ISLAYPUB - A User-Participatory Service for Video Game Creation on the Web

Kazuya Shindome[†], **Shusuke Okamoto**[††], **Michitoshi Niibori**[†††], **Tatsuhiro Yonekura**[†], **Masaru Kamada**[†]

[†]*Graduate School of Computer and Information Sciences, Ibaraki University, 4-12-1, Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan*
[††]Faculty of Science and Technology, Seikei University, Musashino, Tokyo 180-8633, Japan
[†††]Learning-i Co., Ibaraki University, VBL 1F, Nakanarusawa-cho 4–12–1, Hitachi-shi, 316–8511 Japan

Summary
A web-based user-participatory service for video game creation is constructed. Its programming environment is based on a graphical programming language called Islay and is written in Adobe Flex so that it runs on any Flash-capable web browsers. Users define games in terms of the intuitively comprehensive state transition diagrams. Executable game files are automatically generated in the Flash SWF format. The executable files are published along with all the programming resources including the diagrams so that other users can play, modify and enhance the games. This service works as an open source forum devoted to video game creation for everyone who has a web browser.

Key words:
User participation, video game, application, service on the web, consumer-generated media

1. Introduction

User participation is crucial for the consumer-generated media such as Wikipedia[1]. Users are invited to create, develop and enhance the contents evolutionarily. The user-participatory paradigm has also been introduced to the video sharing services. BubblePLY[2] is a service where people can modify video clips taken from YouTube[3] by overlaying word bubbles on the video clips. A similar service based on the Synvie technology [4] is available at NICO NICO DOUGA[5] where users can superimpose their animating comments on the video clips in real time.

Participants of those services can be classified into three classes: creators, semi-creators, and consumers. Creators contribute original contents for the sake of their fame or sacrifice. Their contents give novelty and vitality to the service. Semi-creators enhance or modify the original contents. They keep the service vital and interesting or even make it more attractive. Consumers view the contents just for fun. They contribute financially to the service by means of viewing or clicking advertisements. Although a service needs all those classes of participants to continue its operation, it is the semi-creators that constitute a new class of users crucial for the user-participatory services.

The present paper addresses the problem of making a user-participatory service for creating video games. Unlike the user-participatory video services where creators have only to publish their movies, a user-participatory service for video game creation is required to have the creators disclose the source code and other programming resources to the public. Otherwise, semi-creators cannot play their role of enhancing and modifying the video games. In addition, a common and handy programming environment is required to be available on the web. A web-based system named *Islay Pub* that satisfies the above two requirements is presented in this paper.

2. Programming Language

A comprehensive programming environment is important for inviting many creators and semi-creators to the service. *Islay* is an object-oriented graphical programming language for creating interactive animations and video games. It was developed by [6] and 0. Its programming environment has been available on the Microsoft Windows and Linux platforms. Its users include 10-year-old boys and girls.

Islay employs the classical state transition diagrams to describe the behavior of characters. Users compose a diagram consisting of ovals and arrows, which correspond to states and transitions, respectively. An oval has an image representing the character’s appearance and actions in the state. An arrow is attributed by a condition such as mouse-click and collision on which the transition takes place. The visual comprehensiveness of states differentiates Islay from the script-based languages such as Squeak [8] and Scratch [9]

Figure 1 shows an example of character definition by the Islay editor. The intuitively comprehensive nature of the state transition diagrams allows us to understand that the fire engine will drive to the right until it crushes with

Manuscript received January 5, 2009
Manuscript revised January 20, 2009

the wall and vice versa and that shifts to the states of shooting the water on every mouse click. The character representing the water bubble is defined in another tab.

The video game definition is stored to a single file in the format called XIA, which contains the state transition diagrams and the pictures. An XIA file can be translated into several languages like C, Java, JavaScript and Flash. Figure 2 is a screen shot of the gaming.

