End-User-Initiative Development with Domain–Specific Frameworks and Visual Modeling

Takeshi CHUSHO[a,1], Feng ZHOU[a,1] and Noriyuki YAGI[a,1]
[a] *Department of Computer Science, Meiji University, Kawasaki, Japan*

Abstract. The development of Web applications should be supported by business professionals themselves since Web applications must be modified frequently based on their needs. This paper describes the end-user-initiative development of Web applications based on two approaches. The first approach uses domain-specific frameworks which are effective for UI-driven systems. The second approach uses visual modeling tools which are useful for model-driven systems such as workflow systems. Both approaches suppose the three-tier architecture of user interface, business logic and database. After these approaches were applied to the development of some applications, the composite approach was considered for a typical application which should be easily developed and frequently modified by end-users.

Keywords. End-user computing, domain-specific framework, visual modeling

Introduction

The number of Web applications which end-users have access to has been increasing. Most of these applications are developed by IT professionals. Thus, attempting to achieve automation is limited to particular tasks which calculate profit over the development cost. Furthermore, it is difficult to develop applications quickly. Primarily, Web applications should be supported by business professionals themselves since Web applications must be modified frequently based on users' needs. Therefore, end-user-initiative development has become important for the automation of end-users' fulfilling their own needs.

There are several approaches for end-user-initiative development. The UI-driven approach makes it possible to develop applications for the UI-centered front-end systems easily. It is strengthened by using domain-specific framework technologies. The model-driven approach makes it possible to develop applications for workflow-centered back-end systems easily. It is strengthened by using a visual modeling tool.

Terms for end-user computing (EUC) and papers on EUC often came out in the 1980's. Some papers described definitions and classifications of EUC [6] or the management of EUC [3]. A recent paper summarized the trends of end-user

[1] Takeshi CHUSHO, Feng ZHOU and Noriyuki YAGI are with the Department of Computer Science, School of Science and Technology, Meiji University, Kawasaki, 214-0033, Japan.

This work was supported in part by KAKENHI of Grant-in-Aid for Scientific Research.

development without IT professionals' assistance [15]. End-user software engineering research for end-user programmers and domain experts appeared also[9, 11].

There are some other works related to EUC. In the programming field, the technologies for programming by example (PBE) [12] were studied. PBE implies that some operations are automated after a user's intention is inferred from examples of operations. Non-programming styles for various users including children and for various domains including games were proposed. In the database field, the example based database query languages [13] such as QBE (Query-By-Example) were studied. QBE implies that a DB query is executed by examples of concrete queries. User-friendly inquiry languages were proposed in comparison with SQL.

Our research target was different from these technologies and is for business professionals and business domains. The user's intention is definitely defined as requirement specifications without inference as business professionals with domain expertise develop software which executes their own jobs.

Therefore, this paper pays attention to a Web application in which the user interface is a Web browser because most users are familiar with how to use the Internet. Furthermore, the three-tier architecture is supposed, which has been popular recently. Generally, there are three approaches corresponding to the user interface (UI), business logic and database (DB). In our studies, application frameworks and visual modeling tools based on components were developed for EUC. The tile programming, especially, was tried to be used for the description of the business logic.

This paper presents basic approaches for Web application development in Section 1, domain-specific frameworks in Section 2, visual modeling in Section 3 and applicability and extensions for a composite approach in Section 4.

1. Basic Approaches for Web Application Development

Our approach to Web application development is shown in Figure 1. The business model at the business level is proposed by those end-users who are business professionals and domain experts. Then, at the service level, the domain model is constructed and the required services are specified. At the software level, the domain model is implemented by using components. In this approach, the granularity gap between components and the domain model is bridged by business objects and application frameworks. The semantic gap between the domain model and end-users is bridged by domain-specific technologies [14].

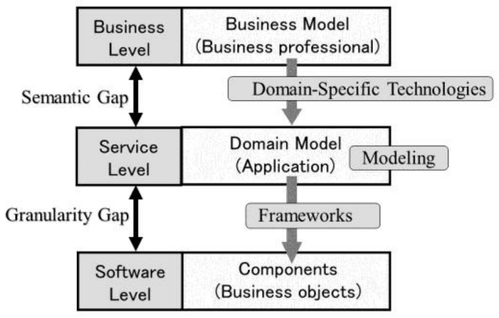

Business Level
Business Model (Business professional)
Semantic Gap
Domain-Specific Technologies
Service Level
Domain Model (Application)
Modeling
Granularity Gap
Frameworks
Software Level
Components (Business objects)

Figure 1. Technologies for end-user computing

The approaches to end-user-initiative Web application development methodologies based on the three-tier architecture are classified into the three categories of UI-driven, model-driven and data-driven processes by first focusing on any one of the UI (user interface), the model (business logic) or DB. The first two approaches are described in this paper because these are suitable for EUC, although the third approach seemed to be difficult for end-users.

