Studies into interactive didactic approaches for learning software design using UML

Stikkolorum, D.R.

Chapter 4

WebUML - a UML Class Diagram Editor for Research and Online Education

In this chapter we discuss our web based educational UML class diagram editor: WebUML. WebUML was developed in order to support the study of i) students' design strategies ii) automatic evaluation of students' software designs iii) guidance of students during their software design activities. WebUML was used during multiple experiments. In this chapter we present an overview of the tool, discuss experiences and future extensions.

4.1 Introduction

Our research aims to observe and analyse students' software design activities. In order to gain knowledge about matters such as strategies or common difficulties we typically designed experiments with classroom settings in which students perform a design task with the use of a UML editor.

Using UML modelling tools often involves installing, configuring and learning to use the tool. What tool(s) to use is an ongoing discussion between educators [2]. Educators discuss the following issues when (considering the) use of tools:

- Tools should be designed specifically for pedagogy.
- There should be enough support available for the tool, either online or facilitated locally by the university.
- Tools can be used to encourage students to produce "good" models.
- Modelling tools could measure the quality of students' software designs.
- Tools are often very difficult to install.
- Students get distracted by the complexity of tools.

For the development of WebUML we wanted to be able to address the points of discussion mentioned above, but also being able to use the tool as a research instrument. This means we want to measure and log every user action; something existing tools typically don't allow you to do. There were three main objectives for the development of WebUML.

First, in our research there is a growing need to upscale experiments in order to address a larger amount of student subjects to observe and being able to do statistical analysis.

Second, we wanted to have a system without the need of installing software.

Third, the tool should be equipped with a system that eventually can give students feedback on the quality of their model and guide them through the process of software design modelling.

We formulated the following research questions:

- RQ1: how can we gather data about student software design activities on a large scale in order to perform statistical analysis?
- RQ2: how can we offer a UML editor that is not complex (easy to understand) that serves a pedagogical context?
- RQ3: how can we provide students the appropriate feedback during their design activities?

In order to answer the research questions we proposed that WebUML should meet the following requirements:

- the ability of drawing basic class diagrams, including packages.

- the ability of giving feedback during the making of, and after completion of a software design assignment.
- the ability of uploading assignment solutions.
- the ability of logging user actions in a comma separated file.
- The tool should be able to import and export files for exchanging diagrams with other tools.
- contains a common graphical interface.
- should be web-based and support off-line use.

Because we mainly focus on structured software design in our research, we choose to only implement class diagrams.

The remainder of this chapter is organised as follows: first we discuss related work in Section 4.2, second we present an overview of WebUML in section 4.3. We discuss WebUML in Section 4.5. At last we conclude and discuss future extensions in 4.6.

4.2 Related Work

Several researchers have proposed specific educational tools for software design. In this section we discuss the ones we explored during the research of this dissertation. In this section we present industrial tools and tools that were developed during research on educational tools. The tools listed in this section are distilled from related work and from discussions about the use of tools in education, in particular in a workshop session at the Educators' Symposium in 2013 [2].

4.2.1 Industrial UML tools

Most universities use industrial UML tools in teaching, this is a list of often used (UML)modelling tools:

- Visual Paradigm - Visual paradigm is a broad development tool that supports process modelling, ERD modelling and UML modelling. It also has project management features and users can build wireframes and story boards for previewing ideas for future systems. To explore possible paths in a model it has an option for animation. (source: https://www.visual-paradigm.com)

- IBM Rational Rhapsody - Rational Rhapsody supports SysML and UML and domain specific languages (DSL). It can generate code. From model validation there is the option to animate models. (source: https://www.ibm.com/support/knowledgecenter/SSB2MU/rhapsody_family_welcome.html)
- Bridgepoint - Bridgepoint is a UML editor that is capable of executing models and contains a set of model compilers (providing translation). It should support the model driven development approach. (source: https://xtuml.org)
- Magic Draw UML - Magic Draw claims to be 'Ease of use' and to have a short learning period. It supports code generation. (source: https://www.nomagic.com/products/magicdraw)
- StarUML - It started as an open source tool (StarUML 1). StarUML 2 supports UML and ERD modelling. It generates code. It is now a commercial tool. (source: https://staruml.io/)

4.2.2 Research and Educational UML tools

Because of our in interest in research tools we compare the tools listed below in Table 4.1. Below we describe the considered tools shortly:

- QuickUML, by Crahen et al. [29] offers a small and simple subset of UML. The authors claim most UML tools overwhelm students that are just learning UML for the first time. According to the authors, the Java based tool supports iterative design because of the possibility to switch between design- and source code view.
- CoLeMo, by Chen et al., is a collaborative learning environment for UML [24]. CoLeMo uses the concept of pedagogical agents [59], autonomous software components that support learning. The agents are capable of providing the student feedback on their collaboration process and advice them on UML modelling. All activities and chat messages performed by the student are logged by the tool.
- Ramollari et al. present StudentUML [109]. Their tool is tailored for education with focus on simplicity, correctness and consistency. The authors claim the tool supports the process the students go through by providing feedback on diagram checks and by offering automatic repairs.
- Baghaei et al. present COLLECT-UML [11] an educational tool for learning UML class diagrams that supports collaborative and individual learning. COLLECT-UML has a submission system that is evaluated by a pedagogical module. The module provides feedback about the submission on the syntax level as well as on the semantic level.

