Tallinn University

Institute of Informatics

Priit Tammets

VISUAL MODELLING LANGUAGE FOR DESCRIBING PEDAGOGICAL PATTERNS

Master thesis

Supervisor:

Hans Põldoja

Author: .... “........” 2008

Supervisor: .... “........” 2008

Head of the Institute….. “........” 2008

Tallinn 2008

Table of Contents

Introduction

Centre for Educational Technology (CET) in Tallinn University has more than seven years of experience in developing different learning environments and courseware platforms, for example VIKO[1], IVA[2], KooliPlone[3], Krihvel[4], OPAH[5] and LeMill[6]. In many cases there has been problem in terms of communication between developers, investors, community and suppliers, also known as stakeholders, who understand projects goals very differently.

OPAH was a collaborative project between two biggest universities in Estonia plus several other organisations and one of the outcomes was a platform for teachers' personal e-portfolio solution. E-portfolio or also known as electronic portfolio collects the user's digital materials and creations to a one place, which enables users to demonstrate their skills and competencies. There are some extra features added as well to support and develop teachers own skills like reflecting, which is supported through the blog, for example.

In that case it was clearly seen that changing information between two sides (developers and stakeholders) was not the easiest part. The common understanding of the situation may become difficult and bridging the communication gap may be turning out as the major obstacle in the end. This gap is actually nothing else than a language barrier, which can be easily deepen by the time, if there are so many different people around the actions.

Next bigger project in the CET will be runner up of the popular learning management system IVA, which first version was launched already in 2002. Definition for the LMS says: Learning Management System (LMS) is specialised Learning Technology Systems, based on the state-of-the-art Internet and WWW technologies in order to

---

[1] VIKO – http://www.htk.tlu.ee/viko

[2] IVA – http://www.htk.tlu.ee/iva

[3] KooliPlone – http://www.htk.tlu.ee/kooliplone

[4] Krihvel – http://krihvel.opetaja.ee

[5] OPAH – http://eportfoolio.opetaja.ee

[6] LeMill – http://lemill.net

provide education and training following the open and distance learning paradigm (Avgeriou et al., 2003). Since then learning in the use of the Internet has changed rapidly – people are blogging, shooting huge amounts of photos, doing mobile films and much more every day and continuously. Although giving up of using the central and closed systems is major step for many teachers and students, creating and sharing open content is becoming more and more popular.

According to those trends, IVA 2 must use all of those opportunities and nuances which are provided in the era of Web 2.0. Characteristics of the Web 2.0 assume that users are the most important part of the Internet and its environment and it is crucial to build applications and tools that harness network effects to get better the more people use them (O'Reilly, 2006). IVA 2 cannot be centralised system in the certain server in the certain university. Openness that should provide users to use their exciting blogs, photo sharing accounts and several other Internet tools is one of the basics that carry its idea.

IVA 2 will be definitely bigger project than its predecessor. The scope of the partners will be wide spreader and their background varies from pedagogical to technical. Communication issues that popped up already in OPAH project are becoming actual again and handling those needs a bit different approach. For the thesis and for the evaluation process, author is using already working example of the course where a wide range of Web 2.0 environments are used, as a base for the research. Reason is that the process of the IVA 2 has not reached to the point, where this outcome is usable and no real design process has been really done yet.

As mentioned earlier, so called communication cap is a result of different languages spoken by the different people with different backgrounds. Even if the native language is the same for them, their visions are something else. One solution for bridging the language barrier could be common visual language for providing wordless communication through the use of clear symbols and pre-defined definitions. But it is not meant to be a replacement for our native languages, this is meant to work as an extension for exciting methods.

Visual language for those purposes should be easily understandable to the wider group of users, who might not be familiar with computer science terms and do not understand programming languages logic and syntax. New solution should involve flexible, understandable and integrative approach that can be spreader as a foundation for basic conceptual design process in social media era. Using visual language for communication purposes between different people is not a new idea – already before the alphabet humans used symbols to communicate about their everyday lives. For humans information that can be understandable via eyes is always more reliable than compared with hear able. As Canadian media theoretic Marshall McLuhan has pointed out, most of the people is suspecting the ear, and are feeling more secure when things are visible (McLuhan & Fiore, 2005).

Requirements for the visual language, that we are trying to define and compile, will declare its usability in different examples, as well with examples of LMS or in terms of distributed learning environments, where higher responsibility for students own learning and using preferred Web 2.0 tools is needed. In addition, traditional semester-length classroom-based lectures are replaced with different methods (Bowman, 1999).

While designing and compiling the language needed for describing the pedagogical patterns, method for describing different expert knowledge of teaching and learning (Bergin), there should be remembered, that the usable language is meant to be used for users, not designers and an evaluation of the final product should involve people, who are able to give feedback far above the average computer user level. As Jakob Nielsen has pointed out (2008), designers and typical users might not be on the same level, because designers know too much, are too skilled and even care too much about their own handwork they've created.

Research questions:

- How can existing workflow languages be used for visualising pedagogical patterns?
- What adaptations are needed to workflow language in order to support analysing and designing distributed learning environments?
- What is a minimal visual vocabulary for describing distributed learning environments in concordance with activity theory?

For answering to those questions, following tasks should be taken and completed:

1. To analyse existing visual modelling languages
2. To develop vocabulary for new visual modelling language
3. To evaluate new visual modelling language

In this thesis, author is using action research method to analyse different graphical modelling languages; to choose one of them and to implementing it in a use of the course design model. As an author of the thesis is part of the same research group in the CET, background study and implementing has been done within the same group, therefore in this thesis, the term "we" has been used.

The thesis is divided into four chapters. In the first chapter different conceptual design issues has been described and some of the solutions has been provided. In the second chapter a visual modelling language has been defined and overview of the different modelling languages has been given. The third chapter is concentrating on compiling the new language according to the literature overview and specified requirements. In the fourth chapter evaluation and testing of the outcome has been done and analysed and some of the suggestions has been provided for the further development process.

1. Conceptual design issues

Every technical tool we are using constantly every day requires us to learn and remember how to use it efficiently enough to complete certain task with minimum time amount. Usability is all about solving the equation: how to help users (us) succeed within their tasks and achieve their activity goals and is defined through the several design goals (Brinck et al, 2001):

- Functionally correct, system should be able to complete task that is needed for the user
- Efficient to use, less time for an action means more efficient tool
- Easy to learn: system, that requires deep and rough learning for completing simple task is not user friendly
- Easy to remember: learned handling is easy to remember and tasks can be performed several times according to the same schema
- Error tolerant and error handling which requires system or tool to have opportunity to prevent or easily detected and corrected upcoming errors or fault messages
- Pleasing use of the system involves decent graphics and lay out, which plays important role in systems usability

Usability means that the people who use the product can do so quickly and easily to accomplish their own tasks (Dumas & Redish, 1999). As it reveals, user is the main target and without the real user it is impossible to achieve maximum usability.

