The Classification of Matching Applications for End-User-Initiative Development

Takeshi Chusho

***Abstract*—The development of Web applications should be supported by business professionals since Web applications must be modified frequently based on their needs. In our recent studies utilizing the three-tier architecture of user interfaces, business logic and databases, web applications are developed using a domain-specific application framework and visual modeling technologies. For case studies of this approach, a matching domain is analyzed and matching applications are classified based on the user view and the system view.**

***Index Terms*—End-user computing, business logic, three-tier architecture, matching domain**

I. Introduction

The number of Web applications for end-users has been increasing in recent years. Most of these applications are developed by IT professionals. Thus, attempting to achieve automation is limited to highly specific 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 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 [6].

There are several approaches for end-user-initiative development. The UI-driven approach makes it possible to easily develop applications for the UI-centered front-end systems. It is strengthened by using domain-specific framework technologies. The model-driven approach makes it possible to easily develop applications for workflow-centered back-end systems. It is strengthened by using a visual modeling tool. Some papers have described a summary of the trends of end-user development without IT professionals' assistance [10]. End-user software engineering research for end-user programmers and domain experts has also appeared [5].

Since the target of our research is that business professionals with domain expertise develop automated applications which execute their own tasks, the user's intention is defined as a requirement specification. Therefore, this paper focuses on 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 which is popular for Web applications is assumed. Generally, three approaches corresponding to the user interfaces (UI), business logic (BL) and databases (DB) exist. In our studies, domain-specific application frameworks and visual modeling tools based on components were developed for an end-user-initiative approach. They support the construction of a graphical user interface and a simple database system [3]. The conceptual model is based on CRUD (create, read, update, and delete) operations.

As for business logic, however, it is rather difficult to support it by the same method as the other two approaches because there are various kinds of business logic. Therefore, for end-user-initiative development, business logic should be expressed from the point of view of the service providers or support systems instead of that of the clients. For case studies of this approach, a matching domain was analyzed and matching applications were classified based on the user view and the system view.

This paper presents basic approaches for end-user-initiative development in Section 2, domain modeling for matching in Section 3, the case studies in Section 4 and implementation techniques in Section 5.

II. Basic Approaches for End-user-initiative Development

A. Domain-specific technologies

Our approach to Web application development is shown in Fig. 1. A 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 using components. In this approach, the granularity gap between components and the domain model is bridged by business objects [8], patterns and application frameworks. The semantic gap between the domain model and end-users is bridged by domain-specific technologies [9].

Approaches to end-user-initiative Web application development methodologies based on this three-tier architecture are classified into the three categories of UI-driven, model-driven and data-driven processes by first focusing on either the UI (user interface), the model (business logic) or DB.

Recently, as Web applications has been increasing sharply, a UI-driven approach has emerged. In our UI-driven approach, forms were defined first and the framework was used. The business logic dependent on the application was

Takeshi Chusho is with Department of Computer Science, School of Science and Technology, Meiji University, Kawasaki, 214-0033, Japan (corresponding author to provide fax: +81-44-934-7449; e-mail: chusho@cs.meiji.ac.jp).

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

defined by form definitions. Another business logic was embedded into the framework. However, this framework did not support the back-end system with the workflow and DB.

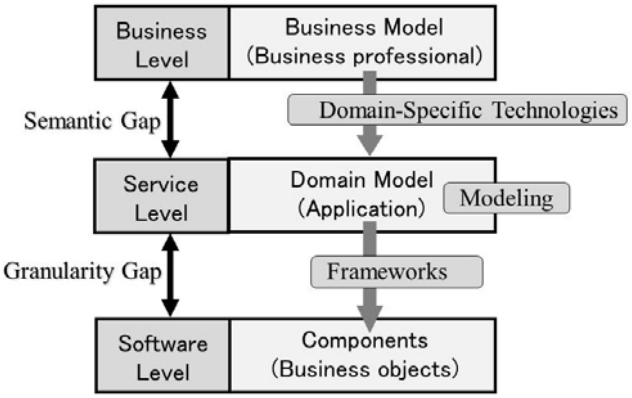

Fig. 1. Technologies for end-user-initiative development. The semantic gap between the domain model and end-users is bridged by domain-specific technologies.

One solution for workflow-centered back-end systems is the model-driven approach. Around the 1990's, object-oriented analysis and design (OOAD) technologies came out and have since become major methodologies. The unified modeling language (UML) is used for definitions of the system model. In addition, UML2.0 requires more rigorous definitions of models for the automatic generation of program codes based on model-driven architecture (MDA) [7]. The initial model is described as a platform-independent model (PIM) and transformed into some platform-specific model (PSM).