We shall develop a web-based user-participatory service for video game creation on the basis of this programming language and its translator [6] into Adobe Flash movies [10]

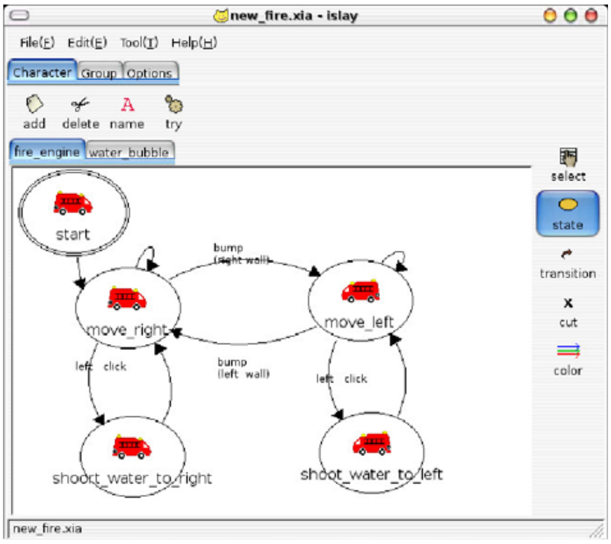
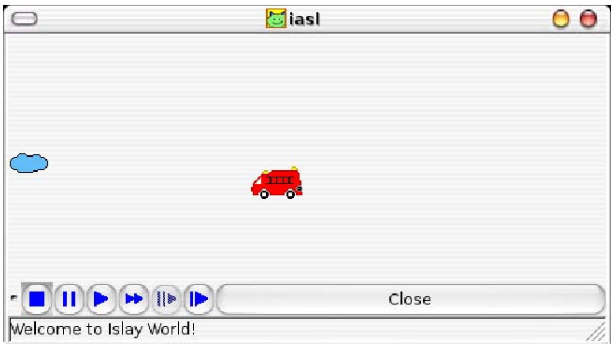


Figure 1: A screen shot of the *Islay* editor.



Welcome to Islay World!

Figure 2: A screen shot of gaming.

3. System Design

As discussed in the introduction section, we have two requirements for the user-participatory services for creating video games:

1. Creators as well as semi-creators must be forced to disclose all of the programming resources including source codes, i.e., an XIA file in the case of Islay, along with the executable game file, i.e., a Flash SWF file.

2. The programming environment must be handy and commonly available on the web.

The requirement 1 can be easily fulfilled by placing the translation capability only at the server. All creators and semi-creators must upload their XIA files to the server in order to obtain the Flash executable SWF files. The server publishes the executable SWF files always accompanied by the XIA source files to be reused or recycled by other semi-creators.

The programming environment must implement the capability of present Islay editor and must run on the web. A new editor on the web shall run on the Flash platform. Then the environment gains not only handiness and common availability to fulfill the requirement 2 but also coherence with the executable file format.

4. User Scenario

Creator's scenario follows:

1. The creator authors a game on the Islay Pub editor.

2. The Islay Pub editor uploads the game definition file to the file manager on the server.

3. The translator works on the game definition file and produces the Flash executable SWF file of the game.

4. The creator plays his own game on the game engine to check if it works.

Semi-creator's scenario follows:

1. The semi-creator plays a game on the game engine.

2. Inspired by the game, he retrieves its game definition file from the file manager to the Islay Pub editor.

3. The semi-creator edits the game definition file to modify the game.

4. The Islay Pub editor uploads the modified game definition file to the file manager.

5. The translator works on the game definition file and produces the Flash executable SWF file of the game.
6. The semi-creator plays his modified game on the game engine to check if it works.

Consumers play the games on Flash-capable web browsers.