1.1. Methodology of conceptual design

The design process that we are using for achieving our goal is similar to what Brinck, Gergle and Wood (2001) has described as Pervasive Usability Process. For our purposes we have modified it slightly, because we are not dealing with web sites and aim is to create a tool that can be used in the conceptual design stage for example:

Requirements analysis

Goals for the visual language

Requirements specification

Competitive analysis

Conceptual design

Sketching draft and abstract design of the visual elements

Task analysis to find critical features

Mock-ups and prototypes

Generating mock-ups – paper prototypes and a digital stencil is the final outcome of the product design

As seen here, there is no production and launch stages listed here which are out of this thesis scope and must be taken into the account in later research process. No user testing is involved also; evaluation has been done using the formative evaluation method, which is described in chapter 4.

1.2. Conceptual design of distributed virtual learning environments

Designing or planning a course for environment which is mainly using distributed learning environment can be challenging for a person who is new in social media context. Other situation we may find a teacher, who has been interested in blogs and wikis for a long time but has not been in the design process of course design, which is using those tools for an educational purposes. One possible way to bring those skills closer to the end-users is through the participatory design, like developers did in the LeMill project design process (Leinonen *et al*, 2008).

Participatory design process was divided into sessions, which were combined of one researcher / designer and 2 – 3 teachers. During the 2 – 3 hours sessions, teachers had to cover 4 different steps of the process, which were preparation and "ice breaking", talking about the scenarios, gathering the specific data and finally making conclusions and wrapping up the session. The second part, scenarios, was prepared by the designer, who wrote the example scenarios and teachers tasks were to answer for the several questions, like: "Is this possible story?" or "could you imagine yourself to the role of the person in the scenario?". Answers were collected by the designer and steps were repeated as long as there were example scenarios left.

Same important step is the third one, which should encourage users to think about the systems functionalities and explain their thoughts through the sketches and drawings on the same time.

1.3. Communication conflicts in conceptual design process

LeMill project's design session, which was described above, can be named as a quite simple case in terms of communication and changing information between participants in the group work. Only one person, who had a technical background (designer) plus small amount of teachers, who formed a test team cannot come up with too many difficulties. One of the issues that may occur during the test between international team members is language barrier and misunderstandings, which is natural, when users' mother tongues differs from each other. Although, LeMill project solved this issue by forming groups by nationalities and the tasks were performed in users' native language.

One of the possibilities to avoid misunderstandings is to formulate groups between test users with the same language background, which also means, that depending on their nationality and characteristics, results are filled with all kind of information and proposals or users are sitting quietly and suggestions are more formal and are not going very deeply into the main issues.

Our proposal for solving those overcoming language conflicts in design process is common visual language that can be used in combination of scenarios and user testing. User, in LeMill's example teachers, should be able to describe and to share their thoughts between each other without depending on language barrier or different technical background they might have. Thus, it cannot be fully modelling language used in programming or modelling the mathematical models and should be able to carry a message as easy as the spoken language with a use of visual symbols.

2. Existing visual modelling languages

Visual language and visual communication can be defined in many ways and context. Creating an image for communication purposes has been used for centuries and it carries a traditional method for socialising between different countries and nations. Images for that purposes are often used as words or phrases not as alphabet, which means that through the unwritten connections and meanings it carries message between strangers without losing its core. Therefore, it is very common to use visual language for supporting the spoken language and vice versa, which makes communication even more efficient.

Visual language is strongly relied on visual thinking, and which is again related to out brain functionalities. As we know, our brain is divided into two hemispheres, left and the right hemisphere, which functions differs a bit (Davidmann, 1998):

Left hemisphere – communicates through the words, is dealing with logical and systematically issues and has advanced verbal abilities.

Right hemisphere – communicates through the images, is intuitive and imaginative, and deals with emotions and feelings.

Those two hemispheres are always working together and are useless in case of them is not functioning. Still, different people have different abilities in logical thinking and being imaginative – their brains have being developed differently. Although it is being discovered, that the right hemisphere is actually the older part of the brain, which leads to the reason, why thinking in pictures is so fast.

Nowadays is visual language associated in people’s minds mostly with artists, who are speaking with an audience through the artworks and installations although there is no common vocabulary between those works. That means in the absence of a theory of visual grammar, neither a speaker nor artist can decide what belongs to a correct use of visual language (Saint-Martin. 1990).

Visual language in terms of software development

As in other terms, software development process involves developers with all kinds of backgrounds. Verbalising their thinking is one of the best known methods for corporate working on the level, where spoken language can be difficult to understand for some parts of the team. For that purposes it is needed to have and use more formalised method than mind map, for example – structure of the development process or product must be clear and visualises the overall process clearly for everyone. Different modelling language has been used for a long time to draw via diagrams how processes and actions are taking place inside of the certain activity. They provide the means to represent knowledge structures as graphs of labelled nodes and arcs (Gaines, 1991). As follows, we are introducing the most known visual modelling languages from variety of areas, some of them are mainly used by programmers, others are simple enough to be used with kids, but the common line of them provides information exchange via graphical elements instead of written text.

2.1. Concept maps & mind maps

One of the best known methods for creating a rough sketch out of the ideas that pop in to the head is drawing simple concept or mind map. Very often we do not make any differences between those two types and combine their principles with each other, which actually do not make any harm, because purpose of using those are basically the same – to support knowledge representation and idea generations from the bottom up level.

Both the mind map and the concept map are beginning from the one central idea, starting point or the topic, which leads further ideas and discussion until the overall picture is forming a whole. Rapidly drawn images are usable for fast communicating or memorising but they are not usable for software development on the deeper level. Lack of formality and different visualisation methods can narrow down the ways of using them in many fields.

Concept maps

Concept maps are graphical tools for organising and representing knowledge. They include concepts, usually enclosed in circles or boxes of some type, and relationships between concepts indicated by a connecting line linking two concepts. For specifying the relationship between two concepts, words and labels on the connectors are being used. Concepts are usually showed hierarchically and visualisations begin from the top and taking directions to the bottom. This structure depends on the context on which knowledge is being considered and therefore focus questions are being used (Novak & Cañas, 2008). One important characteristic that makes different concept map from the mind map is crossing links or relationships between the concepts.

