# clooca : Web based tool for Domain Specific Modeling

Shuhei Hiya*, Kenji Hisazumi, Akira Fukuda, and Tsuneo Nakanishi

Kyushu University 744 Motooka Nishi-ku, Fukuoka 819-0395, Japan
{hiya,nel,fukuda,tun}@f.ait.kyushu-u.ac.jp
http://portal.isee.kyushu-u.ac.jp/

**Abstract.** Clooca is a development environment that allows us to build up domain-specific modeling languages, or DSMLs, and their code generators. DSMLs are widely used to improve productivity and quality of developing software to raise the level of abstraction and to generate a fully functional software codes. Currently some tools to develop software are provided as web services. We can use the tool without installing any software and without professional knowledge of software development. Even domain specialists who do not know how to develop software can use it. Therefore, we tackle with developing software-as-a-service(SaaS) type of DSML tools to bring benefits of both DSML and SaaS type of development tools. We chose to use clooca in an education setting, because this tool solves some problems in that field. The youtube link is http://www.youtube.com/watch?v=VS5VNBOYA_o.

**Keywords:** Domain-Specific-Modeling, DSML, SaaS, Code Generator

## 1 Introduction

Clooca is a development environment that allows us to build up domain-specific modeling languages (DSMLs), and their code generators. Clooca is provided as a service, similar to Gmail. Clooca is simple to use: open a web browser that supports HTML5 and JavaScript, input the URL of clooca, and start using it. Instllation and setup are not required. DSMLs are widely used to improve the productivity and quality of developing software[1][2][3]. Clooca raises the level of abstraction rather than describing programming languages. By describing these abstract models, clooca allows domain specialists to develop software without knowledge of how to develop software using a programming language. The tool can generate fully functional codes from these models. We used clooca in a software modeling education setting.

Education can derive many benefits from using a development tool as a web service. In software modeling courses, many students use modeling tools, which are time consuming to install. As an alternative, teachers can use clooca to

[*] Graduate School of Information Science and Electrical Engineering of Kyushu University.

eliminate the need for students to install the tools and professional knowledge of software development. In addition, teachers and students can share and analyze their model in real time on clooca.

DSML tools has two benefits in the education field. First, It raises the level of abstraction rather than describing programming languages. Thus, teachers can define a modeling language according to each student's level of ability. Second, because DSML tool can generate fully functional codes from these models, students can execute the model soon after draw it, and they can check how the model works.

The remainder of this paper is organized as follows. We discuss the architecture of the clooca and its implementations in Section 2. In Section 3, we demonstrate the clooca in a real usage scenario. We conclude this paper in Section 4.

## 2 Implementation of clooca

**JavaScript Object Notation-Based Model Representation** We identify requirements for the scheme of DSML definition for a SaaS type of DSML tool as follows. The model representation of the scheme should be easy to serialize and deserialize. The model must be serialized and transferred between a web browser and a server, because clooca is a web browser-based tool. The serialized form should be used on many platforms used by constructing web services (e.g., JavaScript, Python, PHP, and Ruby). We are able to serialize and deserialize the model without compiling or generating any stubs. The system should be able to deal with these models dynamically. The scheme allows us to define DSMLs easily. The representation of the model should not be complicated. The model should be defined not only by the tools, but also by hand. In other words, the representation form of the model should be human readable. Figure 1 shows a DSML model. An extract of JavaScript Object Notation (JSON) [4] expression of the DSML model is shown in Figure 2.

**Clooca architecture** Figure 3 shows the architecture of clooca, which consists of a server and a client. The clooca server runs on web servers such as Apache and has functions to load and store models and meta-models according to requests from the clooca client. It is a kind of database wrapper. We employ JavaScript to implement the server. The clooca client runs on web browsers that support HTML5 and JavaScript such as Google Chrome. It consists of the workbench and the model editor, shown in Figure 4 and Figure 5, respectively. The workbench is a tool to define DSLs and to develop code generators. The model editor is a tool to edit concrete DSLs.

**Workbench** The workbench is a tool to define DSMLs and to develop code generators. Figure 4 shows a screenshot of the clooca workbench. It is quite complicated to define a DSML to describe a JSON description. However, with the

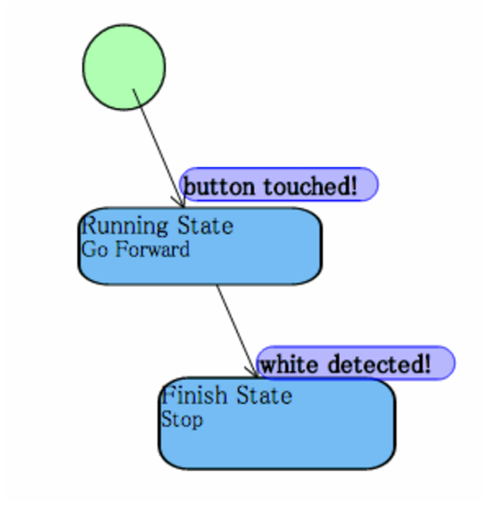
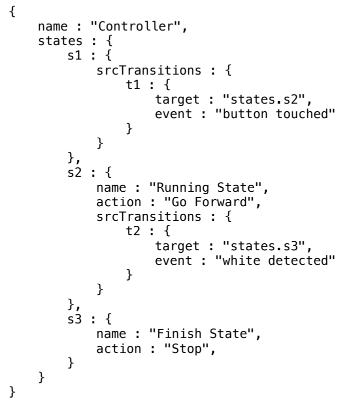