2. Domain Specific Frameworks

2.1. A UI-Driven Approach

Recently, a UI-driven approach has emerged as Web applications are increasing. A typical example of this approach is the Struts framework [2], which is an open source framework for building Web applications in Java. The visual forms are defined first and then components for business logic and access to the DB are defined. In this approach, it seems to be easier for the end-user to define the UI in comparison with definitions of the model or the DB.

We have also been studying this approach for several years [5]. The UI-driven approach is proposed for the front-end system based on CBSE (Component-Based Software Engineering) [4, 7]. The systems are constructed by using UI-centered frameworks [8]. The effectiveness of the UI-driven process has been confirmed through experiences with the development of frameworks.

In our UI-driven approach, the forms are defined first and the framework is used. The business logic depending on the application is defined by the form definitions. The other business logic is embedded into the framework. However, this framework does not support the back-end system with the workflow and DB. When another framework for a reservation task such as a room reservation system was developed, a visual tool for defining the DB table was easily developed simultaneously. Although end-users can use this tool, the target DB table is limited to a simple reservation table. End-users may need an IT professional’s assistance for the UI to be implemented in JSP, components to be newly developed and a complicated DB management system.

2.2. A Target Domain for Feasibility Studies

In our research, the reservation task was selected as a target domain for a feasibility study [17]. According to the type of frame which the resources have been divided into for the reservation, the system of reservation is classified into 3 categories. The space-based reservation refers to the systems that choose the resource first according to time, and then choose the frame according to space. Ticket reservation is a sample in this category. For example, a user first selects the date of an event and then selects a seat for the event. The time-based reservation refers to the systems that first choose the resource according to space and then choose the frame according to time. Meeting room reservation is a sample in this category. For example, a user first selects a room and then selects a date for the meeting. In addition, the systems that don’t belong to these two categories, like the new book reservation system in a library, refer to the others. Table 1 shows the categories of reservation systems.

Table 1. Categories of reservation systems.

Application type	Resource attribute	Reserved item	Example	Comment
Space-based reservation	Time–base resource	Space-based Items	Concert ticket	A seat is reserved.
Time-based reservation	Space-base resource	Time-based items	Meeting room reservation	A time period is reserved.
The others	-	-	New book reservation	A book is reserved.

In reservation systems, there are user functions and system administrator functions, respectively. The function specifications are shown in Table 2. Our framework supports the most functions for users, which are login and logout, reservation, and check, modification and deletion of reservations.

Table 2. Function specifications of reservation.

Function category	User	Administrator
User administration	Login/logout	Login/logout Register/modify/delete users
Resource administration	-	Register/modify/delete resources
Reservation function	Reserve Check/modify/delete reservation	Reserve Check/modify/delete reservation

2.3. Architecture for Frameworks

The three-tier architecture of user interface, business logic and database, was supposed because this architecture is popular with Web applications. The architecture of our framework is shown in Figure 2.

The Struts framework with the MVC architecture was adopted for the presentation layer because the Struts framework is the most famous one among many existing general frameworks for the presentation layer.

The data access layer processes the data with the relational database. The common ways to handle this operation are JDBC or object-relational mapping framework like Hibernate and iBatis. In our framework, we adopted the Data Access Object (DAO) pattern [1] in this layer, and a visual tool was developed to generate the code automatically. The DAO pattern, one of the J2EE core patterns, abstracts and encapsulates all access to the database. It manages the connection with the database to obtain and store data. The DAO pattern can be highly flexible by adopting the Abstract Factory and the Factory Method [10] patterns. Since we only use relational databases, the Factory Method pattern adopted DAO was selected for our framework. In this strategy, the DAO Factory produces a variety of DAOs which are needed by the application, so it becomes much more convenient and easier for extension and maintenance.

