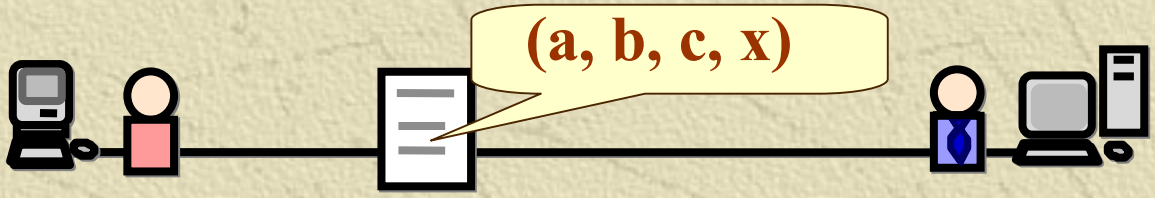
?a, ?b, ... : Inquiries about the parameters, which request help messages.

x, y, ... : Parameters without values.

?x, ?y, ... : Inquiries about the parameters, which request possibles for selection.

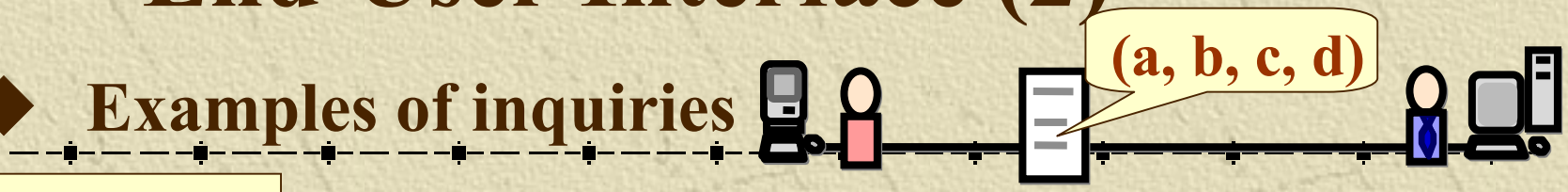
Examples

(a, b, c, x)



The application, b, with the contents, c, is sent to the window, a. A message number will be assigned to the variable, x, by the window.

End-User Interface (2)



- Examples of inquiries

(a, b, , ?d) : The state is inquired.

(a, b, ?x,) : The application form, b, is displayed. How to fill in the form is navigated by the expert agent. Some typical items are filled automatically by the client agent.

(a, ?x, ,) : The title list of all application forms which the window, a, receives, is displayed.

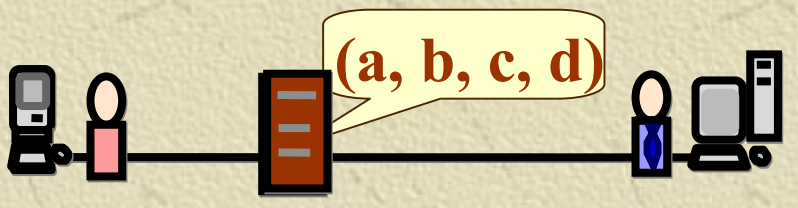
(?x, ?y = (a list of keywords), ,) :
The list of titles of all application forms which relate to the list of keywords, is displayed.

End-User Interface (3)

- Examples of inquiries

(?a, , ,) : The explanation on the work of the window, a, is displayed.

(a, ?b, ,) : The explanation on the application form, b, to be sent to the window, a, is displayed.