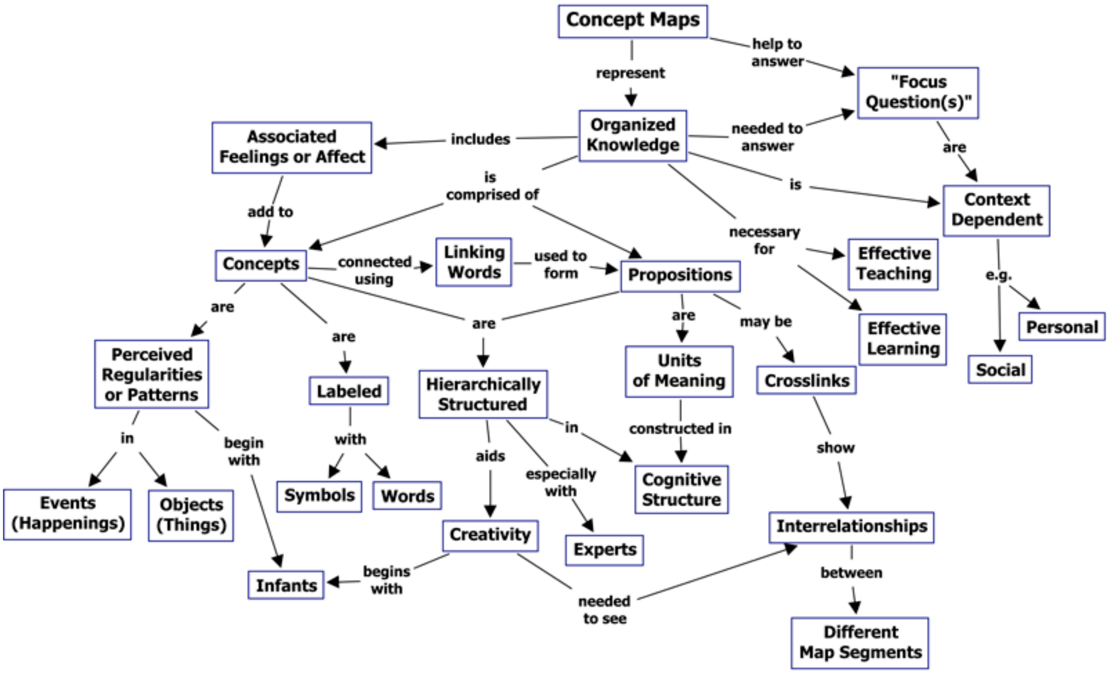


Figure 1: Example of concept map (Novak & Cañas, 2008)

History of the concept maps goes back to the 1972 when they were first used in Novak's research program which was studying children's understandings of science concepts. Program was based on David Ausabel learning psychology, which claims that learning takes place by combining new concepts and findings into prior knowledge. For better representation of children's conceptual understanding emerged the approach of representing children's knowledge in the form of a concept map, which was soon ready to be used in many other research areas and uses (Novak & Cañas, 2008).

As mind maps, concept maps are widely being used for pedagogical purposes and in terms of supporting learning. For example using concept map as an evaluation tool to encourage learners to analyse their actions in the learning process or to identify valid or invalid ideas and to be discussed those later on.

Mind maps – Similarly to concept map, mind map is a graphical tool for representing info, but instead of important relationships between concepts, it is based on radial hierarchies and freely flowing tree structures. Main topics and classes are connected with branches that are starting from the central image. Branches are marked with keyword or graphical image and similar branches are "growing" out from the same source (Budd, 2003). Mind maps are not usually using full sentences for describing its content – single key words should help user to see the entire maps at once and to create more associations.

Mind maps are widely being used in many fields, including education, to encourage a brainstorming among individuals or bigger groups. Mentioned single key word approach is one possibility to collect varying people's thoughts to develop more and more deeply going ideas. For example, Budd (2003) has described different ways to use mind maps in education - for group mind mapping (brainstorming), mind mapping for creativity and even for mapping the reflection on the specific course. Before using the mind map with students, he introduced mind map, its concept and illustrative uses, which helped class to understand the exercise core idea. For the next step, students were divided into groups instead of individual work to encourage deeper analysis of the topic through the group brainstorming. It is also being mentioned, how important is to provide help and feedback from the teacher during the session to help students, who are stuck with their ideas. That kind of personal approach and direct interaction gives teacher an opportunity for getting more close contact to the student through the rapid brainstorming activity.

Of course, mind mapping is not universal activity and several people do not feel comfortable using this method in any way. Reasons vary from lack of formality to the hardly understandable visual concept of the maps. Although mind maps have certain characteristics, it does not require certain methods for using it and there is no universal activity to be followed. On the other hand method is very simple and usable everywhere without deep preparations or hand books, maps can be draw by hands and represented on paper.

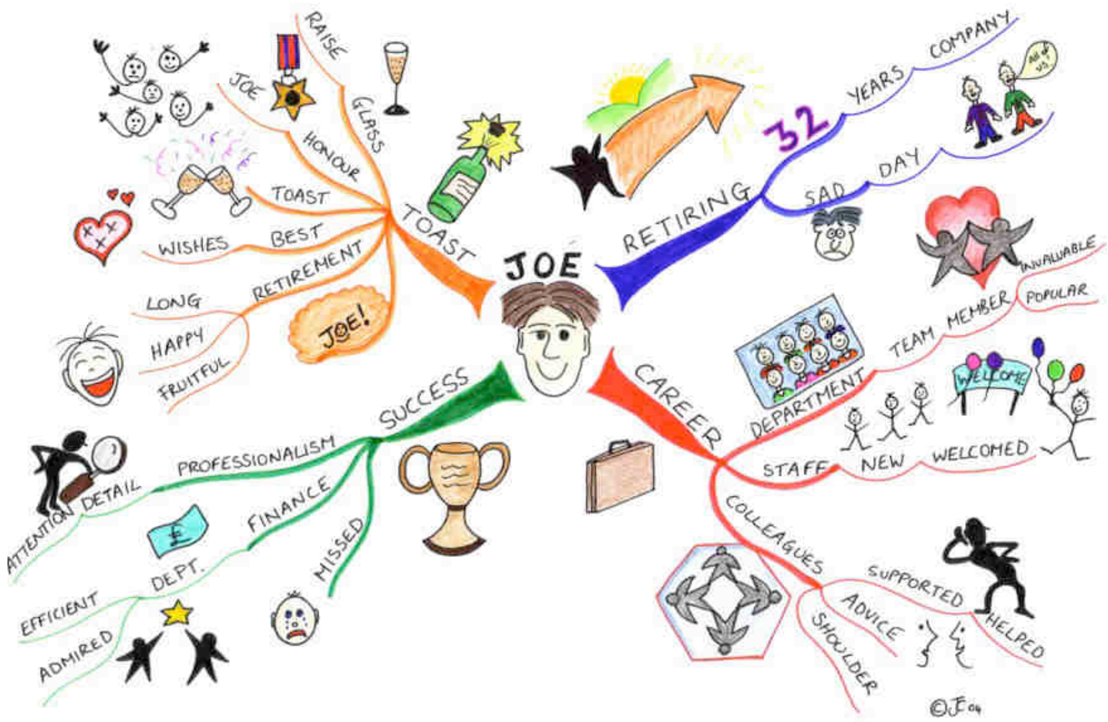


Figure 2: Using your own colour schema and images will help better to understand the overall picture (Cormie, 2007)