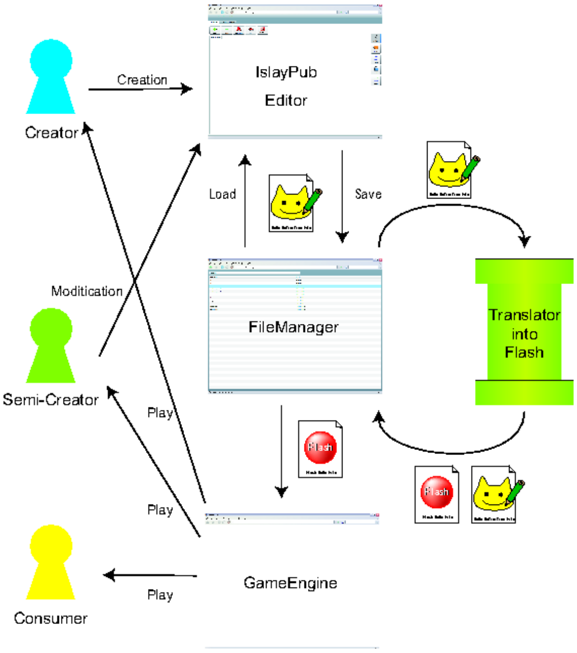


Figure 3: System overview.

5. Implementation

The modules of the system in Figure 3 have been implemented in the following way.

The editor has been developed in Adobe Flex 3[11] as client software that runs on Flash-capable web browsers.

Figure 4 is a screen shot of this editor. Each menu button is an object having a method to implement an editor function. A state transition diagram is managed by the object of tabbed workspace under which objects representing states (ovals) and transitions (arrows) are controlled. The workspace object catches all the click events on any of the objects inside the workspace and activates an editor dialog according to the context. A double-click on an oval activates a dialog to select a picture representing the character appearance.

Each object holds all the data that concerns its own definition. In order to collect the whole data for game definition, the main object of editor requests the tabbed workspaces to send back the data that they have. A workspace object collects the data from the objects of states and transitions under its control in the same way.

A game definition file is uploaded in the JSON format from the editor to the file manager at the server. The translator [6] written in C is employed to translate every uploaded game definition file into a Flash SWF file. The file manager serves users via a file browser where a list of definition files, the executable SWF files, authors, date and time, and abstracts of the games are displayed. The file browser is written in Flex 3 and communicates with the file manager in the PHP.

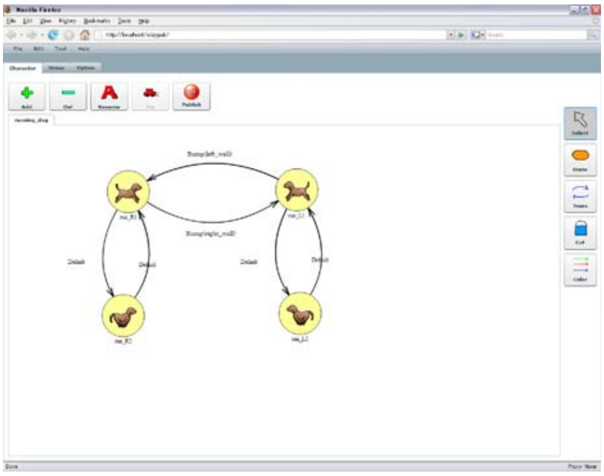


Figure 4: A screen shot of *Islay Pub* editor.

6. Conclusion

A web-based user-participatory service for video game creation was developed of which beta version is released at http://islaypub.net/. Its main feature is that all the programming resources are always published with the executable games. The translator hidden in the central server forces users to follow general public licenses. Another good feature is the comprehensive programming environment that runs on the common platform of Flash. Social mechanisms like ranking the games according to citations and tracing the family tree of the games are yet to be introduced into the system.

Acknowledgments

This work was partially supported by the JSPS Grant-In-Aid no.18300027.