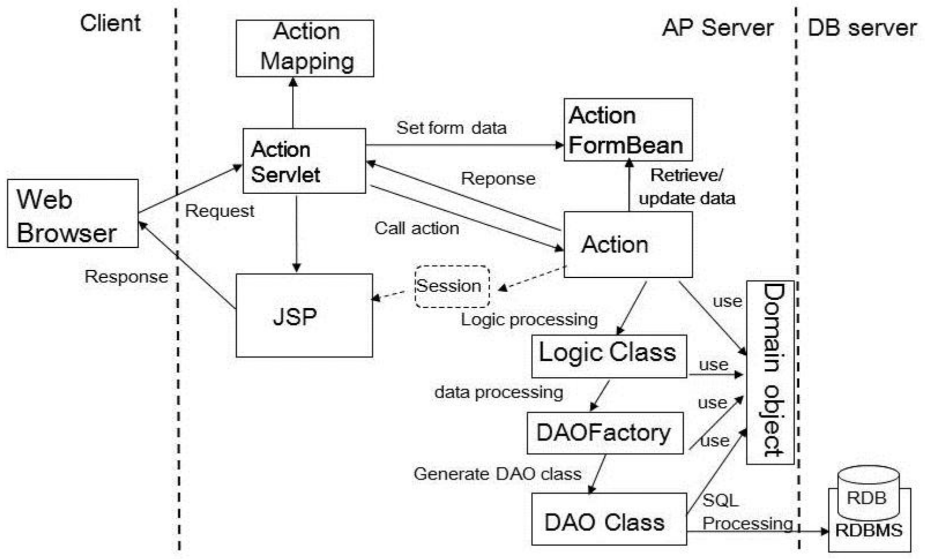


Figure 2. The architecture of our framework

2.4. Reusability by Domain-Specific Frameworks

To develop the framework, first, a sample system is developed. Then, the file is extracted from the sample system. Finally, the framework is realized completely.

A meeting room reservation system which belonged to the time-based reservation category was developed as a sample system. In this application, a user chooses one of the meeting rooms at the top page. Then while logging in, the user reserves some frames of this room which are shown in monthly calendar format or modifies/deletes reserved frames.

When the files which are necessary for the framework are extracted, the detailed contents of each file should be added and modified so that they can be used as a framework. We modified some of the contents of the file so that they could be overridden as base classes or configure files. Also, some of the files were made into libraries so the developers could choose to use them. As a result, the number of the original steps is 996 steps except the libraries from the inner framework, Struts.

This framework was applied to two systems belonging to space-based reservation and the others, respectively, in order to confirm the usability of this framework which was extracted from the application in the time-based categories.

In the first feasibility study, a soccer ticket reservation system was chosen in order to confirm if the framework is reusable for the space-based reservation category. A user first chooses the soccer match according to the time, and then a seat in which is a space notion is reserved. The functions such as reservation, check, modification, deletion and so on were implemented. The rate that the framework supplied in the system is shown in Table 3. Among the total of 1591 steps, 996 steps were supplied by the framework. In other words, the reusability was 63%. It is confirmed that the framework is useful for space-based reservation systems.

Table 3. Reusability on the first feasibility study.

	Action	Action Form	Domain Object	Logic Class	DAO	View	Action Mapping	Total
Overall System	290	94	182	139	419	397	70	**1591**
Framework	245	47	84	90	294	175	61	**996**
Specialized	45	47	98	49	125	222	9	**595**
Rate of Framework	84%	50%	46%	65%	70%	44%	87%	**63%**

Next, the second feasibility study was carried out on the others category in which an online book store system was chosen. Without time or space, a user chooses the book first, then, decides how many of the same book to buy. For this application, the new function for search was implemented besides functions for reservation. The percentage that the framework supplied in the system is shown in Table 4. Among the 1536 steps of the total system, only 540 steps were customized and the reusability was 65%. It was confirmed that the framework is also reusable in the others category.

Table 4. Reusability on the second feasibility study.

	Action	Action Form	Domain Object	Logic Class	DAO	View	Action Mapping	Total
Overall System	286	84	168	124	402	402	70	**1536**
Framework	245	47	84	90	294	175	61	**996**
Specialized	41	37	84	34	108	227	9	**540**
Rate of Framework	85%	56%	50%	72%	73%	43%	87%	**65%**

In addition, the reusability of the meeting room reservation sample system was 64%. Then the high reusability was confirmed by these feasibility studies.

2.5. A Trade-off between Reusability and Applicability

It seems that domain-specific frameworks can gain a much higher reusability than the more general frameworks. That is, the narrower the domain is, the higher the reusability will be. Then we developed another time-based reservation framework in a more narrowed domain in order to confirm the trade-off relationship between the range of the domain and the reusability.

A new framework belongs to a sub-domain of the framework for the meeting room reservation system and supports the reservation of classrooms in schools. For this target, the new framework supports functions for periodic reservation and so on. On the other hand, the time period to be reserved is fixed in advance. The number of steps for the new framework becomes 1227.

The new framework was applied to a system named the classroom reservation system. The rate that the new framework supplied in the system is shown in Table 5. The percentage of the code supplied by the framework was higher, up to 79%. The trade-off relationship between the range of the domain and the reusability was confirmed.