2.2. Flowcharts

According to IBM Flowcharting Techniques manual (IBM, 1969), a flowchart is a diagram that shows the operation performed in an information processing system and the sequence in which the operation are performed. Flowchart symbols are used to represent the operations and sequence of operations. Originally is flowcharting technique dating back to the 1921, when Frank Gilbreth first introduced method for documenting the process flow, which was called flow process flow after what this technology was widely used in industrial engineering. Flowcharts, what we are recognising nowadays are dating from 1946-1947 by Herman Goldstine and John von Neumann from Princeton University. The purpose of the flow diagram, as they called it, was to give a picture of the motion of the control organ as it moves through the memory picking up and executing the instructions it find there and to show the states of the variables at various key points in the course of the computation.

Similarly to mind mapping and concept mapping, flowcharts or process maps are often used by non-programmers for exchanging information with each other. But flowcharts are often being misunderstood because lack on uniformity in the meaning, that are created by the users and use of very specific symbols in a way they should not. For preventing those situations, International Organisation for Standardisation has documented the symbols and conventions of the correct use of the flowcharts (IBM, 1969).

For most of the flowcharts you will need only few most common types of flowchart shapes – rounded box that represents an automatically occurring event; rectangle that is representing an event which is controlled by the process and can be named as a most frequently used symbol in flowcharts; diamond that is representing decision point in the process, typically are requiring the "yes" or "no" response and a circle that is representing the point where the flowchart connects with another process (Cohen, 1998). A flow line is used for connecting the different elements to each other. Specific flowcharts are requiring also some special diagram types for declaring their purposes and functions.

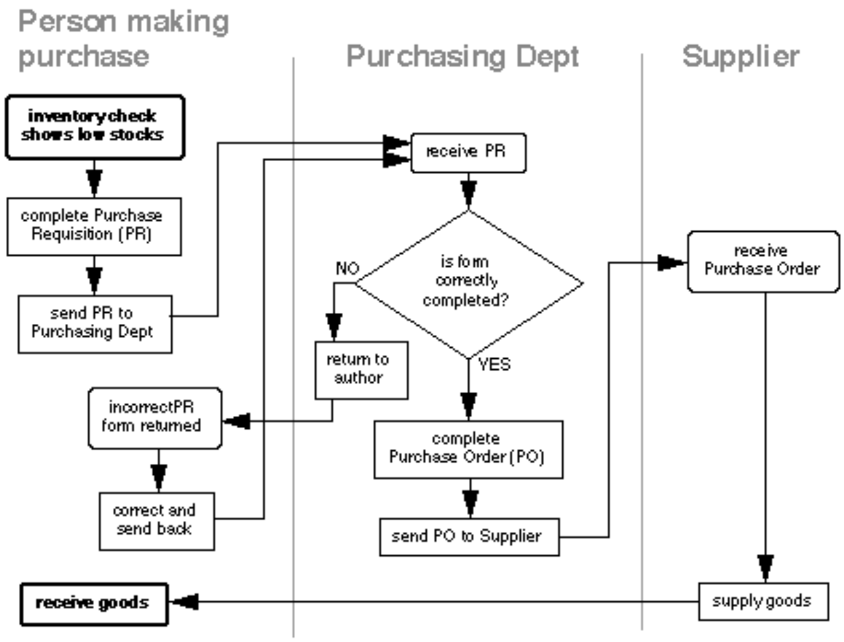


Figure 3: Use of "handovers" in a flowchart (Cohen, 1998)

Similarly to the Activity diagram in the UML (Unified Modelling Language), that is being described in the next chapter flowcharts can be used for describing the processes with a number or different participants or roles. This technique is called "handover", and can be used for tracking various processes done by different people. For better visualisation, flowchart is being divided into as many columns as there are different roles (Cohen, 1998). An example flowchart is seen on Figure 3. Our goal has user role requirement also described in the requirements and this approach is highly usable for our example as well.

2.3. Object modelling languages, UML & UML2

Unified Modelling Language (UML) is one of the most known languages for displaying and drawing different software diagrams. UML is the international standard notation for object-oriented analysis and design. It is defined by the Object Management Group[7] (Graham & Wills, 2007).

UML2 uses 13 types of different diagrams, which are categorized as follows:

Structure diagrams:

- Class diagram
- Component diagram
- Composite structure diagram
- Deployment diagram
- Object diagram
- Package diagram

Behavior diagrams:

- Activity diagram
- State Machine diagram
- Use case diagram

Interaction diagrams:

- Communication diagram
- Interaction overview diagram
- Sequence diagram
- Timing diagram

As it can be seen, this listing of diagram types is quite massive if compared with other visual modelling languages. Clearly most of the listed types are not suitable for our purposes and cannot be adopted for different reasons. For example class diagram is

---

[7] www.omg.org

used for visualising the static system structure and component diagram displays how system is divided into smaller components and how those components are related to each other.

The most relevant diagram model, that will suite for our purpose is activity diagram. Activity diagram shows reliance between activities as moving from starting point to the goal. Similarly to flow charts and Petri nets it is used to model program flow or user activity (Graham & Wills, 2007).

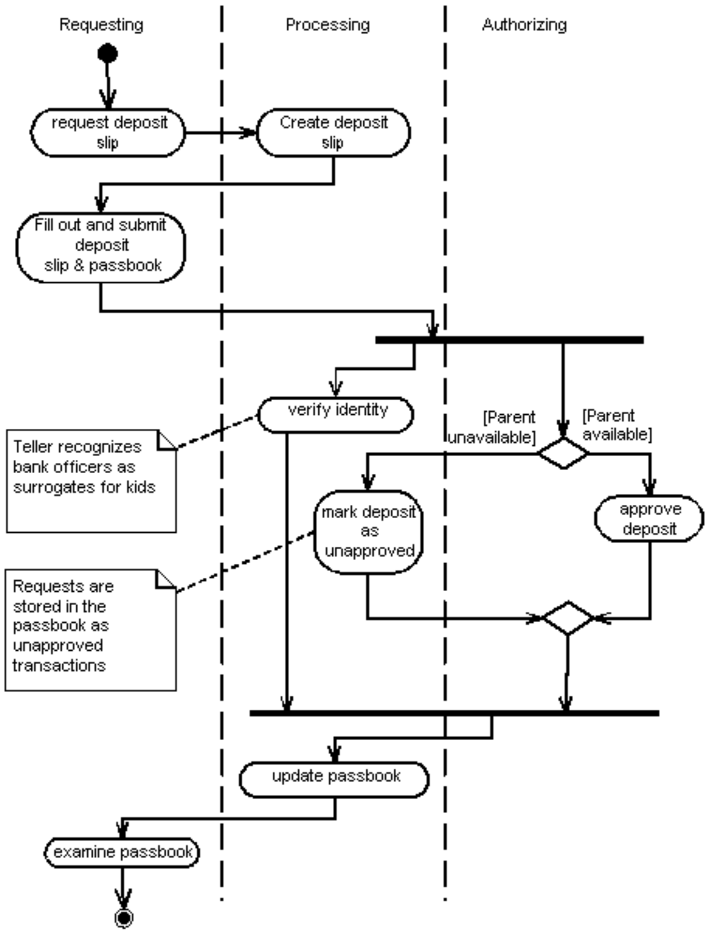


Figure 4: Example UML activity diagram (Holub, 2005)

