Conversion Method for User Experience Design Information and Software Requirement Specification

Ayumi Takeda(✉) and Yosuke Hatakeyama

Interaction Initiative Inc., Tokyo, Japan
{ayumi.takeda,yosuke.hatakeyama}@interaction-i.co.jp

Abstract. In the era of the Internet of Things, system development is becoming increasingly complex and diverse due to the need for advanced user experience design (UXD) and cross-platform systems. However, current software requirement specification methods have no way to describe UXD. We propose a description method to reflect UXD information, such as user behavior, as UXD requirements.

Keywords: UX design · Software requirement specification · Scenario · UML

1 Introduction

1.1 Changes in System Structure

Obviously, software systems are changing in the current Internet of Things era. In the past, many systems have provided functionality for a single device. Such systems tend to be very simple; they provide basic functions tailored to customer requirements.

At present, a variety of devices such as smart phones and tablets have been invented. In current systems, multiple devices cooperate to provide users with more advanced solutions and experiences. Thus, the structure of such systems is more complex than single device systems, and the system requirements must describe the structure and behavior of the system as well as user experience (UX) design (UXD) information.

1.2 Problems in Current Development

However, system requirement specifications used in current development processes have two problems.

One problem is that current system requirements documents have no method to describe UXD requirements. Essentially, engineers implement a system following only the description in a specifications document, such as a system requirement. However, the UXD process does not connect directly to the system construction process. UX designers research user behavior and consider UXs using customer journey maps, scenarios, and personas, etc. These UX design methods have no common description method; thus, description rules differ for each designer or firm. Consequently, it is

A. Marcus (Ed.): DUXU 2016, Part I, LNCS 9746, pp. 356–364, 2016.
DOI: 10.1007/978-3-319-40409-7_34

difficult to reflect UXD information of a system. However, the importance of UXD information is increasing. Therefore, it is necessary to incorporate UXD information into system requirements to develop solutions that consider UX.

In addition, multi-device systems have problems relative to usability and UX. For a multi-device system, each device typically has access to all application functions. From a UXD perspective, we believe that each device and its functions should have a suitable combination of use cases depending on user circumstance and context, and the division of the function depending on device can reduce the development costs.

1.3 Hypothesis

We have found that there is a significant gap between UX designers and engineers. UX designers design a system by focusing on user behavior in a scenario. On the other hand, engineers focus on the data-flow model. Currently, no unified description method to connect the UX design process and data-flow construction exists. UX information and data-flow appear different; however, both describe the system from different perspectives. Thus, we suspect that UXD information and the system data model can be described using unified expressions, and if they have a common method, it is possible to connect the UX design and data modeling processes.

2 Methodology

2.1 Basic Idea

According to this hypothesis, we consider it necessary to develop a common description method to express and share both types of information throughout the entire development process.

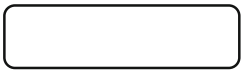
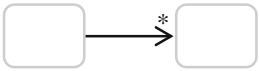
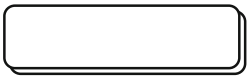
In this paper, we have adopted Unified Modeling Language (UML) as the basis of our method because it is the most common language among software engineers. With the proposed method, UX designers can write UXD information using a definite description method and engineers can receive the information to build the system according to the documentation. In addition, by adopting UML, developers can use pre-established knowledge; thus, they do not need to spend time to study this method from the beginning. UML has definite description rules and simple expressions, which makes it easy to adopt into an actual development process.

2.2 Development of the Description Method

To adopt UML for this method, we first examined UML notation and definitions. The selected diagrams for this method are use case diagrams, activity diagrams, and state machine diagrams. In UML, the use case diagram describes the required usages of a system, the activity diagram shows the flow of execution, and the state machine diagram shows the behavior of a part of the system through state transitions. We considered whether it is possible to describe a system requirement that includes UXD information using these diagrams.