- Comparison with UDDI

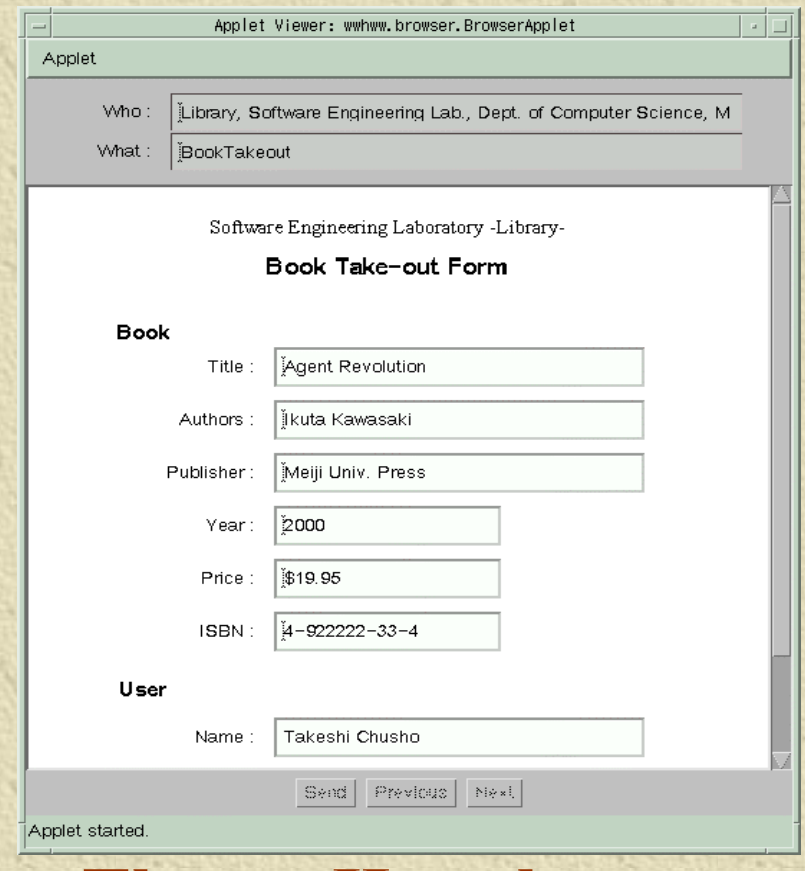
These inquiries must be simpler for end-users.