**Fig. 1.** An example of model.



```
{
    name : "Controller",
    states : {
        s1 : {
            srcTransitions : {
                t1 : {
                    target : "states.s2",
                    event : "button touched"
                }
            }
        },
        s2 : {
            name : "Running State",
            action : "Go Forward",
            srcTransitions : {
                t2 : {
                    target : "states.s3",
                    event : "white detected"
                }
            }
        },
        s3 : {
            name : "Finish State",
            action : "Stop",
        }
    }
}
```

**Fig. 2.** JSON expression.

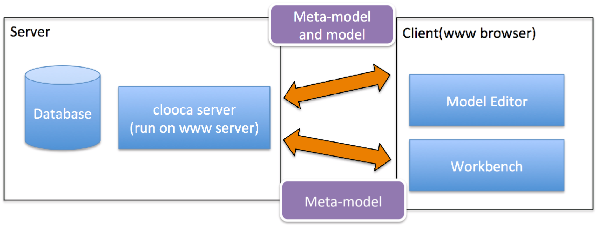
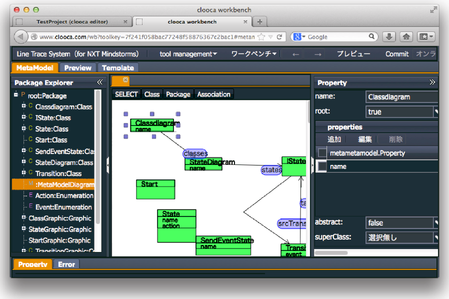


**Fig. 3.** Architecture of clooca.

workbench, we can define DSMLs using a graphical user interface (GUI), rather than describing the JSON description mentioned in the previous section. The workbench allows us to write the definition with table forms of diagrams, objects, relationships, and properties. We can develop templates to generate codes in the workbench. The next subsection shows how to define code templates.

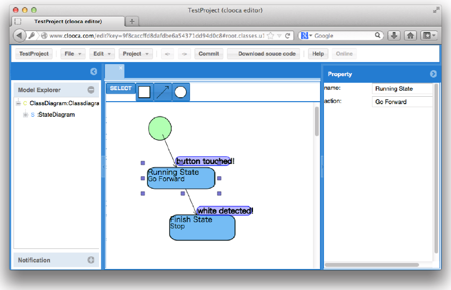
**Model Compiler** The model compiler framework facilitates the development of model compilers which obtain a DSL description and generate source code according to a template and the DSL description. The model compiler runs on the server side. The model compiler framework is developed with ejs[5]. ejs is a template library written in JavaScript.

Figure 6 shows an example of the code template to generate a name list of objects of“ action ”defined in Figure 2. We can access all of the diagrams, objects, relations, and properties in the DSML model to describe an object reference code



name: Classdiagram
root: true
abstract: false
superClass: 選択無し

**Fig. 4.** A screenshot of the workbench.



name: Running State
action: Go Forward

**Fig. 5.** A screenshot of the editor.

of JavaScript in the code template. All of them are bound to ”_” as a top-level variable. For example, if we want to access all the objects of ”action,” we can get a reference to the states to describe ”_.states”. In the code template, we can use Javascript expressions with special markers. A line that begins with the ” <%” marker indicates that the line is a control structure.” <%expression% >” means that it will be substitute a concrete string generated to evaluate JavaScript expression.

```
<%for(var key in _.states) {%>
 <%=_.states[key].action%>
<% } %>
```

**Fig. 6.** An example of a code template.

**Model Editor** The editor is a tool to draw models defined by the workbench and to generate codes according to code templates and the model. Figure 5 shows a screenshot of the clooca editor.

## 3 Evaluation

As a test, we use clooca to define diagrams and code templates for some applications.

### 3.1 A software modeling course

We used clooca in a software modeling course[7]. where approximately 80 students designed a class diagram and a state machine diagram for a line trace robot and then ran the robot. The teacher defined the DSML for the line trace robot on clooca, and the students used the DSML. Teachers are always concerned about modifying lab environments to employ special tools; however, clooca eliminates this concern because it does not require installation, as mentioned above.

### 3.2 For a project-based learning and in a classroom

We also used clooca in a classroom setting to lecture about the concept of model-driven development and DSMLs. The teacher defined a simple class diagram and a state-machine diagram of the UML and developed code generators, which generate JavaScript codes running on a web browser according to the input model. We can execute the generated code on browsers. The number of students in this classroom setting was approximately 20. In this test, we successfully used clooca without any installation efforts.

## 4 Conclusion

Clooca is a SaaS type of development environment that allows users to build DSMLs, and their code generators [6] . We can use clooca without any installation efforts. In this paper, we proposed a novel method to define DSMLs that is suitable for web based services including clooca. We described the clooca architecture and its implementations that employ a method to define DSMLs. We described some cases to prove the usefulness of clooca. These cases demonstrate benefits of SaaS type of DSML tools.

Our future work is to evaluate quantitatively with effort according to the deployment of clooca: for example, we should compare the time it takes to set up and installation with the conventional tool.

## References

1. S. Kelly and J.-P. Tolvanen. Domain-Specific Modeling: Enabling Full Code Generation. Wiley-IEEE Computer Society Press, 2008.
2. S. Cook, G. Jones, S. Kent, and A. C. Wills. Domain Specific Development with Visual Studio DSL Tools (Microsoft .Net Development). Addison-Wesley Longman, 2007.
3. Graphical Modeling Project. http://www.eclipse.org/modeling/gmp/ [Online; accessed July 1 2013].
4. Introduction JSON, `http://www.json.org/` [Online; accessed July 1 2013].
5. ejs Javascript Template, `http://embeddedjs.com/` [Online; accessed July 15 2013].
6. clooca `http://www.clooca.com`
7. K. Masumoto, M. Kayama, S. Ogata, K. Itoh, M. Hashimoto, and M. Otani. A Basic Study of Teaching Methods for Model-Based Design with Error Analyses of Beginners. ESS2012. pp101-109, 2012.