- Soler et al. [124] propose a web based UML class diagram editor as part of their ACME e-learning framework. The tool can check a student's solution and provides feedback messages with suggestions for improvement.

- Hiya et al. present Cloocal, a web based tool for domain specific modelling [55]. It offers a model compiler for generating source code.

- Ma et al. developed ASE [84], a web based modelling tool designed for touch screen devices. It is developed to be cross platform. With the growing application of web browsers as application platform the authors saw the need to develop a modelling tool that is web based. The tool handles different touch commands and specific gestures which uses a whole new human-machine interaction mode.

- Lethbridge's Umple [78] is a tool that supports programming and UML modelling in parallel. It offers feedback to users to guide them away from errors and let students correct them.

- Jolak et al. present their tool OctoUML [60]. The tool is capable to recognise hand drawn sketches and turns them into formal UML notation. Users can mix formal and informal notation at the same time on the same canvas. It can handle multiple users on a single input device which supports collaborative modelling.

Tool	Web based	Desktop	Feedback	Collab.	Logging	Active
QuickUML	no	yes	no	no	no	unknown
CoLeMo	no	yes	no	yes	yes	unknown
StudentUML	no	yes	yes	no	no	unknown
COLLECT UML	no	yes	yes	yes	yes	unknown
ACME	yes	yes	yes	no	no	unknown
Cloca	yes	yes	no	no	no	yes
ASE	yes	no	no	no	no	unknown
Umple	yes	yes	yes	no	no	yes
OctoUML	no	yes	no	yes	yes	yes
WebUML	yes	yes	yes	no	yes	yes