<FACL>	:	<UDDI>
who	:	a white page
what	:	a yellow page
how	:	a green page

The First Application

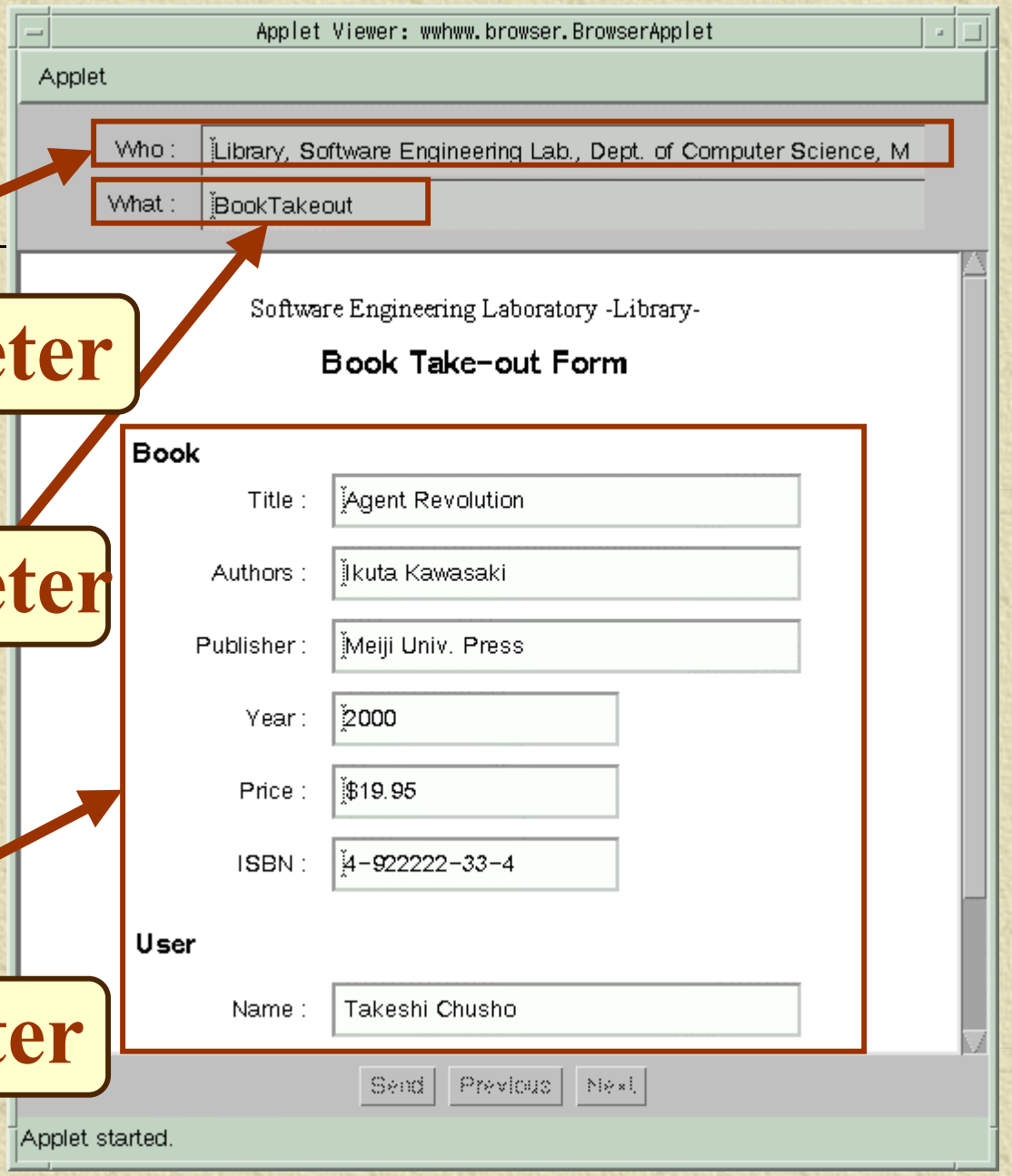
- A library system in our laboratory -

- There are no librarians.
- It is easy to know who borrowed a book because everyone fills in a form when taking out.
- It is easy to know whether a book has been already registered or not because everyone fills in an form after he or she bought the book.



The wwHww browser at a client terminal

The wwHww Browser



The who-parameter

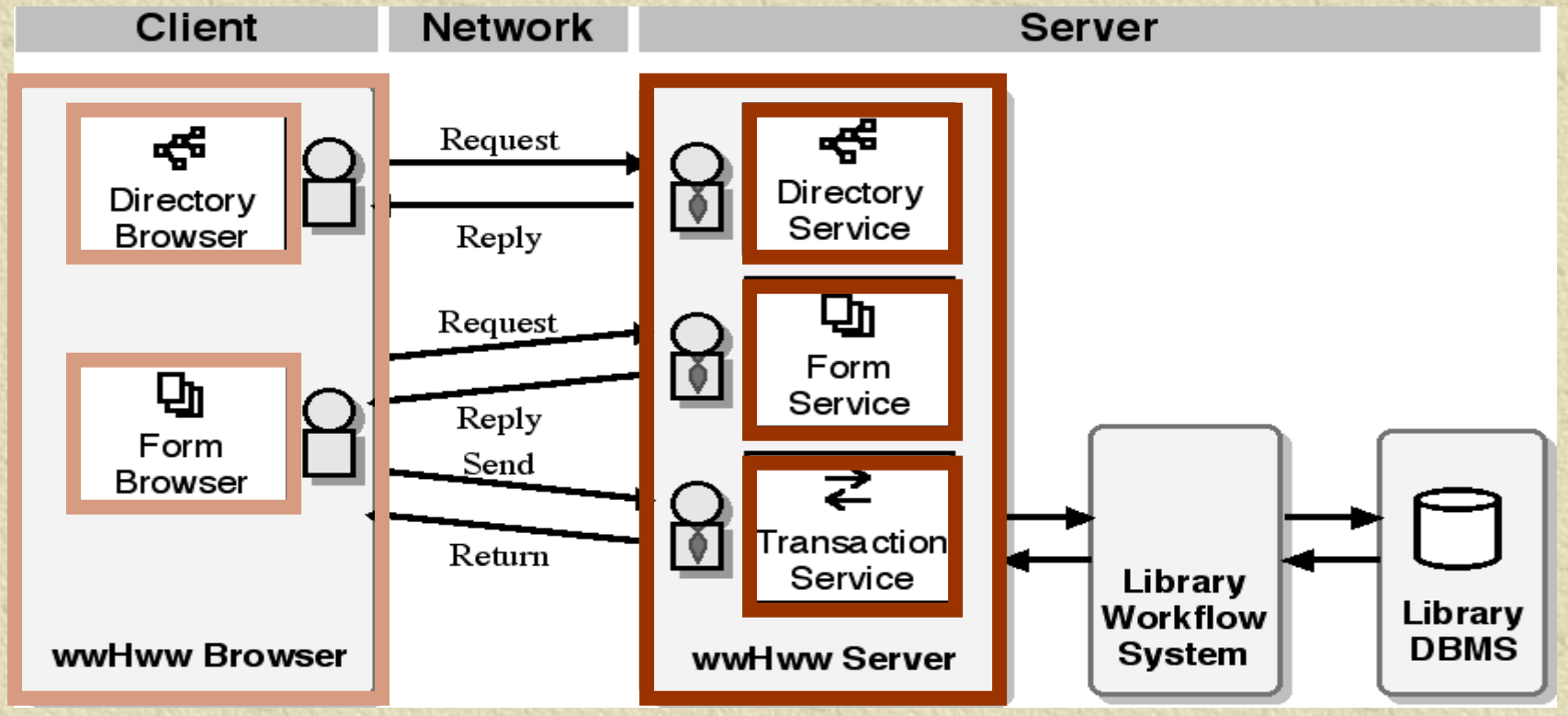
The what-parameter

The how-parameter

The Software Architecture

◆ Application framework

- A client / server model
- The wwHww browser has two subsystems
- The wwHww server has three subsystems