Then, we examined the meaning of each original UML notation. As indicated in Table 1, the meaning of these notations in software requirement specification (UML) can be used as UXD information, such as Scene, Action, and Flow. For example, the round-cornered rectangle expresses one scene, and the text inside the rectangle is a description of the action in the activity diagram. This is interpreted as certain clipped time range. Therefore, it is possible to describe UXD information using UML notation by reading the meaning of the official UML notation. For development of complete solutions with UX, system requirements should have detailed UXD information, such as frequency, importance, and user feeling, for each scene. The UXD information can be converted to actual system requirement design, including GUI design. The expression of these additional notations is extended from the official notation to maintain UML description rules.

Table 1. Correspondence between UXD information and SRS

UML Notation	Software Requirement Specification	UXD information
	Node	Scene: Clipped time range
	State	Action: User action in a scene
	Transition	Flow: Length of arrow indicates duration
	*Additional notation*	Frequency: Number of layers is relative to frequency
	*Additional notation*	Importance: Line width is relative to importance
	*Additional notation*	Feeling: Complaints and requests about the scene

*Indicates notation except for the grey object

3 UX Description Method

3.1 Proposed Method

The proposed method realizes smooth implementation of UXD information into software requirement specification by developing a description method to describe exhaustive user activities. Thus, we have added scenario to the method.

Generally, there are three types of collaborators in solution development: business owner who build a business model, UX designers who research user behavior and design the UX, and engineers who construct the system model.

After UX modeling, each section of the system is developed, such as the software system, accounting system, and human service. In this paper, we focus on the software system; however, we believe that the proposed method can be applied to other development processes (Fig. 1).

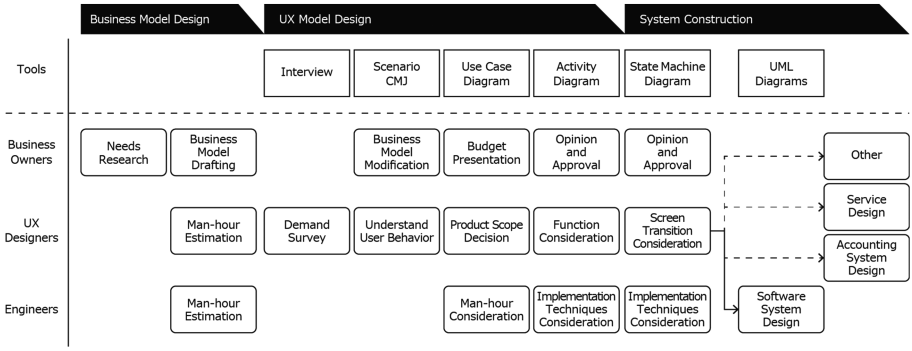


Fig. 1. Solution development process

The proposed method uses four diagrams for the UX design process. First, a scenario is used to determine unknown information about user activities and set the UX goals of the system. Next, a use case diagram is used to determine the scope of the system and its utility. Then, an activity diagram is used to describe the details of the system from the user's perspective. Last, a state machine diagram is used to describe appropriate state transitions for each platform and UXD requirement.

3.2 Scenario

The purpose of the scenario is to understand user behavior, thought and feeling. Exhaustive user information is essential to increase user satisfaction. Our scenario has a matrix structure, and it is possible to organize user information by 5W2H, i.e., when, where, who, what (which), why, how, and how much (Fig. 2), and timeline. Using this matrix structure, it is easy to find unknown user information in the matrix scenario.

In addition, this scenario has UX information, such as frequency, importance, and user feeling. These additional notations can be used to clearly describe the degree of UXD information. For example, the round-cornered rectangle means one scene (clipped time range) in the scenario, and the line-weight indicates the importance of the scene. An expression can be extended using official UML notation (Table 1). Thus, the status of a scene can be determined easily. Such UXD information can be used to determine which user actions should be systemized.