Filled circle in the diagram stands for the starting point and a circle with a smaller dot inside of it marks the end point of the action. Rounded boxes in the diagram are standing for the actions and arrows indicate that action must be completed before heading towards next one. Thicker black lines in the middle of the example are (also known as forks and joins) visualising the certain set of parallel actions. Each action has given to a certain role, which name is written into the action inside of the brackets. Those partitions are used to identify actors, departments, systems or objects.

One of the minuses UML has is that its presentation conventions mean that some lines are thicker than others and that some characters are in bold or italics. Some of these conventions are difficult to accomplish when drawing by hand (on a piece of paper or on a whiteboard, for example), however only the italicised text is really important, so the other conventions can be ignored when hand-drawing (O'Docherty, 2005).

2.4. Mathematical modelling languages, Petri nets

Petri nets is a state based modelling language for mathematical representations of discrete distributed systems. Language was formulated in 1962 by German mathematician and computer scientist Carl Adam Petri. With his mathematical background involved, he worked out the graphical language, which has strong mathematical background, which makes processes visualised in it to be completely formalised (Waardenburg & van Emmerik, 2004). Although a Petri net is having strong mathematical background, it has been used for modelling and analysing business process models. Its theory makes possible to use several analysis techniques and tools to resolve and analyse the precision of workflow structure.

The original Petri net graph is consisting of two node types, places and transitions. Places are visualised with circles and transitions by rectangles. The nodes are connected via directed arcs but connections between two nodes of the same type cannot be used. The classical Petri net can be used for modelling of states, conditions, events, synchronisation, parallelism, iteration and choice but does not allow for the modelling of data and time. For solving this issue, some of the extensions have been proposed to the Petri net model (van der Aalst, 1998):

- The extension with colour to model data.
- The extension with time.
- The extension with hierarchy to structure large models.

A Petri net extended with colour, time, and hierarchy is called a high-level Petri net.

Examples of using the Petri nets in education can be found several and even despite of the fact that this tool is intended to use for mathematical representations. SeungSoon and YoungIn (2008) have applied Petri nets to model e-learning components and activities.

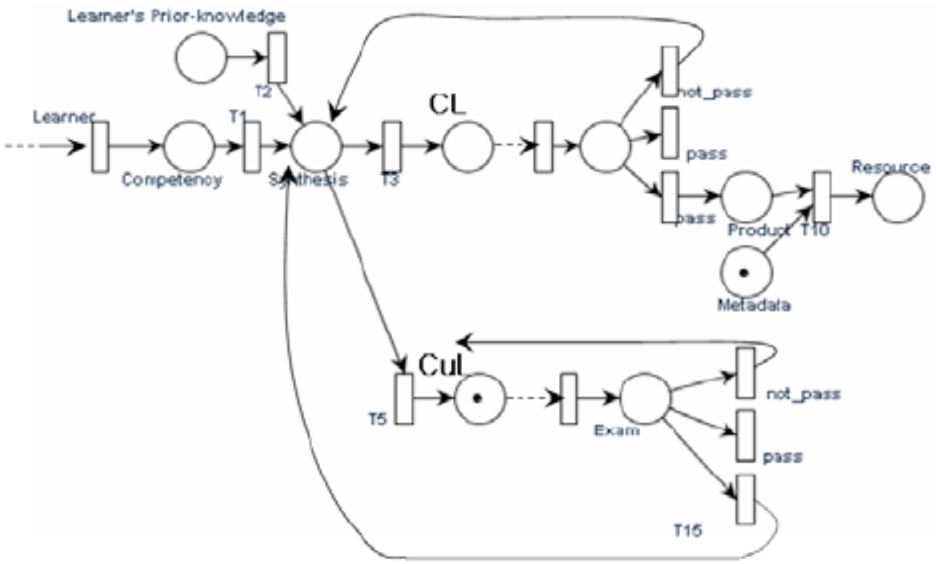


Figure 5: Modelling the e-learning in Petri nets (SeungSoon & YoungIn, 2008)

Authors are describing effective and individual customised learning which is considering individual grade with co-operative learning which creates knowledge for their needs with involving at the education positively. Visualisation has been modelled by using Petri Net. In their case, Petri nets was just a tool for representing the examples and visualisation of the different learning methods: „ ... knowledge to substitute for real experiences can be accumulated through positive interaction at the community which helps learner serve positive, creative and co-operative learning to reorganise and originate the knowledge which they need by involving at learning program positively”. But they also believe that it’s not possible for people to have direct studying for appearances at many fields, the appearances can be studied through examples and these examples can be modelled through a modelling language tool like Petri Net.

2.5. Process modelling languages, newYAWL

newYAWL is a modification of the popular process description language YAWL (Yet Another Workflow Language), which is originally based on mathematical representation language Petri nets, as it requires a formal foundation and graphically clear representation to visualise different models (Waardenburg & van Emmerik, 2004). Basically YAWL is an extension to Petri nets, which can be used for modelling workflow processes and patterns to some extent and YAWL should be manage further from that point. YAWL is developed at QUT's BPM research group in collaboration with Eindhoven University of Technology.

All the YAWL elements have been retained and are having the same functions in the newYAWL as well. New elements have been added according to the new workflow patterns that need to be described.