An Application Building Procedure

(1) Service definitions :

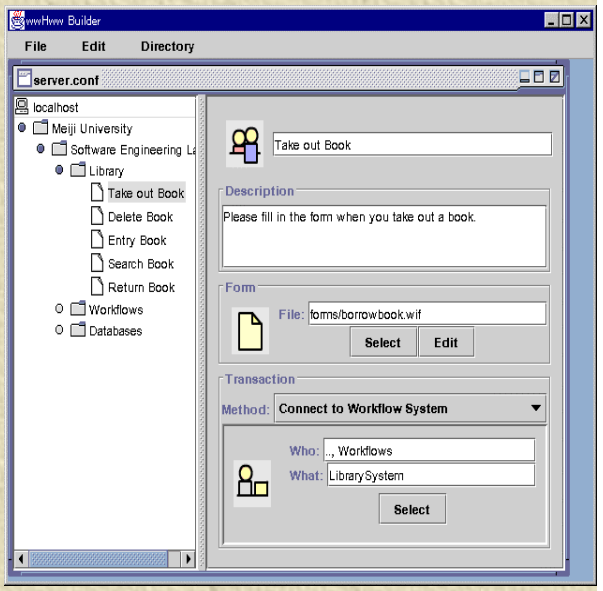
Services at the window, are defined.

(2) Form definitions :

Electronic forms for these services are defined while embedding navigation information.

(3) Registration :

These definitions are registered into the corresponding servers.