Our approach is different from the MDA-based approach. For end-user-initiative development, our model-driven approach is based on component based software engineering (CBSE) with a formula consisting of "a domain model = a computational model." This formula implies that one task in a domain model of cooperative work corresponds to one object in an object-oriented model. Therefore, it is not necessary for end-users to convert a domain model into a computational model with application architecture. The domain model is considered a requirement specification.

End-users can get application software by visual modeling which defines forms and form-to-form transformations. A Web application model which is defined by end-users is finally transformed into the Java codes of the Web application. One of the main problems for end-user-initiative development is how to describe business logic. In our past studies, some scripting languages and rules were attempted. However, in these methods, end-users were required to learn some programming concepts. Therefore, tile programming was adopted as a visual modeling tool [4]. The system prepared some tile templates for instruction statements. End-users construct business logic by combining these templates. However, it may be difficult to prepare a sufficient set of tile templates. Further studies are needed to reinforce the modeling for an easy description of business logic.

B. The UtoU Template for Business Logic Definition

The analysis of business logic is indispensable to support that end-users describe the various kinds of business logic. Therefore, first, many types of business logic were gathered by a survey on the Internet and classified into several categories in our previous work [2]. The following five types of business rules [11] were adopted since it seemed general and reasonable for the case studies: {Facts, Constraints, Action Enablers, Inferences, Computations}. In this classification, there were some problems. For example, many rules were represented from the view of service clients. For the software requirement specifications (SRS), however, rules are represented from the view of the service providers or the support system. These case studies confirmed that the classified category of each rule varies by the expression of the rule which depends on the view of the relevant client or system.

Consequently, business logic is defined from the view of service providers or the support system in this paper because the end-users for end-user-initiative development are implied to be the service providers. Then the business logic at the requirement specifications level is mapped into the combination of user interfaces (UI), business logic (BL) and databases (DB) based on the typical three-tier architecture. The following template is introduced because the UI-driven approach is suitable for the end-user-initiative development:

(1) UI: The system gets a request from a client.
(2) BL: The system processes the request.
(3) DB: The system accesses the database.
(4) BL: The system processes the results.
(5) UI: The system displays the results.

This template, named UtoU, implies that the typical process for business logic is {UI > BL > DB > BL > UI}. It is easy for an end-user to understand this process because the end-user as a business professional or domain expert is familiar with the following typical work flow such as getting a resident's card: (1) A client fills out an application for the card and hands it to the service counter at a city office. (2) A clerk at the service counter checks the application and passes it to a computer operator in the back. (3) The operator inputs the information about the client and gets the resident's card. (4) The clerk receives it and confirms the contents. (5) The clerk finally hands it to the client.

III. Domain Modeling for Matching

A. Domains of Web Applications

In our studies of end-user-initiative development, the following application domains were selected:

* Lending books
* Lending equipment
* The reservation of meeting rooms
* The reuse of second-hand articles

All of them required at least two DB tables for services. One is for member registration and the other is for object registration. The member implies a system user, including a system administrator. The object is a book, a piece of equipment, a room or an article. The basic operations are

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

CRUD (create, read, update and delete). Although the columns of a record are dependent on the object, these differences are unified by the concept of "matching" between an object provider and an object requester. Fig. 2 shows the basic behavior of system users in the use cases of UML.

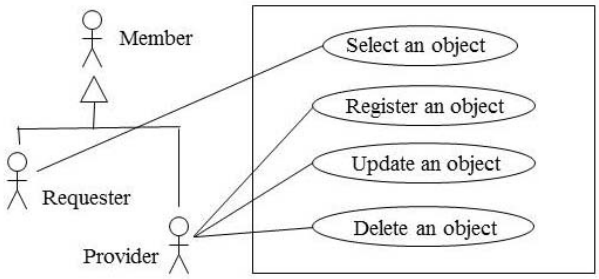

Fig. 2. A use case diagram for a matching service. The practical naming of the selection may be different for each application.

The practical naming of the selection may be different for each application. The lending of books or equipment, the reservation of meeting rooms and requests for disused articles may be employed. Furthermore, business logic must be very different for each application. Especially, within the first three applications, a matching concept is a little complicated because the use period is added into the factors used for matching in addition to a member and an object. For example, someone borrows a book for a week, borrows some equipment for one day or reserves a meeting room on the morning of the following Monday. On the other hand, in the fourth application, a matching concept is rather simple because the matching is limited to being between a member and an object with no consideration about the use period. Someone requests an article for reuse.