Scenario Basic Information					
Title	Win a recorded musical disc that is no longer produced at an auction	Precondition	Own a paying member account	Frequency	twice a month
Use Context	Bid at an auction site	Postcondition	Win an auction	Notes	He is correcting the records of a favorite artist. He already has 3 of all records of the artist.

Persona Basic Infomation				Prsona Experience Informaiton		Persona Character	
Age	50s	Occupation	Manager	Condition	a Bidder	Appearance	Slim
Gender	Male	Workplace	Tokyo Japan	Condition Lv.	Intermediate	Fasion	Sophisticated
Residental Area	Tokyo Japan	Position	Account Section	Motivation to Start	Correcting favorite artist	Possesion	approx. 50 records
Family Structure	Wife, 2 children	Work Environment	20s worker:50%	Purpose	Getting rare records at a low price	Personality	enthusiasm

When	Where	Who	What	(Which)	Why	How	How much	Items
Jan 7 Thu 20 : 00	Home	Mr. Brown	Get home Dinner					
			Watch TV					
Jan 7 Thu 21 : 00	Home -own room		Visit the auction web site Search the record Add the record to watch list		Want to find the item he want to buy Add to the watch list for now. Consider whether bid or not later	((device)) Home PC Keep log in Want to do watch list setting with PC		watch list info -item -My maximum bid
	Home		take a shower					

Fig. 2. Sample scenario

3.3 Use Case Diagram

A use case diagram is used to determine the scope of the system using the user information in the scenario. First, an important scene is selected from the scenario. If a scene has user feeling information in the "why" column of the scenario (Fig. 2), such as complaints and requests, the scene can be concerned a new extend function or solution in this process. In the next step, i.e., the activity diagram, user feeling information is transformed to the activity flow of a function or a solution.

Each scenario has one use case diagram because the importance of the scene changes depending on the context of each scenario.

3.4 Activity Diagram

The activity diagram can describe the flow of the solution. The functions in the scope of the system are determined by the use case diagram. Then, the activity diagram shows the detailed steps of each function from a user perspective (Fig. 3).

The scope of a single activity diagram is essentially a single solution, and a solution can potentially contain multiple types of devices. To apply suitable devices for each step in a solution, it is necessary to understand the user's situation and circumstance from the scenario (Fig. 4).