Table 4.1: *overview of UML editors for educational and/or research purposes*

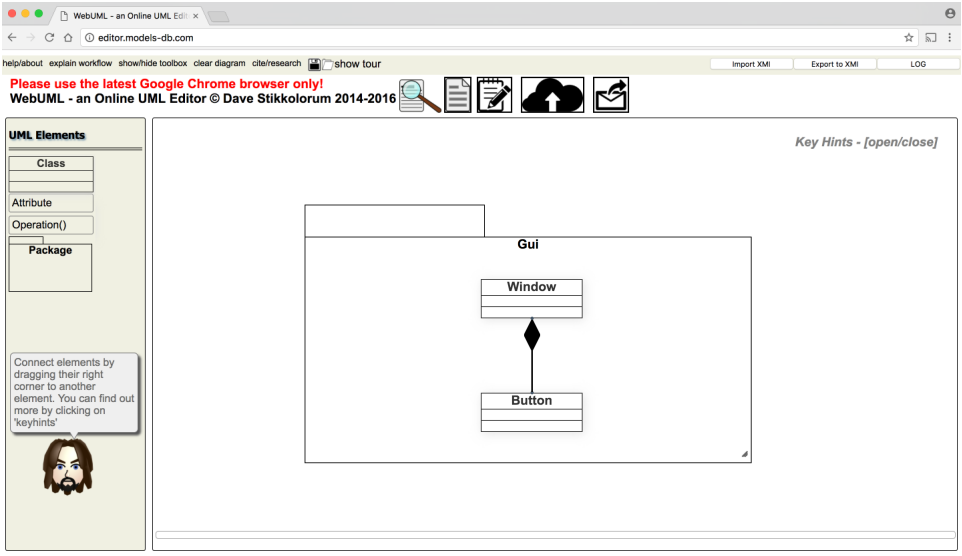

Figure 4.1: *WebUML in action*

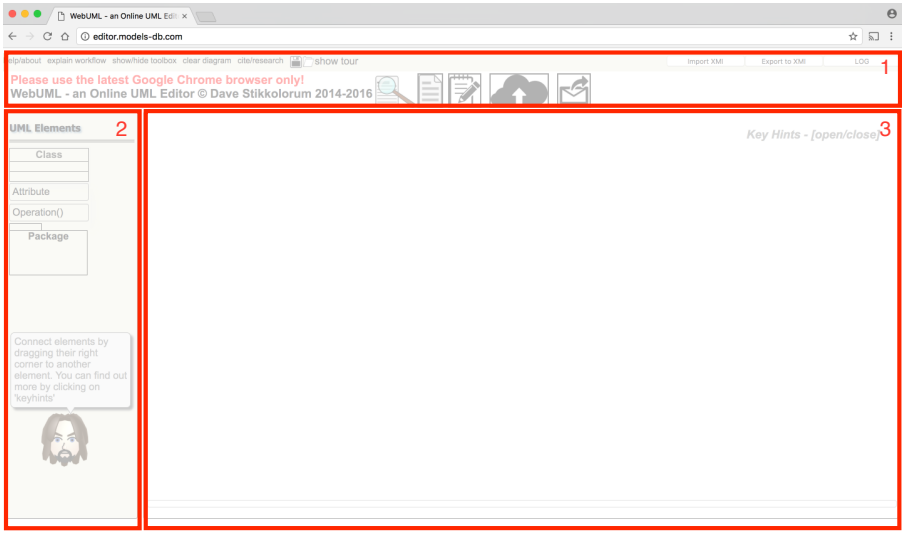

Figure 4.2: *WebUML's divisions*

4.3 Overview of WebUML

In this section we present an overview of the WebUML class diagram editor. We explain which functions we included to fulfil our research and education needs.

Figure 4.1 shows a screenshot of WebUML during a class diagram modelling session. Basically WebUML can be divided into 4 area's (shown in Figure 4.2): 1) navigation and control, 2) the toolbox, 3) feedback agent and 4) the drawing canvas. This section discusses WebUML's most important features organised per area.

4.3.1 Area 1: Main Control and Navigation

Area 1 is meant for the main navigation of the application and houses launcher icons for different kinds of services. We discuss the most important ones.

- *Import and Export XMI Button*: WebUML is capable of importing class diagrams that are described in the XMI[1] format. Through a dialogue it is possible to paste XMI code and import it into the editor. The canvas (area 4) will show a class diagram of the imported code. When exporting to XMI, WebUML downloads the current diagram to the users computer packed in a XML file [2].
- *Export Log Button*: Users' modelling activities are logged during a session with WebUML. It is possible to download this log as a comma separated file. For further reading about the logging system, see 4.3.5.
- *Evaluate Button*: This button activates the feedback agent. The result of the agent is displayed in the text balloon of the avatar (area 3) on the left bottom of the screen.
- *Mode Button*: Switches between reading an assignment text and modelling.
- *Register Question Button* Users' questions during modelling can be recorded by clicking a button. The remarks and/or questions will be added to the log.
- *Upload Assignment Button*: When users use the upload possibility, a zip file is generated and uploaded to a server. For own use the same zip file is downloaded to the users computer.

[1] The xmi specification is found at: https://www.omg.org/spec/XMI
[2] XMI is a specific application of XML

4.3.2 Area 2: Toolbox and Feedback Avatar

Area 2 consists of the standard UML elements Class, Attribute, Operation and Package. They can be dragged onto the canvas (area 4). We choose to draw associations between classes by drawing lines directly from the dropped classes instead of offering a association icon in the toolbox.

4.3.3 Area 3: Feedback Agent

The avatar that represents the feedback agent can be used to communicate to the user. It has a text balloon that can be filled with messages for giving hints or evaluation feedback. Chapter 10 discusses the feedback agent.

4.3.4 Area 4: Drawing Canvas

Area 4 is the drawing canvas. All dropped elements appear here. The user is able to move UML elements, associate classes and, change the types of the associations, adding association names and adding multiplicity elements. Element names can be edited and elements can be deleted.

4.3.5 General Services

WebUML provides two core services: logging user actions and providing (textual) feedback. In this section we discuss both.

- *Logging of Students' Design Activities* WebUML is equipped with an automatic logging service. The service logs different kinds of activities of students during their design sessions. Currently we log the following activities for UML elements: creation and removal, movement of the element on the canvas and changing names. Other activities that are logged are: submission of assignment, switching from reading to modelling and vice versa. All log items are timestamped using Unix time [3] (also known as POSIX time or epoch time).

  An activity log can be exported to a comma separated file in order to import the data into applications that are capable of visualising the data or support statistical analysis.

[3] https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap04.html#tag_04_16

With the analysis of the logs we aim to gain knowledge about typical design strategies, measure design time, identify design decisions, measure corrections and discover unknown patterns of student behaviour.