B. Target Domain for Matching Service

There are many kinds of applications in the domain of matching services. It is difficult to develop an application framework that can be used for all kinds of matching services because such a framework requires a lot of customization by application developers and the merit of easy development of an application is lost. Therefore, it is necessary to focus our research target on a limited subdomain. For this purpose, we analyze and classify matching services.

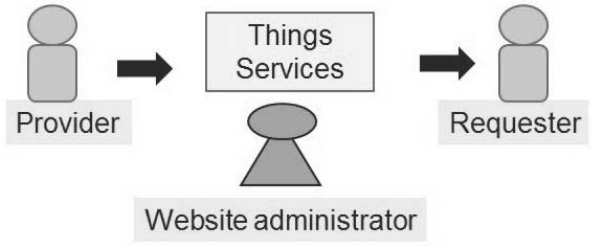

Fig. 3. A model of a website for matching service. The WHO factor of providers and requesters, the WHAT factor of things and services, and the HOW factor of an algorithm for matching on a Website are selected for classification.

A typical website for a matching service is modeled as shown in Fig. 3. The following three factors are selected for classification based on this figure:

- WHO : providers and requesters
- WHAT : things and services
- HOW : algorithms for matching decision

For the WHO factor, providers and requesters are limited to ordinary users who are equal. Services which are provided as business activities are not our research target. The requirements for such Web applications as online shopping and hotel reservations are quite different from the requirements for Web applications which are developed by end-users with the domain-specific framework of our research product, and are too complex. Such Web applications are developed by IT professionals.

Furthermore, even if both the providers and requesters are ordinary users, auction websites or websites dealing with the buying and selling of secondhand goods are not our target as well for the same reason. Reuse promotion services supported by local governments, however, are our target because almost all these services are operated at actual counters, instead of websites, which often face a shortage of talents or funds. Our research product will solve this problem effectively.

Regarding the WHAT factor, our research targets are such things which are reused or are lent as well as services such as volunteers work for snow shoveling or the repair of houses damaged by floods. Matching services which are provided as business activities are not our research target. Furthermore, such things as illegal drugs and such services as unsafe babysitter mediation are similarly not our targets. Domains for matching persons such as matrimonial agencies are similarly omitted.

As for the HOW factor, our research target is limited to domains with simple algorithms. For example, an algorithm for matching between things to be reused and requests for those things or between requests for snow shoveling and volunteers for providing the service must be relatively easy. On the other hand, applications with complicated algorithms are omitted because it is difficult for end-users to define business logic and the system generates codes automatically from the defined business logic. For example, the matching algorithms will be difficult for calling taxis or rental houses since the business logic will inevitably be complicated.

C. Analysis and Classification for Matching Service

For the analysis and classification of many kinds of matching domains, the following two criteria are introduced:

- Request for the trustworthiness of participants
- Request for quality of things or services

There are three kinds of participants, namely providers, requesters, and website administrators as shown in Fig. 3. In this paper, the request for trustworthiness of providers and requesters is set as the criterion of the horizontal axis in Fig. 4. For example, a babysitter assigned by a website may mistreat the baby. The trustworthiness of website administrators is omitted because of complexity avoidance. On the other hand, requests for quality of things or services are set as the criterion of the vertical axis in Fig. 4 because the quality of things or services is indispensable for matching domains.

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The reuse-1 in the lower left of Fig. 4 implies that a thing to be reused is given free to a requester by a provider. The reuse-2 in the upper right implies that a thing to be reused is sold to a requester by a provider. In this case, it may be difficult for the provider and the requester to be satisfied with the deal. The reuse-3 in the upper left implies that a thing to be reused is sold to a requester by a provider as well. In this case, however, the risk is reduced since the provider pays to the requester via the website. Finally, a voluntary snow shoveling in the lower right implies that a volunteer may visit a house in which an old person lives alone. Therefore, trustworthiness of the volunteer is requested.

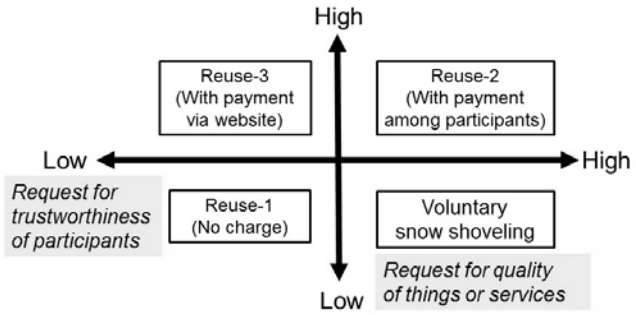

Fig. 4. The two criteria for the classification of matching services. The four kinds of matching applications are set in this map for the explanation of two criteria.