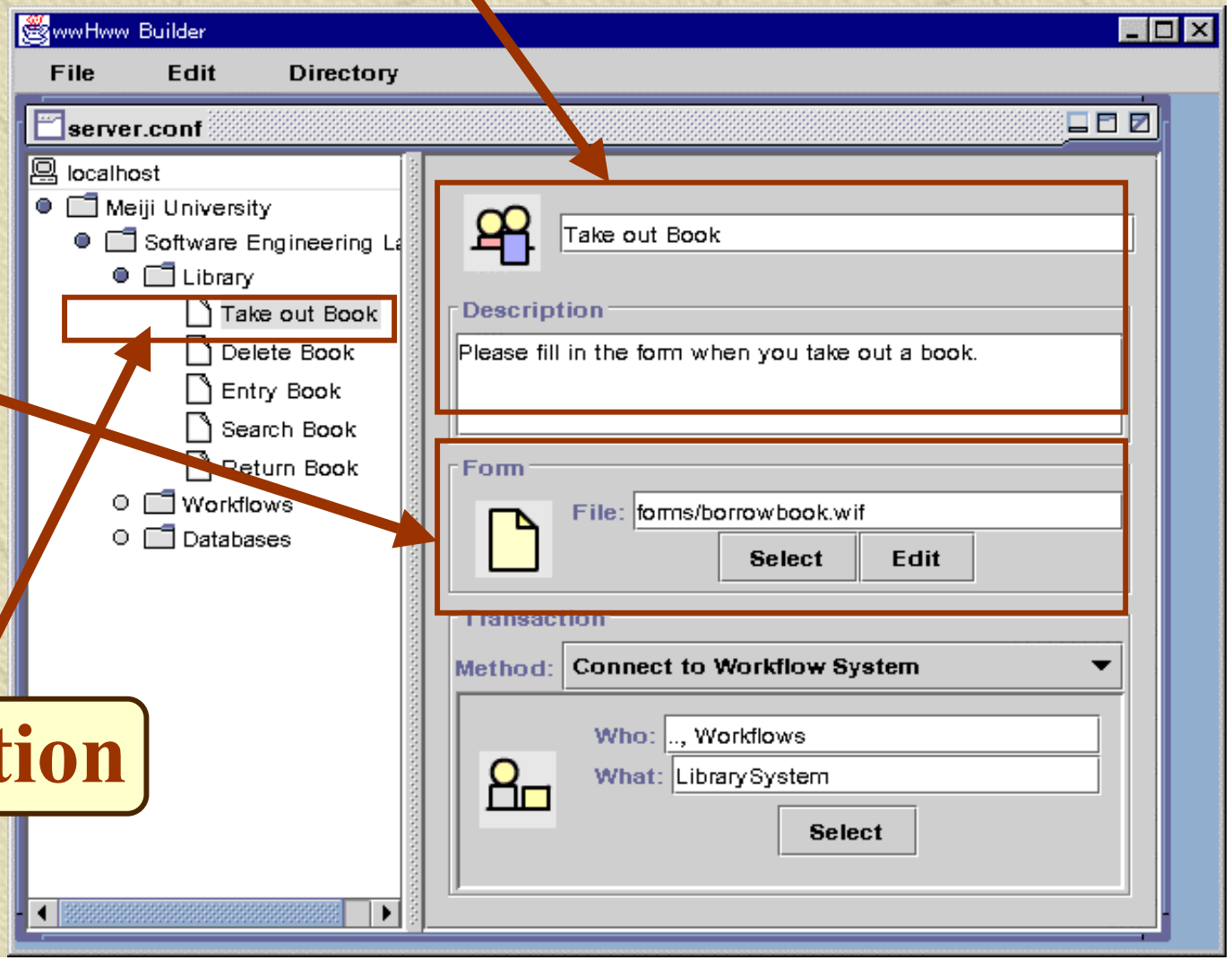


The Browser for System Definitions

(1) Service definitions

(2) Form definitions

(3) Registration



Examples of XML-base Navigation

◆ Intelligent navigation by agents

- The meta data for a window is described in an RDF style.
  < RDF : Resource Description Framework >
- While forms are defined in HTML in the conventional way, the semantics of forms are defined in an RDF style.
- FACL messages are described in XML.

Meta Data Definition for a Window
- An example of a library (part) -

```
<!DOCTYPE RDF [  <!ENTITY site 'http://se.cs.meiji.ac.jp'>] >
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-....."
         xmlns:d="http://purl.org/dc/elements/1.1/"
         xmlns:w="http://wwhww.org/1.0/">
  <w:Agent rdf:about="&site;/ library /">
    <d:Title>Library</d:Title>
    <d:Description>The library system of SElab.</d:Description>
    <d:Text>Lib. of lab.&lt;br&gt;Book take-out service</d:Text>
    <w:name>/Meiji-U/CS/SE/Library</w:name>
    <w:service rdf:resource="&site;/library/takeout"/>
    <w:service rdf:resource="&site;/library/return"/>   • • •
  </w:Agent>
  <w:Form rdf:about="&site;/library/takeout">  • • •  </w:Form>
</rdf:RDF>
```

The Semantic Definitions of Forms

- An example of form definitions (part) -

```
<w:Form about="http://se.cs.meiji.ac.jp/library/takeout/">
 <d:Title>take-out</d:Title>
 <d:Description>A procedure for book take-out</d:Description>
 <w:input>

  <w:FormItem>
   <w:name>usr</w:name>
   <w:datatype resource="http://imc.org/vCard/3.0#FN"/>
   <w:value resource="urn:userprofile:@user.name.fullname"/>
   <w:help resource=" http://inside.se.cs.meiji.ac.jp/
                              library/takeout/help.html#usr "/>
  </w:FormItem>

 </w:input>
</w:Form>
```

An Example of a Request Message
- (a, ?x, ,) for a list of services -

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
                                    syntax-ns#"
 xmlns:w="http://wwhww.org/1.0/">

 <w:Message rdf:about="">
  <w:who>/Meiji-U/CS/SE/Library</w:who>
  <w:what><w:query/></w:what>
 </w:Message>