Table 5. Reusability of the new framework

	Action	**Action Form**	**Domain Object**	**Logic Class**	**DAO**	**View**	**Action Mapping**	**Total**
Overall System	296	98	119	116	396	459	63	**1547**
Framework	245	47	84	100	316	372	63	**1227**
Specialized	51	51	35	16	80	87	0	**320**
Rate of Framework	83%	48%	70%	72%	79%	81%	100%	**79%**

3. Visual Modeling

3.1. A Model-Driven Approach

The best solution for workflow-centered back-end systems is that end-users can get application software by visual modeling which defines forms and form-to-form transformation. Furthermore it must be easier for end-users to develop Web applications if almost all forms are visual forms as actual user interfaces and form-to-form transformations are user interface transitions.

Our model-driven approach with model transformation is shown in Figure 3. End-users develop a Web application as follows:

1. A Web application model is defined by using a visual modeling tool.
2. The Web application model is transformed into a design model of a Struts2 model by a model transformation tool.
3. The design model is transformed into Java codes of the Web application by a code generation tool.

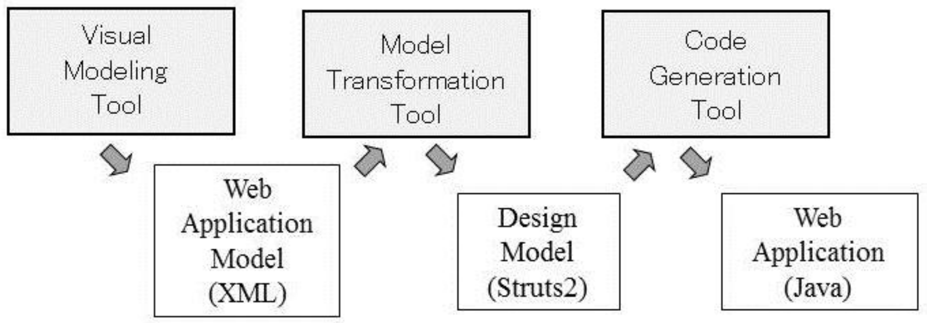


Figure 3. Model-driven approach

3.2. Web Application Model by Using the Visual Modeling Tool

For constructing the visual modeling tool, an event management system was developed as a sample. The system has the main functional specifications of registration of a user, registration of an event and recording of a reply to a request to the user for attending the event. General components with respect to user interfaces and their transition relations such as form widgets and link widgets were extracted from this application. The template commands for tile programming were extracted also.

Figure 4 shows a Web application model for a lending library which was designed by using the visual modeling tool. The left side shows a palette of general components and the right side shows an application model with five pages. The page at the upper left is the top page of the application. The page at the upper middle is the form for the registration of a book. The page at the lower right is the form for the definition of business logic of the registration of a book. The page at the lower left is the form for the announcement of termination of the registration of the book. The page at the upper right is the form for database manipulation.

The top page is displayed when the Web application is initiated at the execution time. The form for the registration of a book is displayed when the link to the book register is selected at the top page. When the properties of a book, that is, the title, the author and the publisher, are entered and the button of “send” is clicked, the page for the business logic is executed and the book information is registered to the database. Then, the form for the announcement of completion of the registration of the book is displayed. The top page is displayed again when the link “Return to Top Page” is selected.

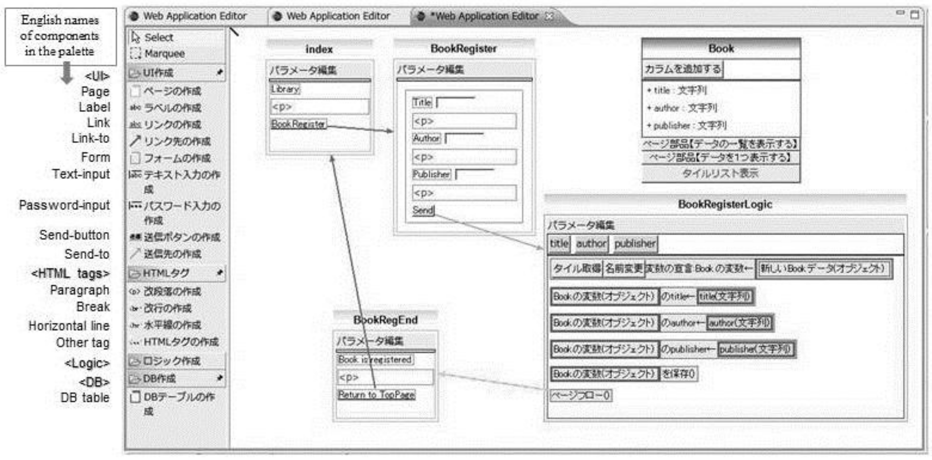


Figure 4. A Web application model by using the visual modeling tool (in Japanese)

The components for Web application models on a palette shown at the left side in the Figure 4 are classified into four categories. The English names of components in the palette are noted on the left side of the Figure 4. There are {Page, Label, Link, Link-to, Form, Text-input, Password-input, Send-button, Send-to} in the UI category, {Paragraph, Break, Horizontal line, Other tag} in the HTML tag category, {Logic, If-statement, Numeric comparison, String comparison, Object comparison, Number, String, Page flow button, Page flow to} in the Logic category and {DB Table} in the DB category.