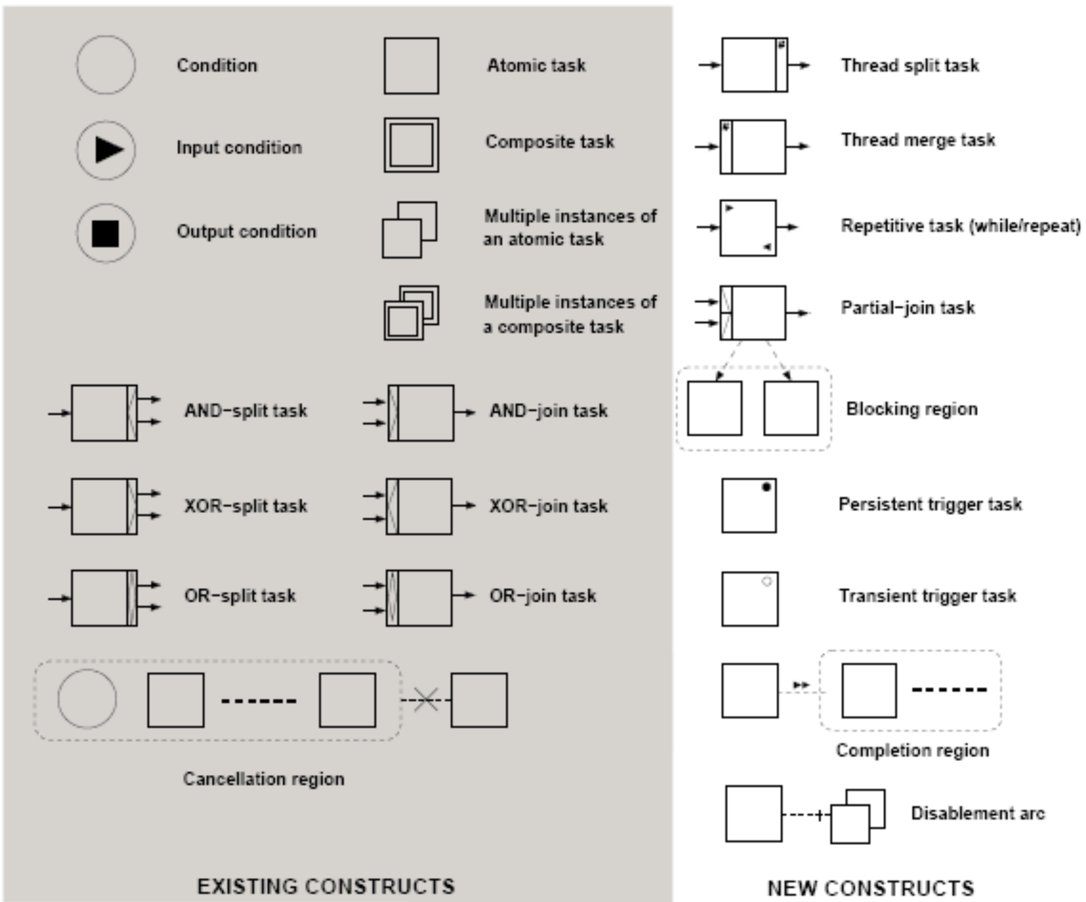


Figure 6: Elements of the newYAWL (Russell et al., 2005)

newYAWL is a business process modelling language to support the visualising the control-flow, data and resource perspectives to be captured in a level they can be stipulated (Waardenburg & van Emmerik, 2004). It has not been used deeply for visualising the pedagogical activities which makes it tempting to test its possibilities in that field.

2.6. Other modelling languages, Garrett's visual vocabulary

While we are talking about vocabulary, not only written words and metaphors are not those we are interested in – visual representation will guide us through the new languages much faster. On 2000 Jesse James Garrett released first version of his Visual Vocabulary. It is basically a set of symbols for describing system, structure or process (Garrett, 2002). Garrett realised that only architecture diagram couldn't document conceptual relationships: which categories go together and which remain separate; how do the steps in a given interaction sequence fit together (Garrett, 2003)? Garrett went even deeper: he divided his vocabulary into two parts, simple (macrostructure) and complex (microstructure), to ensure that his vocabulary could be usable both on projects clients and programmers.

Garrett's visual vocabulary is based on conceptual model that defines:

- The system presents the user with paths.
- The user moves along paths through actions.
- The action causes the system to generate results.

Vocabulary is consisting of 17 different elements, which are divided into smaller elements groups, for example relationship elements, commonalities, reusable components etc. Example of the use is seen on the figure 6, where the simplest and the most common element is a page (rectangle), few decision points (diamonds) and conditional selectors (triangle without the top) are also presented. Elements are connected with different relationships (arrows).

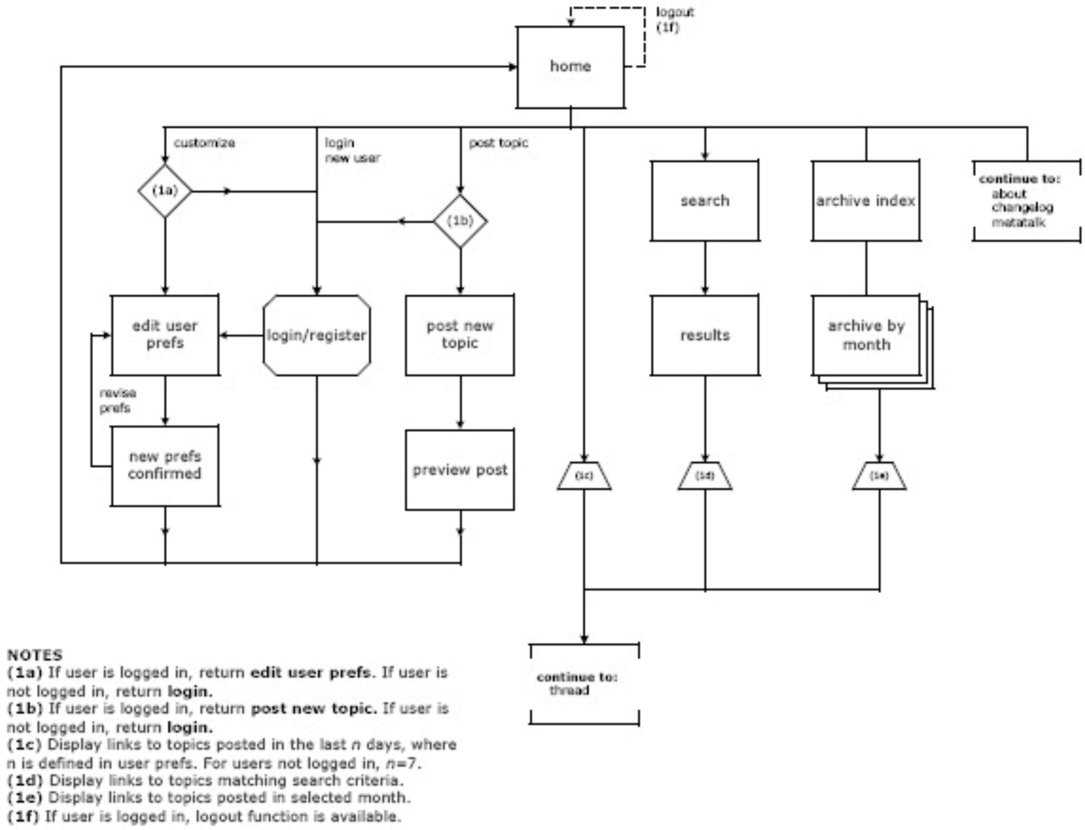


Figure 7: Information architecture and interaction design of MetaFilter.com (Garrett, 2002)

The Garret's example cannot be used as a tool for visualising the pedagogical patterns in a way, we are intending to do. It is used for describing the information architecture and interaction design in terms of web pages. But there are some ideas that we can implement in a bit different are than visualising the patterns.