Some applications are classified into Fig. 5. Applications which are out of the scope of this paper are included in Fig. 5 for the validation of two criteria. For example, the taxi calling service is set in the lower left because taxi companies are already trusted in general. The correction of documents written by foreigners is set in the upper left since the job quality is more important than the trustworthiness of the worker. Crowd funding is set in the lower right since the trustworthiness of the requester is more important than the provider's amount of money.

Crowdsourcing for Web design or programming, renting rooms for short stays, and the babysitter assignment require both criteria. Strictly speaking, although the place of each application depends on the preconditions, Fig. 5 shows the effectiveness of the two criteria.

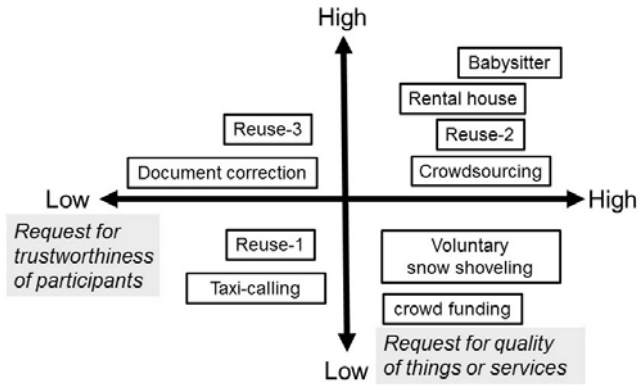

Fig. 5. Classification of matching services (the user view). This classification with two criteria is based on the user view. Applications which are out of our research target are included for the validation of the two criteria.

This classification with two criteria is based on the user view. The end-user-initiative development, however, requires the system view instead of the user view. Therefore, the following two criteria are introduced as shown in Fig. 6:

- Algorithmic complexity
- Quantity of business rules

The possibility of end-user-initiative development depends on these criteria. If the business logic which the end-users define requires complex algorithm, the automatic generation of the corresponding source codes will become difficult. If the number of business rules increase too much, the consistency among these rules will become unsatisfactory and the system may get confused executing these rules. Furthermore, it may be difficult for end-users to define the business logic even if they do so through the use of visual tools.

The system view in Fig. 6 is different from the user view in Fig. 5. Strictly speaking, the place of each application depends on the preconditions. For example, let's consider if the services of a crowdsourcing website are limited only to members. Crowdsourcing with membership is set in the upper left. Similarly, crowdsourcing without membership is set in the lower right. Our research target is limited to the domain in which the algorithm is simple and the business rules are not too many.

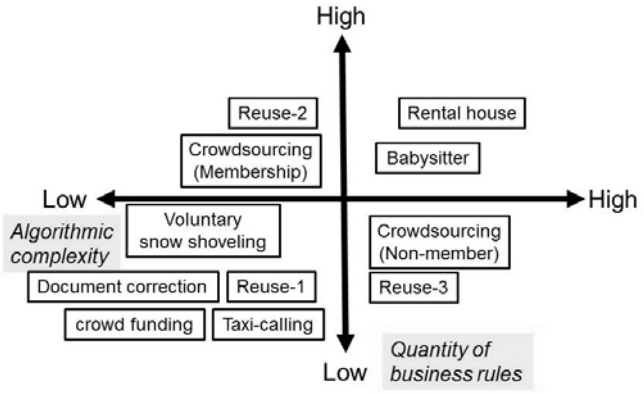

Fig. 6. Classification of matching services (the system view). This classification with two criteria is based on the system view.

IV. Case Studies

A. Reuse Promotion Services

A reuse promotion service is a kind of a matching service between a service requester and a service provider. Many similar sites are found on the Internet. There are many auction sites for buyers and sellers of secondhand items. Some free market sites for smart phones have also come out, and there are even shopping sites supporting young females selling and buying secondhand clothing. These sites require users to install smart phone applications for easy operations. For example, a user takes some pictures of a secondhand jacket and uploads them with some features and the price.