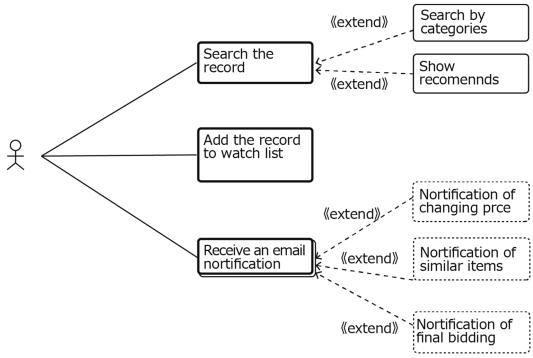


Fig. 3. Sample use case diagram

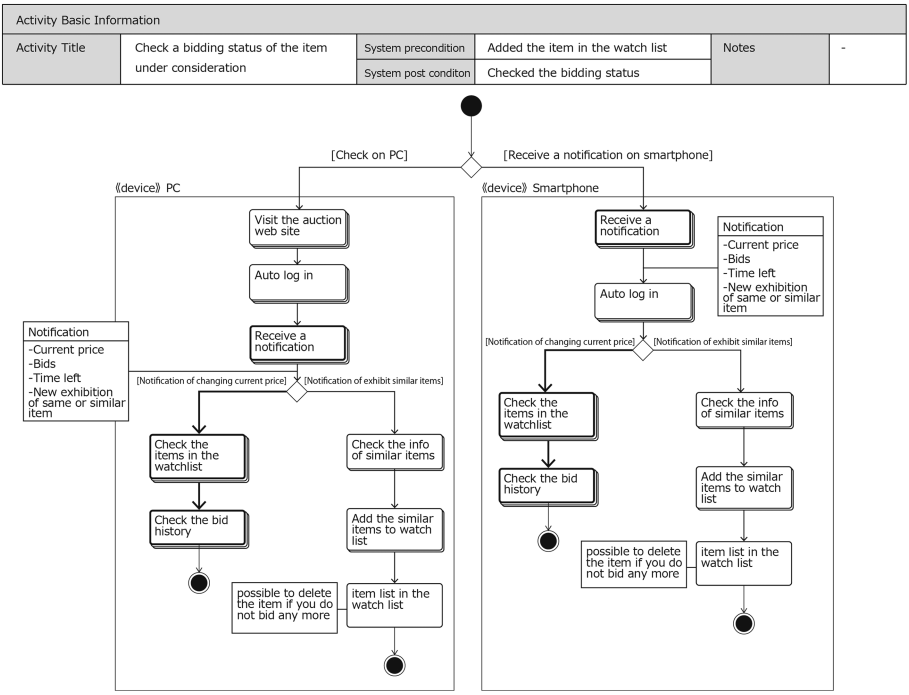


Fig. 4. Sample activity diagram

3.5 State Machine Diagram

The state machine diagram describes state transitions, such as a screen transition diagram for a GUI and its UXD requirements. Note that when the system has two devices, such as PCs and smart phones, two state machine diagrams are required.

UX designers consider a screen transition based on the activity diagram with UXD information to express an appropriate transition and GUI elements. UXD requirements, such as frequency and importance, are already applied to state transitions in a state machine diagram. For example, elements with an expression of very frequently should be accessible from any page; thus, such elements are placed in the main menu of the system. The state machine diagram can precisely describe which elements appear in the window; therefore, there is no confusion for GUI designers and engineers (Fig. 5).

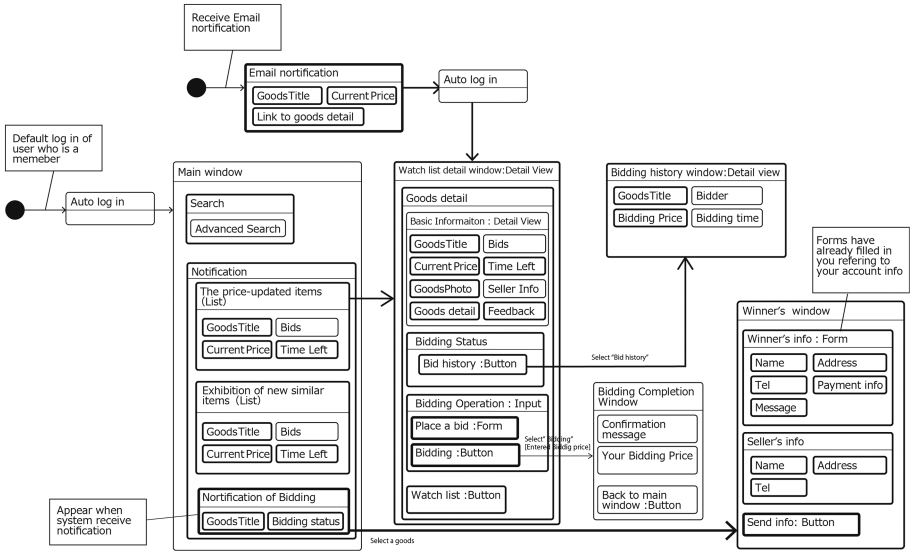


Fig. 5. Sample state machine diagram

4 Case Study

4.1 Preconditions

We examined the effectiveness of the proposed description method and evaluated whether GUI designers could perform actual GUI design using the state machine diagram. As discussed in the previous section, UXD information is already reflected as screen transitions and elements in each window.

In this case study, two GUI designers performed GUI design using the state machine diagram. The theme for the case study is a fictitious exclusive web auction system. The system is described as follows.