References

[1] Wikimedia Foundation, Inc., Wikipedia: http://wikipedia.org/, (Accessed Jan. 16. 2009).
[2] PLYmedia, Inc., Bubble PLY: http://www.bubbleply.com/, (Accessed Jan. 16. 2009).
[3] YouTube, LLC. YouTube: http://youtube.com/. (Accessed Jan. 16. 2009).
[4] Project Synvie, Synvie: http://synvie.net/index_en.html, (Accessed Jan. 16. 2009).
[5] niwango, Inc. , Nico Nico Douga: http://www.nicovideo.jp/, (Accessed Jan. 16. 2009).
[6] M. Nakagawa, S. Okamoto, M. Kamada and T. Yonekura, “Flash movie authoring environment based on state diagram”, Proceedings of the Fifth Workshop on Network and System Support for Games 2006, (NetGames 2006), Singapore, CDROM Proceedings, 45, Nov. 2006.S. Okamoto, M. Kamada and T. Nakao, “Proposal of an interactive animation authoring tool based on statetransition diagrams”, IPSJ Transactions on Programming, 46:19–27, Jan. 2005.
[7] S. Okamoto, T. Hashimoto, M. Kamada and T. Yonekura, “Easy creation of network-based interactive animations using islay”, IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 2005, 1, 412-415, Aug. 2005.
[8] Squeak Foundation, Squeak Smalltalk: http://www.squeak.org (Accessed Jan. 20. 2009).
[9] Lifelong Kindergarten Group, Scratch: http://scratch.mit.edu (Accessed Jan. 20. 2009).
[10] Adobe, Flash: http://www.adobe.com/products/flash/, (Accessed Jan. 16. 2009).
[11] Adobe, Flex 3: http://www.adobe.com/products/flex/, (Accessed Jan. 16. 2009).

Kazuya Shindome was born in Minamisastuma, Kagoshima, Japan in 1985. He received his bachelor's degree in engineering in 2007 from Ibaraki University. He is soon finishing his master's degree at the same university. He has been working on advanced technology as programmer and system engineer.

Shusuke Okamoto was born in Tokyo, Japan, in 1965. He received the B.Eng., M.Eng. and Dr. Eng. degrees from Seikei University, in 1989, 1991 and 1994. In 1994 he joined the University of Electro-Communications as a research associate. From 1997 to 1999, he was an Assistant Professor at the Graduate School of Information Systems, the University of Electro-Communications. From 1999 to 2005, he was an Assistant Professor at the Department of Computer and Information Sciences, Ibaraki University. He is currently an Associate Professor at Faculty of Science and Technology, Seikei University. His current research interests are programming environments, parallel processing and computer architectures. Dr. Okamoto is a member of IEEE, ACM, IPSJ, IEICE and JSSST.

Michitoshi Niibori was born in Tsukuba, Ibaraki, Japan in 1979. He received his bachelor's degree in engineering in 2005 from Ibaraki University. Just before graduation, he founded a software company, Learning-i, Ltd., and is currently working as its president.

Masaru Kamada was born in Hitachi, Ibaraki, Japan in 1962. He received his bachelor's (1984), master's (1986) and doctoral (1988) degrees in engineering from the University of Tsukuba and worked for the same university as a faculty member. In 1992, he joined Ibaraki University where he is currently a professor of computer science. He was a pos-doc fellow at Mathematische Systemtheorie of ETH Zurich in 1993-1995. He served the engineering science society of IEICE as an associate editor of its Japanese transactions (1997-2000), publications secretary (1998-1999) and webmaster (1999). He has been serving *Sampling Theory in Signal and Image Processing* as its secretary since 2003 and *IEEE Transactions on Industrial Electronics* as an associate editor since 2009. He is working also on web-based services, signal processing, and computer forensics. Dr. Kamada is a member of IEEE, IEICE, and EURASIP.

Tatsuhiro Yonekura was born in 1954. He received his B.S. and M.S. from Nagoya University in 1979 and 1981 respectively. He then joined Yamatake Honeywell, working on development of a distributed computer control system till 1990. He received his Ph.D. Degree in Information Science from the Nagoya University in 1991. He joined Ibaraki University in 1991. He is a Professor at Department of Computer and Information Sciences at Ibaraki University. He is a Professional Member of ACM, IEICE (The Inst. of Electronics, Information and Communication Engineers), BMA (Business Model Association) and VRSJ (Virtual Reality Society of Japan). His research interests are in the area of Human Communication on Internet, E-business, E-society and Distributed Virtual Environment.