In parallel with elements creation, Garrett has defined the requirements for the elements in a very handy and clear way (Garret, 2002). Some of them can be applied for our purposes as well:

- Whiteboard compatible – the elements should be simple enough that the diagrams can be sketched fast by hand on a paper or on a whiteboard.
- Tool-independent – the elements should be available for wide range of diagramming and drawing tools, so that user can choose the most comfortable one.
- Small and self-contained – usable for variety of users, from project sponsors to programmers.

2.7. Conclusion

After the analyse and comparison of the different modelling languages it can be said, that almost every listed language can be used as a starting point for designing suitable solution. There is no precise and fully compatible vocabulary and every language has its own strengths and weaknesses. As our research has started almost from the scratch, it can be started by choosing the most relevant version and to be continued by developing our solution on chosen language.

Mind maps and concept maps were not been chosen because lack of formality. Mind map's radial structure makes it hard to draw and activity pattern whereas concept map do not offer different views for representing the moving between the actions or tasks.

Flowcharts "handover" technique, similarly to the UML Activity diagram's swimlane, is one element that we are going to use in our visual representation. One more extra feature, we are going to use, is persona or user, which concept is being used in different UML diagrams.

Petri nets is formal and strict but on its visual side and elements lacks of eye catching structure and the documentation for it is usually hardly understandable for non-mathematicians.

According to those and other smaller reasons, we are going to use newYAWL as a base to accomplish our tasks:

- newYAWL is strongly related to Coloured Petri nets, its formalisation is based on it, which means its semantics could be used for analysing the pedagogical patterns on deeper level.
- Understandable documentation, which provides opportunity for further research and development.
- Clear and easily to remember symbols that will help us to design and develop intuitive visual language.

3. Compiling new visual language

A new solution should involve a language, which should be as natural as possible to give future users some kind of guarantee that they do not need to learn another huge set of different icons and definitions. Fast understanding of the language visual side and semantics will help users to give a fast start towards implementing it in real cases. According to discussion in our research group we defined a list of requirements that should be implied in final solution:

Clear symbols – set of symbols must be clearly understandable to very wide group of users, without depending on their background in computer skills or spoken language.

Easy to draw by hand – target group for our outcome are users, who have been missing clear method, which could help them to formulate their ideas of the course. Usage of the symbols must allow them to use it even in situations where no technical tools except whiteboard and a marker is nearby. Simplicity of the visualising the course design is one of the key points to be followed and practised.

Can be easily remembered – as strongly related to the previous statement, symbols and the semantics of the visual language should be intuitive and memorisable. Clear table of with the symbols explanations is needed but not necessary to carry it to the classroom or stationary on users' desk.

Usable for describing different pedagogical patterns - in terms of closed (WebCT etc.) and distributed systems.

Expandable – although the elements and the basis of the modified result is borrowed from the newYAWL, we cannot assume, that our research will reveal the most appropriate result. Therefore we encourage users to expand the offered language with their thoughts and even their own symbols and other visual elements or semantics.

3.1. Pedagogical patterns

One of the aims of this research is to wind an appropriate way to find and implement visual modelling language in use of the different pedagogical patterns. All the activities in our everyday life are consisting of activities and actions that are repeating their self once in a while. We are used to freshen ourselves in the morning same way like yesterday, we are drinking coffee sitting on the same chair and using the same amount of sandwiches. The same effect can be noticed in the educational activities as well, like Eckstein and Voelter (2001) are pointing out – for teaching something particular and specific, specialists of their field are asked to read the lecture for the students. Negative side of this action is that often those specialists in their own field lacks of pedagogical background, which will ensure their skills as a teacher or lecturer. On the other hand teachers who are having years of experiences in schools, do have enough knowledge to share it with others too. Their teaching methods have been developed in years and their repeating characteristics which are forming to certain patterns are worth for sharing it. Those patterns are called pedagogical patterns that are designed to capture best practice in a specific domain and expert knowledge of the practice of teaching and learning. Presenting the information in an accessible form makes it easily usable for bigger teaching community (Bergin).

The idea of the patterns is originally part of the architecture, when in 1977 architect Christopher Alexander defined patterns: Each pattern describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice (Eckstein and Voelter, 2001).

The most known usage of the patterns in the education is the project called Pedagogical Patterns Project[8], which aims to attempt to capture the knowledge of experienced teachers in a way that can be used by novice teachers (Eckstein and Voelter, 2001). Project is also working on a pattern language, which is consisting of set of patterns that work together to generate complex behaviour and complex artefacts, while each pattern within the language is itself simple. In this paper we are not trying to compile a pattern language, because the visual languages we are describing here cannot be used for that purposes. Pattern language does not have only one correct path to be followed, like for example Flowchart has. But the example languages and the modified one might be possible to be used for describing the different pedagogical patterns.

---

[8] http://www.pedagogicalpatterns.org/

3.2. Visual representation of the new language

Although newYAWL provides a decent set of language elements to be used for different models, it cannot be used directly and unmodified easily in terms of pedagogy and Web 2.0. Secondly it should be compatible with Activity theory, which provides the methods for describing the community side and the individual work patterns.

Element	Type	Description
Event	Object	Fixed point on the timeline, may contain a description, value 0
User	Subject	Single user, teacher, student, facilitator etc. Can be part of the group
Group	Subject	A set of individual users. Tasks in the group may vary between the group members. In case there are many tasks in the group, its linkable and the group activity will be displayed on the different level.
Task	Activity	Simpliest task type
Join Task	Activity	Task type, which is collecting incoming brances. Previous tasks must be completed before continueing process, but thay can be completed in different times.
Split task	Activity	Starts multible tasks in one time or with the intervals. Tasks that are starting from the split task should follow the tool, that is described in the spilt task or in the task before it.
Repetitive task	Activity	Loop task, that can be performed as many times than the result is positive or meets the requirement(s).
Blog	Instrument	Task, that is using tools, which can be used for bloging
Media (photo, film, audio)	Instrument	Task, that is using tools, which can be used photo, film or audio creating, editing, sharing.
Wiki	Instrument	Task, that is using tools, which can be used for collaborative writing in terms of affordances that are provided by the wiki tools.
Instant messaging	Instrument	Task, that is using tools, which can be used for instant messaging and VOIP calls
Social bookmarking	Instrument	Task, that is using tools, which can be used for social bookmarking
Office suite	Instrument	Task, that is using tools, which can be used for creating presentations, work with spreadsheets and text documents
GTD	Instrument	Task, that is using tools, which can be used as calendar and note taking
RSS	Instrument	Task type, which is visualising using the RSS in the pedagogical process.
-> (arrow)	Indicator	
Input contition	Conditional task	Marks the starting point of the activity
Outout condition	Conditional task	Marks the ending point of the activity

Figure 8: Elements of the new visual representation

At the beginning of the design process of the elements, we defined the need for the object, subject, activity, instrument, indicator, and other types of activities and features it is necessary to have and use in the visual representation (Figure 8). This listing cannot be final and should be extended by the users of the language according to their needs and tasks.