As for this Web application model, {Page, Label, Paragraph, Link} for the Index page, {Page, Form, Label, Text-input, Paragraph, Send-button} for the BookRegister page, {Page, Label, Paragraph, Link} for the BookRegEnd page, {Logic} for the BookRegisterLogic logic and {DB table} for the Book DB table are used. The latter two items are explained in the next section.

3.3. Business Logic in Tile Programming

One of main problems for end-user-initiative development is how to describe business logic. In our past studies, some scripting languages and rules were tried. However, in these methods, end-users are required to learn some programming concepts. Therefore, tile programming was adopted for the visual modeling tool. The system prepares some templates for instruction statements as shown in Figure 5. End-users construct the business logic by combining these templates.

The lower right page in Figure 4 shows that business logic for book registration is described by tile programming as follows:

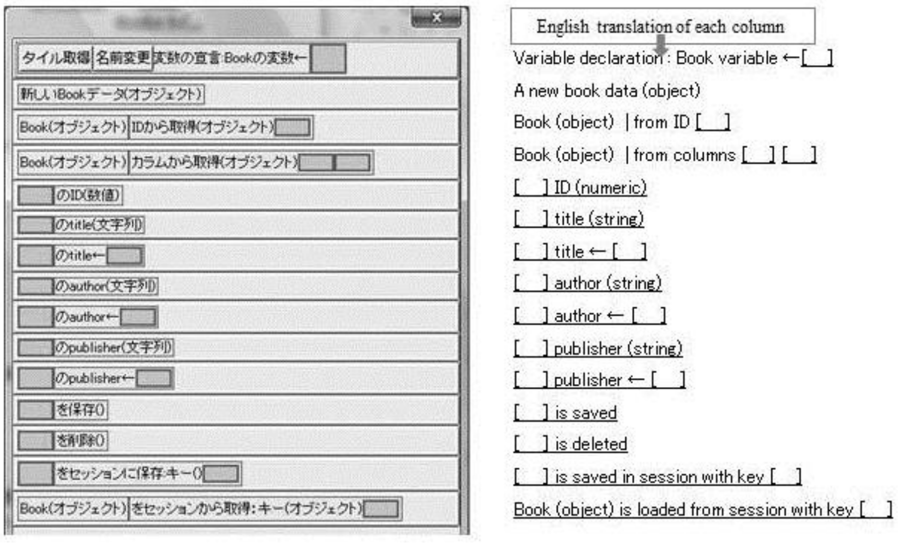
Variable declaration : Book variable <- [A new book data(object)]
[Book variable] title <- [title(string)]
[Book variable] author <- [author (string)]
[Book variable] publisher <- [publisher (string)]
[Book variable] is saved

For example, the bottom line of the form for database manipulation, which is at the upper right in Figure 4, is labeled “Tile list.” When this line is clicked, the fifteen templates for database manipulation are displayed as shown in Figure 5. The English translation of each column in the tile list is noted on the right side of Figure 5. The first template is the statement of variable declaration. This template is dragged and dropped on the form for business logic for book registration. Then the second template is the tile of a new book to be registered, and it is dragged and dropped on the blank column of the previous statement for variable declaration.

Next, the seventh template is the assignment statement which is displayed as “[__] <- [__]” and it implies that the right-hand value is assigned into the left-hand variable. It is dragged and dropped below the statement for variable declaration. Then, after the variable declaration statement is clicked, the left-hand blank column of this template is clicked. Furthermore, after the sixth template which implies the first parameter of the parameter list of “title,” “author” and “publisher” is clicked, the right-hand blank column of this template is clicked. This assignment statement implies that the title of the new book, which is passed from the form for the registration of a book at the upper middle in Figure 4, is assigned into the new record of the BOOK database via the form of database manipulation at the upper right in Figure 4. The assignment statements on the author and the publisher of the book are defined likewise.

Then the twelfth template for writing to the BOOK database is dragged and dropped below these assignment statements. After the variable declaration statement is clicked, the blank column of this template is clicked.

After the business logic are defined, the Page flow button component is put at the bottom of the BookRegisteLogic logic, and the page flow from this logic to the BookRegEnd page is defined.