</rdf:RDF>
```

Application Development Environment, M-base, for Back End

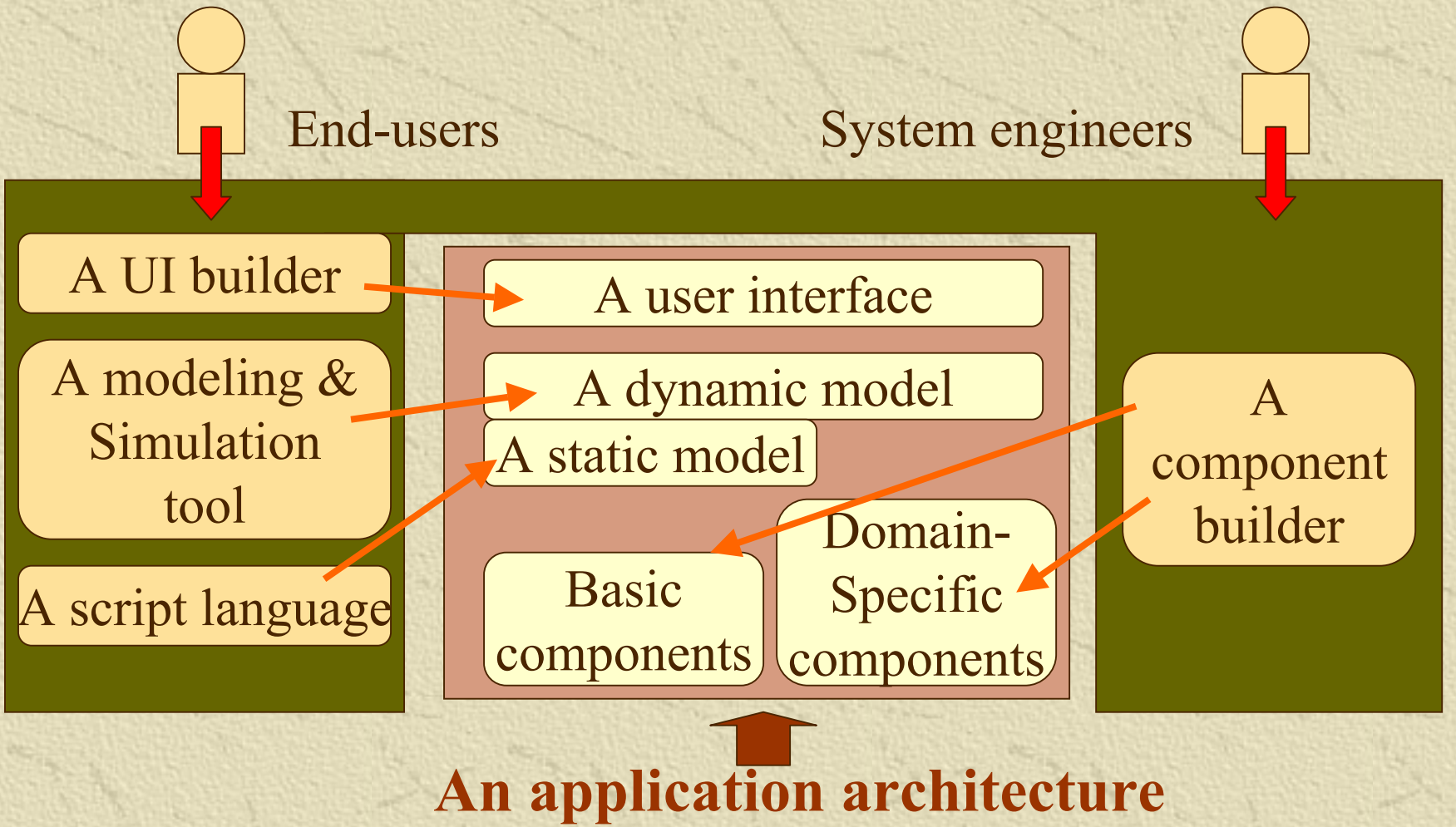
- **Modeling process**
  - The business logic by end-users of business professionals or domain experts.
  - One solution for enduser-initiative development :

“ A domain model ≡ A computation model ”

  - One task in a domain model corresponds to one object in an object-oriented model.

“ Analysis ≡ Design ”

Framework of M-base



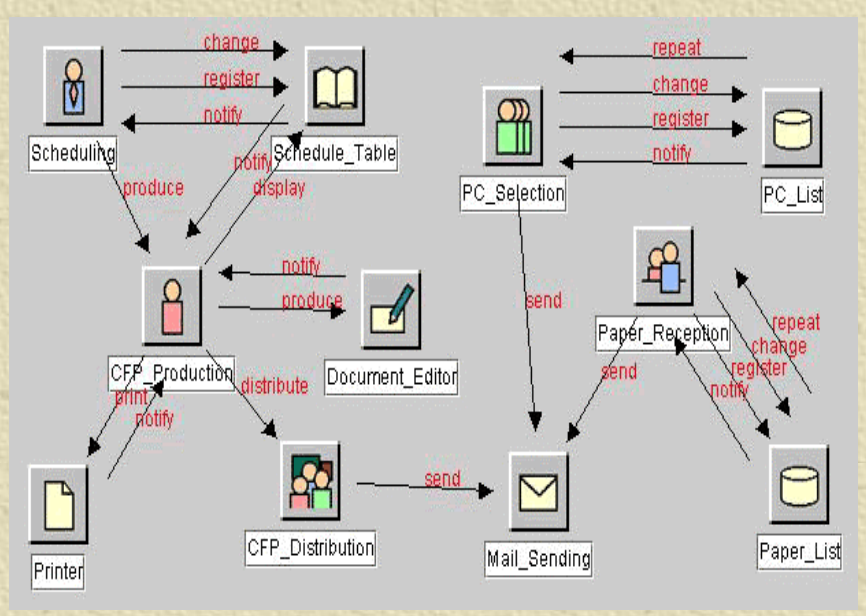
Modeling Procedure