System Requirements.

- System: exclusive web auction system
- Devices:
  - PC
  - Smartphone

- Target user age: 20–50
- Target user gender: Male
- Basic functions:
  - Search auction
  - Watch list
  - Bidding
  - Notification

Participants.

- 1 UX designer
- 2 GUI designers

4.2 Process

1. A UX designer receives a system requirement from the business owner and researches auction web sites as a preparatory investigation.
2. The UX designer generates a scenario for the theme by interviewing target users.
3. The UX designer generates a use case diagram and extends the function.
4. The UX designer develops solutions based on user behavior in the scenario and the function(s) in the use case diagram. The UX designer then generates an activity diagram for each solution.
5. The UX designer generates a state machine diagram for each device.
6. The GUI designer receives the state machine diagram from the UX designer to design the layout.
7. Finally, the UX designer receives the layout from the GUI designer and evaluates how well the UXD requirements have been applied to the GUI compared to the UX design intention.

4.3 Case Examples

Here, we show the results and comments from the GUI designers. From the results, we identify limitations of the proposed method. Then, we consider a new solution as the next step.

GUI designer A. Designer A commented about the layout with the state machine diagram received from the UX designer. In particular, the degree of importance of elements was reflected in the layout. Designer A emphasized elements of the importance in various way of GUI layouts. For example, there is a chart that has many elements and text components; therefore, it this chart would occupy a large area of the window. If the chart is less important than other elements, the designer can place the chart in a pop-up window and increase the number of operating procedures that can be used to reach the chart.

GUI designer B. Designer B was also able to design the layout. The state machine expresses the group structure of the elements in the window; thus, the designer could place each element in an appropriate group. However, designer B indicated that the state machine diagram lacked some information about the GUI design, i.e., a detailed explanation of each element, such as functions and data length requirements.

Currently, there is no item to describe detailed element information in the state machine, such as functions and data length requirements. As a result, we have added items to be entered into the diagram or additional documentation.

5 Conclusion

To summarize, using the proposed method, it is possible for UX designers to describe UXD requirements that can be shared with engineers and GUI designers. The effectiveness of the proposed method was verified for software and GUI development.

However, the proposed method has some limitations. First, it is possible that the proposed method could increase development person-hours because it incorporates UXD into system requirements. Current system requirements do not include UXD requirements; therefore, we are concerned that the proposed may incur additional development load. Second, the proposed method describes a system using natural language and diagrams. In this description method, it is difficult to create a prototype, e.g., actual system screens, from the diagrams.

In the future, we will consider the effectiveness of the proposed method for human services other than software systems. Other issues are worth considering. First, this method might incur extra person-hours during development because it incorporates UXD requirements. Next, the proposed method should be examined for the affinity to the iteration of design solutions, including UXD development and verification. In addition, we are considering applying the proposed method to broader components of a service solution, such as interaction among humans.

References

1. Yamazaki, K., et al.: Experience Vision. Maruzen Publishing Co., Ltd., Japan (2012)
2. Gou, K., et al.: Structured scenarios method for human-centered design. In: Proceedings of the Annual Meeting of Japan Ergonomics Society Japan Ergonomics Society 49th Conference, Japan (2011)
3. Kodama, K.: The Essence of UML Modeling. Nikkei Business Publications Inc., Japan (2004)
4. OMG: Unified Modeling Language (OMG UML) Version 1.3, OMG Document number formal-00-03-01, USA (2000)
5. OMG: Unified Modeling Language (OMG UML) Version 2.5, OMG Document number formal-15-03-01, USA (2015)
6. Pilone, D., Pitman, N.: UML 2.0 in a Nutshell. O'Reilly Media Inc., USA (2006)
7. Takeda, A., Hatakeyama, Y.: A Proposal for a Description Method of UX Design Supporting Communication among Developers. Human Interface Society SIG UXSD, Japan (2015)