	English translation of each column
タイル取得 名前変更 変数の宣言:Bookの変数←	Variable declaration : Book variable ←[ ]
新しいBookデータ(オブジェクト)	A new book data (object)
Book(オブジェクト) IDから取得(オブジェクト)	Book (object) | from ID [ ]
Book(オブジェクト) カラムから取得(オブジェクト)	Book (object) | from columns [ ] [ ]
のID(数値)	[ ] ID (numeric)
のtitle(文字列)	[ ] title (string)
のtitle←	[ ] title ← [ ]
のauthor(文字列)	[ ] author (string)
のauthor←	[ ] author ← [ ]
のpublisher(文字列)	[ ] publisher (string)
のpublisher←	[ ] publisher ← [ ]
を保存()	[ ] is saved
を削除()	[ ] is deleted
をセッションに保存:キー()	[ ] is saved in session with key [ ]
Book(オブジェクト) をセッションから取得:キー(オブジェクト)	Book (object) is loaded from session with key [ ]

Figure 5. Templates for tile programming (in Japanese)

3.4. Model Transformation Tool

The Web application model which the end-user defines by using the visual modeling tool is independent of the particular platform or the particular architecture and is described in a logical level. Therefore, the Web application model is transformed into the design model under the condition of the particular platform of the Struts 2 framework by using the model transformation tool.

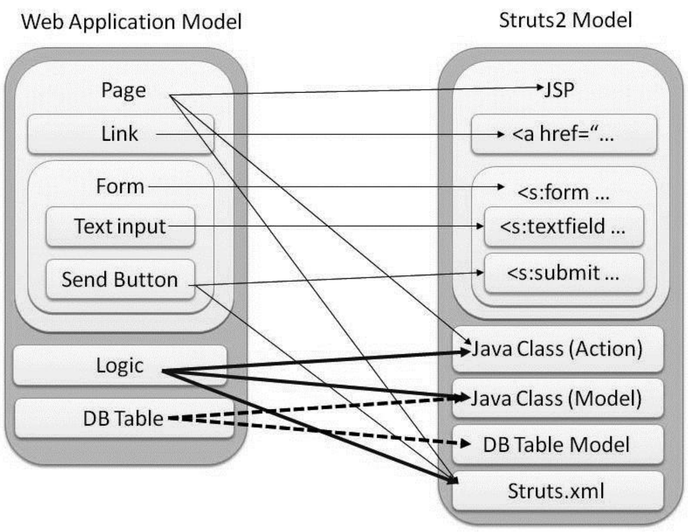


Figure 6. Mapping from the Web application model into the design model

Examples of mapping from the Web application model to the Struts 2 model are shown in Figure 6. Some of them are as simple as one page being mapped into one JSP document. Others are as complex as the business logic being mapped into Java classes and a Struts.xml document. This transformation program is described in XSLT since both models are stored in XML in the system.

3.5. Code Generation Tool

The code generation tool generates the source codes for the application, which include Java classes with class names, properties and methods, and JSP files. On the other hand, a set of files which are common to Web applications, are not included in the design model, and are appended to the source codes by the code generation tool.

3.6. Feasibility Study

This visual modeling tool was applied to library management system development [16]. The main functions were registration and deletion of a member, registration and deletion of a book, lending and return of a book, and a display of a list of books.

The Web application model is composed of 25 form definition pages, 8 business logic definition pages and 3 database table definitions. It was confirmed that it is easy for end-users to construct the Web application model.

4. Applicability and Extensions for a Composite Approach

4.1. Survey on Reuse Promotion Services

It is expected that information technology (IT) contributes to saving resources and environmental preservation for a sustainable society. For this purpose, application software is required, and then funds are needed for its development by IT professionals. However, the preparation of funds is difficult unless a profit is calculated over the development cost. The end-user-initiative development of application software is indispensable for the solution of this dilemma.

For example, let’s consider a charity shop or a thrift store which sells limited goods to limited customers in a local area. The number of goods and the number of customers will increase if business professionals develop the application for the web site in which customers can register goods to be reused or search the list of registered goods for their own use easily.

As for another example, let’s consider service counters which exist everywhere. Although some service counters already support Internet usage, many service counters have not yet done this because of a lack of funds, not a lack of technologies. If business professionals at a service counter can develop the application for a Web site, they will save resources because of the paperless system and reduce the cost of electricity by not using elevators when going to the actual counter.

For our further studies, the reuse support system for a charity shop or a thrift store is considered as a typical Web application for end-user-initiative development because it is supposed that there are a lot of variations.

First, actual support systems for the promotion of reusing second-hand items were surveyed by the searching the Internet. As a result, the following facts were confirmed:

- Many local governments support reuse promotion activities for ecological movements. Most of them use the Internet for announcements of the activities, but do not use it for practical operations. Instead practical operations are executed at the counters.
- An Internet site in practical operation for reuse could not be found.
- There are a lot of regulations for reuse promotion services and the regulations are strongly dependent on each local government's policy.