◆ **The typical procedure :**

(1) Definitions of external specifications
(2) Construction of a domain model
(3) Refinement of user interfaces
(4) Simulation of behavior

◆ **An example :**

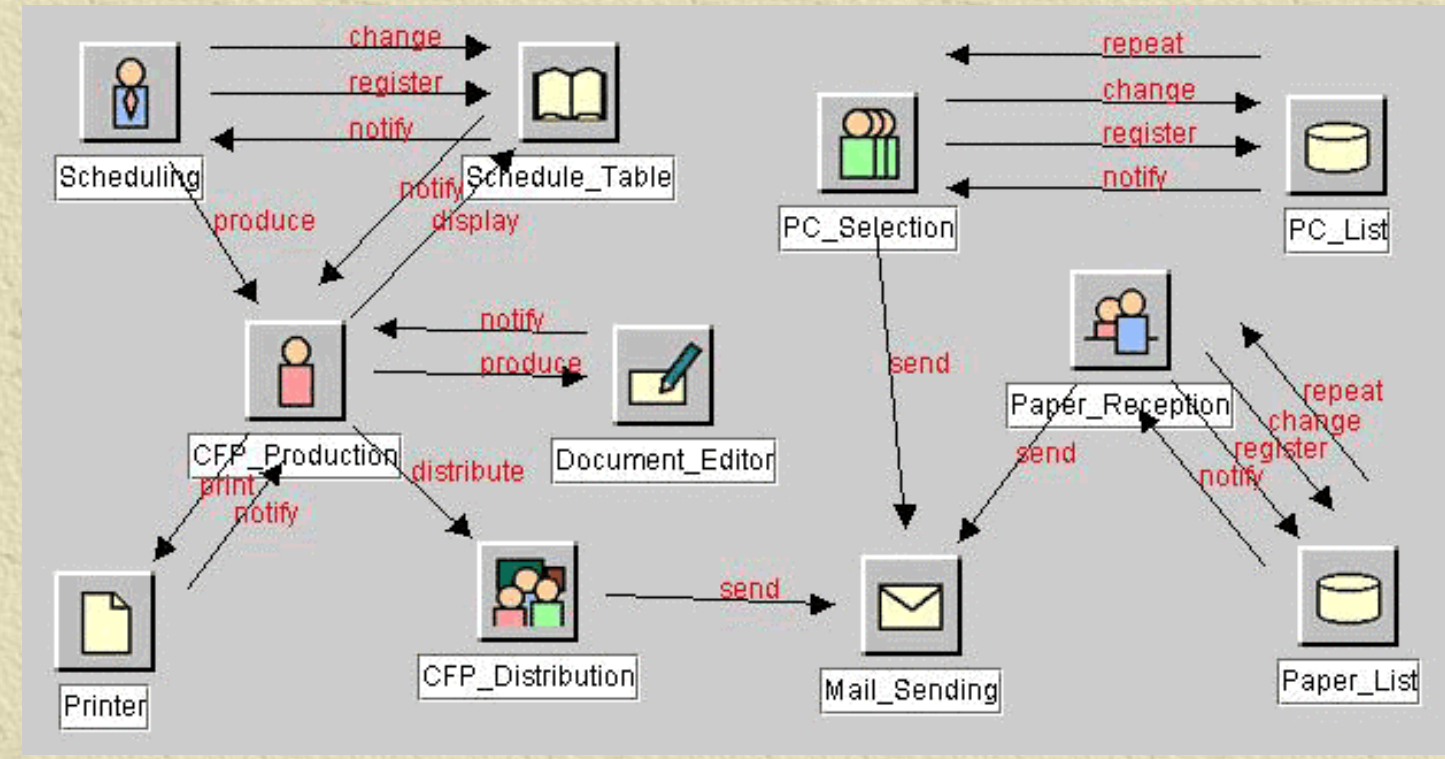
“ Tasks of a program chair for an international conference ”



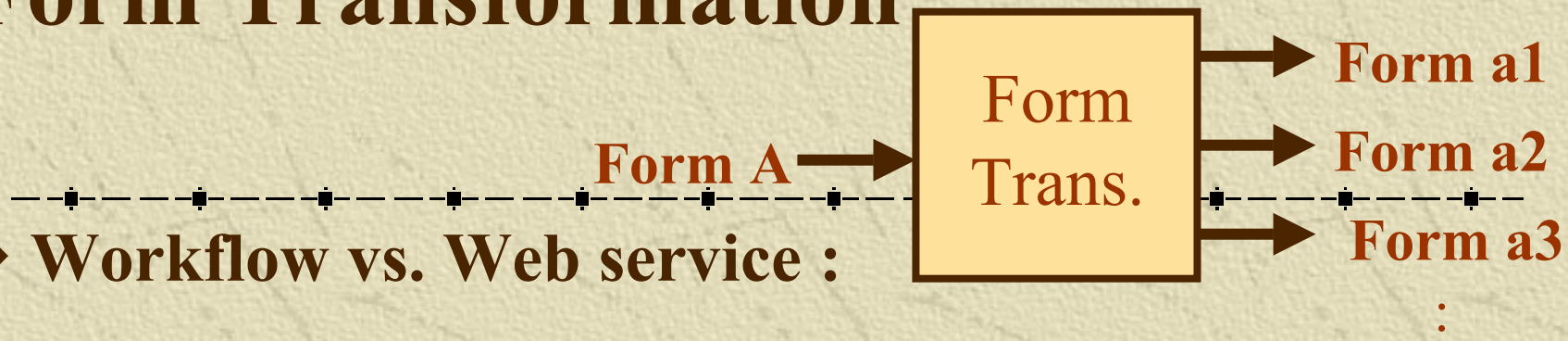
The Domain Model

◆ By using the modeling tool :

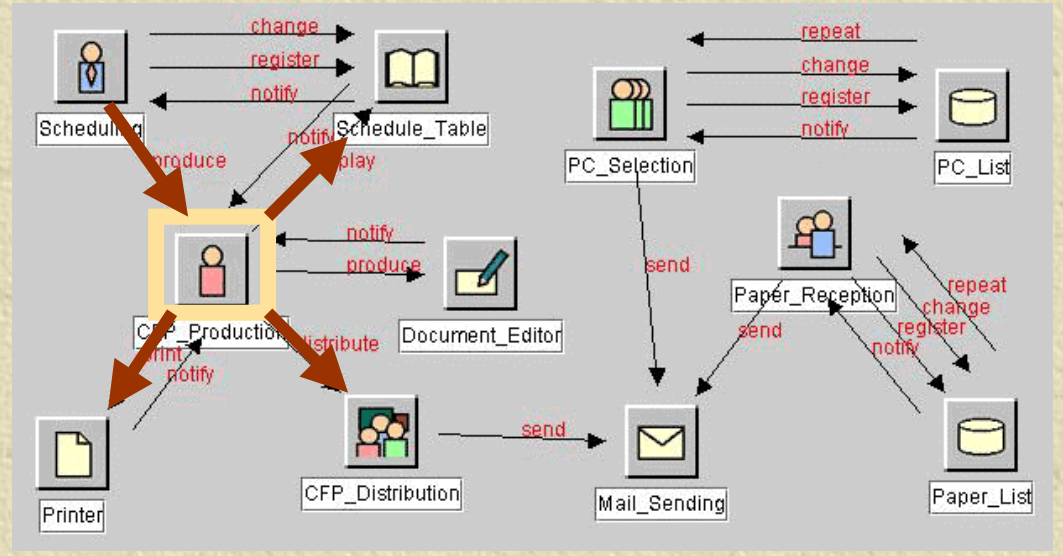
- Objects are defined by drag-and-drop from the palette of icons.
- A message is defined by drawing an arrow line.



Form Transformation



- **Workflow vs. Web service :**
  - **Business objects → web services**
  - **The message flow → the form flow**
  - **Message transformation → form transformation**
  - **The form flow = web service integration :**



Conclusions

- The Form-based approach for web services by enduser-initiative application development was proposed.
- The front end of the system is supported by application frameworks and multi-agents.
- The back end is supported by domain modeling and business objects.