In this paper, we focus our attention on reuse promotion service only for free goods because the business logic for the reuse of free goods is limited by simple business rules in comparison with applications for auction sites or free market

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

sites. In this way, reuse support systems are the preferable application for our study. This is because it is expected that information technology contributes to saving resources and environmental preservation for a sustainable society. For this purpose, application software is required, and funds are needed for their 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 an application for the web site in which customers can easily register goods to be reused or search the list of registered goods for their own use.

B. Survey on Reuse Promotion Services

Actual support systems for the promotion of reusing secondhand items were surveyed by searching websites. As a result, the following facts were confirmed:

1) Many local governments in Japan support reuse promotion activities for ecological movements. Most of them use Web sites for announcements of activities, but do not use them for practical operations. Practical operations are executed at the actual counters.
2) There are a lot of regulations for reuse promotion services and the regulations are strongly dependent on each local government's policy.

We introduce some cases of big cities in Japan. In Kawasaki city where our faculty exists, two kinds of ecological supports were found. One is a reuse promotion event in which a citizen can get free and available goods which the city office has collected from places of disposal. The other is an open-air market in which a citizen can sell unwanted goods to other citizens. These supports involve face-to-face dealings, although the announcement is performed via the Internet. Furthermore, there are some additional rules and regulations. In the reuse promotion event, a receiver must be more than seventeen years old and can acquire less than four items at one event. In the open-air market, only citizens can participate in the event and goods to be sold are limited to unwanted household items. In some wards of the Tokyo Metropolis and Osaka city, similar events are supported.

If the Internet is utilized, the number of donors and donees can increase and the effectiveness of the reuse promotion service can be drastically improved. However, it is necessary that the general application system for reuse promotion services can be customized since these rules and regulations are dependent on the policy of each office.

C. Case Study of Business Logic Definition

Some examples of rules for the reuse support system are shown. The first example is the requirement for identification and qualification of donors and donees. The following business rules are given:

1. If a citizen is a resident in the city and is more than 17 years old, the citizen can be registered.
2. Dealers cannot be registered.
3. If a citizen requests registration, then the citizen's identification must be checked.

These rules are merged into one complicated rule and the main process of the rule is defined as follows:

(1) UI: The system displays a form for registration and gets a request from a client.
(2) BL: The system checks the request according to these rules.
(3) DB: The system accesses the database for registration.
(4) BL: The system gets the results from the database.
(5) UI: The system displays the identification number.

In this process, some details are omitted such as error handling, identification checking and identification number generation. The common error handling will be defined at the design phase. The method of the identification check depends on the status of e-Government. Citizens may already have an identification method via the Internet or they must visit an actual service counter once before using the support system. As for the identification number generation method, the system will prepare a common method such as a sequential number generator and sometimes may make the user select the form of the identification number.

The second example is the requirement for the registration of items. The following business rules are given:

1. The system must require that the donor declares that the item has been used in domestic life.
2. Large pieces of furniture are registered and kept at home.

If the reuse promotion services are limited to website services, the second rule is not necessary because it is assumed that every item will be kept at home. The main process is defined as follows:

(1) UI: The system displays a form for registration and gets a request from a client.
(2) BL: The system checks the request according to the rule.
(3) DB: The system accesses the database for registration.
(4) BL: The system gets the results from the database.
(5) UI: The system displays the results including the item registration number.

In this process, some details are omitted. The displayed form includes the check box for the declaration in addition to information about the item.

These are examples of case studies. Although it is thought that there are a lot of variations in business logic, it is confirmed that the template is useful for defining the requirements based on the typical three-tier architecture of user interfaces (UI), business logic (BL) and databases (DB). The definitions of business logic obtained using this template will promote end-user-initiative development, especially when the domain-specific application framework and the domain-specific visual modeling tools are introduced. This is because it must be easy to understand the necessary facilities for business logic.

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. Implementation Techniques

A. Web Application Generation Process

As a result of the previous case studies, a Web application based on the typical three-tier architecture is defined using a business logic definition tool and a CRUD definition tool. Our Web application generation process, named the ABC development model [1], is expressed as follows:

Application = Business logic + CRUD

A Web application is defined by end-users as follows:

(1) The user interface is defined at the logical level.

(2) The DB table is also defined at the logical level.

(3) The business logic is defined based on (1) and (2).

The CRUD definition tool is used at the step (1) and (2) and the business logic definition tool is used at the step (3). It is important to describe these three steps at the same abstraction level.