Let's give some cases of big cities in Japan. In Kawasaki city where our faculty exists, two kinds of ecological supports were found. One is the reuse promotion event that a citizen can get free and available goods which the city office collects as garbage. The other is an open-air market that a citizen can sell unnecessary goods to other citizens. Both are face-to-face dealings, although the announcement is performed via the Internet. Furthermore, there are some rules and regulations. In the reuse promotion event, a receiver must be more than eighteen years old and can propose less than three items at one event. In the open-air market, only citizens can participate in the event and goods to be sold are limited to unnecessary ones in their houses.

In some wards of the Metropolis of Tokyo, similar events are supported. In the reuse promotion service in Chuo-ward, the office keeps such goods as clothes and tableware, which become unnecessary for citizens in their houses, and are given or sold to other citizens. Furthermore, big pieces of furniture and expensive goods which cannot be kept at the office are registered with cards. In Sumida-ward, the same reuse promotion service is supported with detailed rules and regulations. The number of goods to be kept is less than six, their prices must be less than about sixty dollars, and the keeping period of each item is five weeks. The office does not bear the responsibility for trouble between a donor and a donee. These two wards use the Internet only for announcement of services.

On the other hand, in Meguro-ward, the same reuse promotion service is supported and items which are registered are displayed on the Internet site with respect to kind, specifications, price and condition. However, this site does not support the dealings with a donor and a donee.

In Osaka city, the office holds open-air markets in several wards but does not support a daily reuse promotion service.

This survey on the present status of reuse promotion service confirmed some facts. First, most reuse promotion services use the Internet for announcements but do not use it for actual dealing of second-hand items. If the Internet is used, the number of donors and donees can increase and the effectiveness of the reuse promotion service can be drastically improved. Secondly, almost all the reuse promotion services are provided under detailed rules and regulations. Although these rules and regulations are dependent on the policy of each office, the general application system for reuse promotion services must be customizable.

4.2. Requirements for the Application of Reuse Promotion Services

For extracting the basic requirements, let's consider a Web application for the reuse of laboratory equipment. In the School of Science and Technology which we belong to, a lot of second-hand equipment such as PCs are thrown away although many of them can still be used. If the reuse site is open, the available but unnecessary equipment can be registered there and someone can find and receive the reusable equipment easily. Therefore, our end-user-initiative requirement definitions method was applied to such a system, ICRS (the Ikuta Campus Reuse System).

The main rules for this system are as follows:

- Users are limited to members who have a mail address which is managed by the university, that is, teachers, officers, students, etc. for security check.
- The equipment to be given should be free for avoiding illegal dealings of university property.
- The site administrator takes no responsibility for any troubles since this site is supported by volunteers.

The main functions are described in the usecase diagram of UML, as shown in Figure 7. The user actor is the superclass of both the donor actor and the donee actor. The relations between actors and usecases are described as follows:

- The donor registers the unnecessary equipment.
- The donor replies to the donee.
- The donee receives the equipment for reuse.
- The donee requests the necessary equipment.
- The donee inquires about the registered equipment.
- The user searches a list of the registered equipment.
- The user changes his/her password.
- The administrator registers a user.
- The administrator initiates the system.

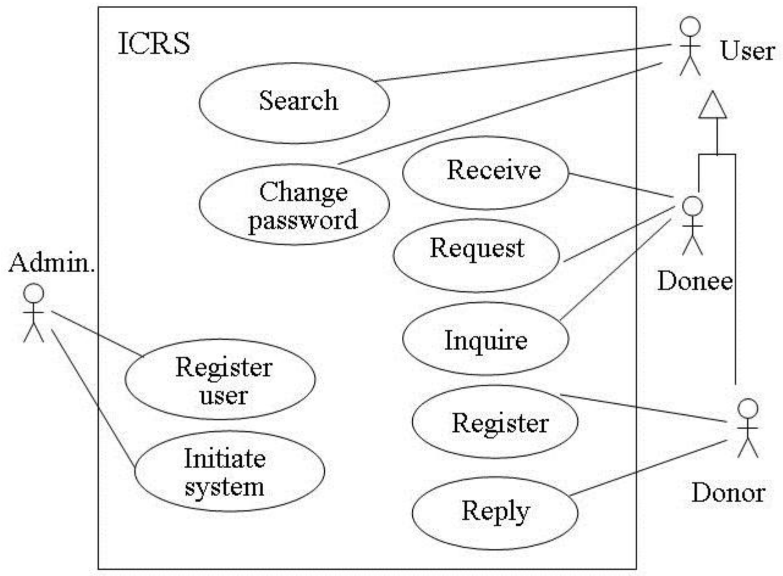


Figure 7. Usecases of the system for the reuse site

4.3. Business Logic for the Application

Most business logics for this application are related to rules and regulations, which are classified into two categories of the front-end system functions and the back-end system functions.