- *Feedback and Diagram Evaluation* WebUML aims to develop to a powerful educational tools that is capable to guide students through the process of software design and is able to evaluate students' models. Currently WebUML is equipped with a feedback agent that is capable of evaluating a diagram based on an example solution. The feedback agent is inspired by the concept of pedagogical agents [123, 85]. Agents, often used in games, are autonomous entities that have different behaviour based on events from the environment. Pedagogical agents react on actions of learners and/or the state of their products - in our case, diagrams - with the aim to support their learning process.

4.4 WebUML in research activities

We used WebUML in several experiments and classroom research. In this section we briefly describe the aim of the research and the role of WebUML in this research. The different research topics are explained in more detail in the following chapters in this dissertation.

4.4.1 WebUML and LogViz

In [136] we used WebUML in combination with another tool: LogViz [107] in order to investigate different strategies that students use for solving software design tasks. With the help of visualising WebUML's log files we were able to discover the dominant breadth-first and depth-first strategies. More about this research can be read in Chapter 5

4.4.2 WebUML as online tool

In [137] we were able to scale up the classroom setting by conducting an online experiment with 120 student pairs. Because of the web-based character of WebUML we were able to address a larger group of students. More about this research can be read in Chapter 6

In this research we also enabled and explored the assignment submission feature of

WebUML and the possibility to ask questions via an online form. More about this research can be read in Chapter 6

4.4.3 Teaching Agile Modelling

In this research we use WebUML as tool for doing an incremental analysis and design assignment that would fit in an agile development process. We asked student pairs to reflect on each other's challenges. More about this research can be read in Chapter 7

4.4.4 Automatic Grading and Feedback

We explored the possibilities of automatic grading and giving feedback to the students while making a design. Preliminary results are discussed in Chapters 9 and 10.

4.5 Discussion

Based on the experience with the use of WebUML in different experimental settings we answer and discuss the research questions in this section.

4.5.1 RQ1: how can we gather data from student software design activities on a large scale in order to perform statistical analysis?

We were able to equip WebUML with a logging function that produces interchangeable log data. The data was analysed in two experiments and statistical analysis was possible. Different activities such as movement of UML elements or edits of names can be logged. We were able to recognise different strategies students use to deliver a software design. The online mode enable us to address and research larger groups easily.

4.5.2 RQ2: how can we offer a UML editor that is not complex (easy to understand) that serves a pedagogical context?

WebUML is less complex than the industrial software design tools. It contains a slimmed down amount of functions, focused on relevance to the learning task.

By choosing for a web-based architecture WebUML does not need to be installed on the operating system of the students' computer. The web-based approach also has the benefit students can use any operating system they want. WebUML aims to be cross-platform, although, at the moment of writing this dissertation Google's Chrome browser seems to best support our tool. Chrome can be installed on all popular desktop operating systems (Windows, MacOS, Linux).

4.5.3 RQ3: how can we provide students the appropriate feedback during their design activities?

We explored the possibility of using a module that is based on the concept of pedagogical agents. The first pilot study shows encouraging results and suggest further research on this topic (further explained in Chapter 10).

4.5.4 Limitations

Currently WebUML only serves the students to create class diagrams.

4.6 Conclusion and Future Work

In this chapter we presented our educational web-based UML class diagram editor. WebUML was created because of the need of a UML tool that could be used as an educational tool as well as a research tool. The web-based architecture combined with the logging system enabled larger scale classroom studies that could be analysed with statistical analyses (RQ1). To avoid the complexity of most industrial tools we implemented a tool that does not require installation and uses a small subset of UML (RQ2). We were able to use the tool in our research in order to collect data about student activities during class diagram analysis and design. In pilot studies we used the feedback system that was implemented in WebUML in order to investigate the value of providing guidance during modelling activities (RQ3).

At the moment of writing this dissertation WebUML2 (version 2) is being developed. WebUML2 is going to serve a wider range of UML diagrams, to start with sequence diagrams and use case diagrams. Also the following functions are being added and researched:

- Replay - the replay function enables the user to replay the construction of a

model. In this way a lecturer can discuss possible errors they made in previous steps. At the same time it helps with students' self-reflection as well. Students can continue their work from the point they made the mistakes.

- Collaborative modelling - this extension is meant for user to collaborate and work on a model together. The collaboration is threefold: i) the users can work on the model at the same time, the model synchronises on the peers' screen when a change is made ii) the users can chat together via a text chat window iii) a video chat window enables users to have a live conversation about their progress.
- Feedback agent - the improved version of the feedback agent is guiding the student through a software design task. The agent is going to be configurable. The students should be able to decide on what items they want to receive feedback.
- Assignment mode - students will have the possibility to work on assignments provided by WebUML. Assignments can be uploaded (which was already part of WebUML version 1), but also receive direct feedback on the quality of their work.

The above mentioned functions should be tested in experimental settings in the future. We aim to have university software design programs involved to understand learners needs even better.