B. Visual Tools for Modeling GUI, Logic and Data

Let's consider the previous example for the registration of items and apply this process to it. The first and second steps are necessary for the step of business logic definition. Actually, these two steps are often performed simultaneously or the DB table is defined prior to the GUI definition

At the first stage of step (1), the first user interface of the UtoU template: { *UI* > BL > DB > BL > UI} is defined by listing all input columns at the logical level as follows: {the name of the donor, the member identification number, the name of the item, the details of the item, the number of photographs, the check box for a declaration that the item has been used in domestic life, the check box for a declaration that the item is not included in the list of items to be prohibited}. At the second stage of step (1), the last user interface of the UtoU template: { UI > BL > DB > BL >*UI*} is defined by listing all output columns at the logical level as follows: {the name of a donor, the item registration number}.

During step (2), the DB table of the UtoU template: { UI > BL > *DB* > BL > UI} is defined by listing all columns at the logical level as follows: {the item registration number, the name of the item, the details of the item, the number of photographs, the member identification number of the donor, the registration date, the status of "registration", "requested" and "deletion," }

At the first stage of step (3), the first business logic of the UtoU template: { UI > *BL* > DB > BL > UI} is defined by listing all input columns to be checked and internal processes as follows:

* All input columns are filled in.
* All input data are valid.
* Photographs meet the conditions if necessary.
* The identification of the registration date
* The generation of the item registration number

In the second stage of step (3), the last business logic of the UtoU template: { UI > BL > DB > *BL* > UI} is defined by listing all output columns and internal processes as follows:

* The name of the donor
* The item registration number

In this case study, exception handling is omitted to describe in detail.

As a result, it is possible that end-users of business professionals define requirement specifications of a Web application. Sometimes, they need the support of IT professionals for the implementation of complicated business logic if the components for the business logic have not been prepared in advance. One of the problems in building domain-specific framework is determining which components should be prepared. The range of applications is dependent on a related library of components. For example, it is difficult to define the suitable specifications of a registration number generator which the system supports. This is because end-users prefer a simple function to a rich function. On the other hand, some business professionals in local governments may like to customize the functions since they do not like to change from the conventional way.

VI. Conclusion

In this paper, a matching domain was analyzed and matching applications were classified based on the user view and the system view. The two criteria of the request for trustworthiness of participants and the request for quality of things or services were introduced to satisfy the user view. The two criteria of the algorithmic complexity and the quantity of business rules were introduced for the system view. It was confirmed that the latter criteria is important for end-user-initiative development.

References

[1] T. Chusho, "Conceptual modeling for Web applications and definitions of business logic for end-user-initiative development," in *Proc. The IADIS International Conference on Information Systems 2014 (IS 2014)*, 2014, pp.184-192.

[2] T. Chusho, "Classification and definitions of business logic for end-user-initiative development," in *Proc. The 11th International Conference on Software Methodologies, Tools and Techniques (SoMeT_12)*, Genoa, Italy, 2012, pp. 41-56.

[3] T. Chusho, F. Zhou and N. Yagi, "End-user-initiative development with domain-specific frameworks and visual modeling," in *Proc. The 10th International Conference on Software Methodologies, Tools and Techniques (SoMeT_11)*, Saint Petersburg, Russia, 2011, pp. 57-71.

[4] T. Chusho, and N. Yagi, "Visual modeling and program generation for end-user-initiative development," in *Proc. The 9th Joint Conference on Knowledge-Based Software Engineering (JCKBSE'10)*, Kaunas, Lithuania, 2010, pp.102-115.

[5] G. Fischer, K. Nakakoji and Y. Ye, "Metadesign guidelines for supporting domain experts in software development," *IEEE Software*, vol. 26, no. 5, pp. 37-44, 2009.

[6] A. J. Ko, R. Abraham, M. M. Burnett and B. A. Myers, "Guest editors' introduction: End-user software engineering," *IEEE Software*, vol. 26, no. 5, pp. 16-17, 2009.

[7] OMG, Unified Modeling Language, http://www.uml.org/

[8] A. P. Sinha, and H. Jain, "Ease of reuse: an empirical comparison of components and objects," *IEEE Software*, vol. 30, no. 5, pp. 70-75, 2013.

[9] J. Sprinkle, M. Mernik, J. Tolvanen and D. Spinellis, "Guest editors' introduction: What kinds of nails need a domain-specific hammer?," *IEEE Software*, vol. 26, no. 4, pp. 15-18, 2009.

[10] A. Sutcliffe and N. Mehandjiev (Guest Ed.), "End-user development," *Communications of the ACM*, vol. 47, no. 9, pp. 31-32, 2004.

[11] Wiegers, K. E., Software requirements (Second Ed.), Microsoft Press, 2003.