The following rules and regulations require the front-end system functions:

- Identification and qualification of donors and donees, including a check on limitations of age and resident area
- Consent to the rules and regulations
- The list of items which are possible to be registered and the list of items not to be registered
- Etc.

The following rules and regulations require the back-end system functions:

- Registration of donors and/or donees with the option if a donee should be registered or not
- Registration of items to be reused, including a check on the necessary conditions for registration, with the option to be mandatory about kind, attributions, photographs and price
- The issue of the item registration number with the option to be composed of figures or alphanumeric
- Search for registered items with the option of full text search and/or search by kind and attributes
- Decision of a donee of a item with the option of first come first service rule or others
- Deletion of registered items with the option of deletion time when it is received by a donee and/or after a period of time has passed
- Etc.

4.4. A Composite of both UI-Driven and Model-driven Approaches

The reuse support system seems more difficult than the reservation system or the library system which were mentioned before. The user interface is more complicate than the reservation system and the business logic is more complicated than the library system.

As for the user interface, the same technologies of domain-specific framework are applied to the reuse support system. On the other hand, considerable variations in business logic of reuse support systems exist. The visual modeling tool must support these variations. It is possible by increasing templates for tile programming.

5. Conclusions

The end-user-initiative application development approaches based on domain-specific frameworks and visual modeling were studied for Web applications with the three-tier architecture of user interface, business logic and database. The systems were implemented and the effectiveness was confirmed. Based on these results, the composite approach of the UI-driven approach and the model-driven approach was considered for more complicated applications. Further studies are needed for reinforcements of the modeling by tile programs for easy description of business logic.

References

[1] D. Alur, J. Crupi and D. Malks, *Core J2EE Patterns: Best Practices and Design Strategies*, Prentice Hall/Sun Microsystems Press, 2003.

[2] The Apache Software Foundation, Struts. [Online] Available from: http://struts.apache.org/ [Accessed 24 March, 2011].

[3] J. C. Brancheau, and C. V. Brown, The management of end-user computing: status and directions, *ACM Computing Surveys*, **25**, **4**(1993), 437–482.

[4] A. W. Brown(Ed.). *Component-Based Software Engineering*, IEEE CS Press, 1996.

[5] T. Chusho, H. Tsukui, K. Fujiwara, A Form-base and UI-driven approach for enduser-initiative development of Web applications. Proc. Applied Computing 2004, IADIS, pp.II/11-II/16, 2004.

[6] W. W. Cotterman, and K. Kumar, User cube: A taxonomy of end users. *Communications of the ACM*, **32**, **11**(1989), 1313-1320.

[7] I. Crnkovic, et al., Specification, implementation, and deployment of components. *Communications of the ACM*, **45**, **10**(2002), 35-40.

[8] M. Fayad, D. C. Schmidt, (Ed.), Object-oriented application frameworks, *Communications of the ACM*, **39**, **10**(1997), 32-87.

[9] G. Fischer, K. Nakakoji and Y. Ye, Metadesign: Guidelines for supporting domain experts in software development, *IEEE Software*, **26**, **5** (Sep./Oct. 2009), 37-44.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addsion Wesley, 1995.

[11] A. J. Ko, R. Abraham, M. M. Burnett and Brad A. Myers, Guest editors' introduction: End-user software engineering, *IEEE Software,* **26**, **5** (Sep./Oct. 2009), 16-17.

[12] H. Lieberman, (Ed.), Special issue on programming by example. *Communications of the ACM*, **43**, **3**(2000), 72-114.

[13] G. Ozsoyoglu, H. Wang, Example-based graphical database query languages, *IEEE Computer*, **26**, **5**(1993), 25-38.

[14] J. Sprinkle, M. Mernik, J. Tolvanen and D. Spinellis, Guest editors' introduction: What kinds of nails need a domain-specific hammer?, *IEEE Software*, **26**, **4** (July/Aug. 2009), 15-18.

[15] A. Sutcliffe, N. Mehandjiev, (Guest Ed.), End-user development, *Communications of the ACM*, **47**, **9**(2004), 31-32.

[16] N. Yagi and T. Chusho, A method for Web application development by end-users based on model transformation (in Japanese). IPSJ SigSE Technical Report 2009-SE-163, 81-88, 2009.

[17] F. Zhou, T. Chusho, The reusability evaluation of a domain-specific Web application framework, (in Japanese), IPSJ SIG Technical Report, 63-70, 2008.