NewYAWL is mainly built on different types of tasks, which are playing similar role to transitions in Petri nets (Russell, 2005). Simplest task (single task) is called atomic task and can be used where no extra conditions are needed. In our case, where many tools with varying purposes are used, tool based approach is reasonable decision. Problem occurred when we realised that naming tools, that can be used for pedagogical purposes will lead us nowhere – we can point out tens of blog platforms, tens of different wikis, and so on. There is no practical reason to try to combine them all in the same solution and it may feel molest for user to decide in the very beginning of the course design process that what kind of wiki (namely!) he or she should use. Possible method to built logically structured tool based tasks would be using the tools families – media tools (photo and video repositories, podcast solutions, photo and video editor etc.), blogging platforms (Wordpress[9], Blogger[10]), Wikis (MediaWiki[11], Trac[12]), instant messaging tools (Windows Live Messenger[13], Skype[14]), office suits (Google Docs[15], Zoho[16]), calendar and note taking tools (Remember The Milk[17]). Visual representation for each of the family is clearly understandable icon which is consisting of atomic task element with an alphabetical letter in it: M stands for media (photo, video, audio), B stands for blogging, and W stands for wiki tools, IM for instant messaging solutions, O for office suits and T for time managing and calendars.

---

[9] http://www.wordpress.com

[10] http://www.blogger.com

[11] http://www.mediawiki.org

[12] http://trac.edgewall.org

[13] http://get.live.com/messenger

[14] http://www.skype.com

[15] http://docs.google.com

[16] http://www.zoho.com

[17] http://www.rememberthemilk.com

Figure 9: Tool types (instruments)

Roles of the different people have not been used in graphical representation neither in YAWL or newYAWL. YAWL supports resource perspective view, which assigns work items either direct way (direct allocation), or according to the role/s a person plays (role-base allocation) (YAWL, 2007). Work items are being used as soon they are created, by offering them to one or more users. User is able to choose allocation of work and time of starting the activity. Multiple items can be executed at the same time, which makes possible to use simultaneous processes by the same user or role (Russell, 2005). In our case we have found, that visualising the roles in the graphical model is crucial and one of the possibilities is to use Activity diagram swimlanes from UML to visualise different roles on the graphical model. Sometimes are partitions displayed in rows or columns to display more clearly their grouping (O'Docherty, 2005). Even more visual method is to use swimlanes in activity diagram. That means actions are organised into separate boxes (or to be precise, into swimlanes) according to organisational units (customer, sales, stockroom etc.), for example (Accueil, 1997). In our model, we are using dotted line (swimlane separator) between different roles instead of solid line in both sides, which should be less distracting on line crossing areas. Although there is no certain ordering of the swimlanes in UML activity diagrams (Accueil, 1997), for course design we are using the order teacher – group – student. That kind of formal row do not indicate super role of the teacher but helps users to get better full-scale overview of the course design.

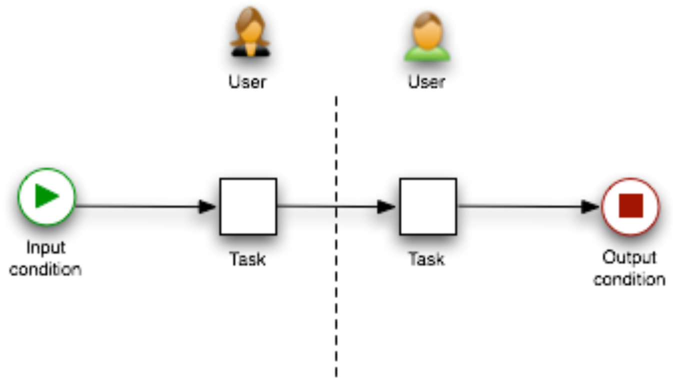


Figure 10: Swimlanes are separated with the dotted line and marking the different roles on the same diagram

Describing events in newYAWL has been defined through the event-based triggering tasks, but the way of visualising these can be a bit indistinct. Triggering tasks that are showing work items that are directly controlled from outside of the context of the process instance. Triggers are having always an unique name and related with certain task (Russell et al., 2005). We have been defined event as a certain place at the timeline, which can be named as a starting or ending point of the activity. Activities can consist of many different tasks, which are forming finally certain activity (introducing the grading, for example). Basically one swimlane between separators can be showed as a one activity, which can be used in one certain model or exported and used in some other.

Different events that are related with varying user roles are having slightly different type. There is no predefined event types but examples can be group work, individual study, reflection, communication, grading, analysing or whatever event, that is consisting of more than one task.

Like newYAWL, our product uses various tasks to model the whole diagram. As mentioned before, atomic tasks with tools are one of the kinds, but there are several other types of tasks: repetitive task, join task and split task. Repetitive task is unmodified, as it is a part of the newYAWL and marks the tasks that can be performed as many times than the result is positive or meets the requirement. Basically it can be handled as a loop, which has while, repeat and combination loops (Russell et al., 2005). Repetitive task can be draw as a square with the small green arrow head on the top left corner and one in the bottom right corner. Direction of the arrow head represents the loop process inside of the task.

Join and split tasks are simplified versions of similar tasks used in YAWL, where are defined AND-split task, AND-join task, XOR-split task, XOR-join task, OR-split task and OR-join task. To simplify the split and join tasks, we left one of the each type. Join task marks the task type, which is collecting incoming branches while it does not matter whether the tasks before the join tasks are completed at the same time or not but it is impossible to continue with the joint task if one of the tasks before it is unfinished. Split task starts multiple tasks in one time or with intervals and there should be remembered that each task should follow the tool, which is described in the split task or before it. Join and split tasks can be attached to each other into one task but it is crucial to pay attention, whether join task has finished before the split task. Visual element of the joint task is combination of the square and triangle, which is directing to the right, split task, is using the triangle on the right side of the square and its direction is to the left.

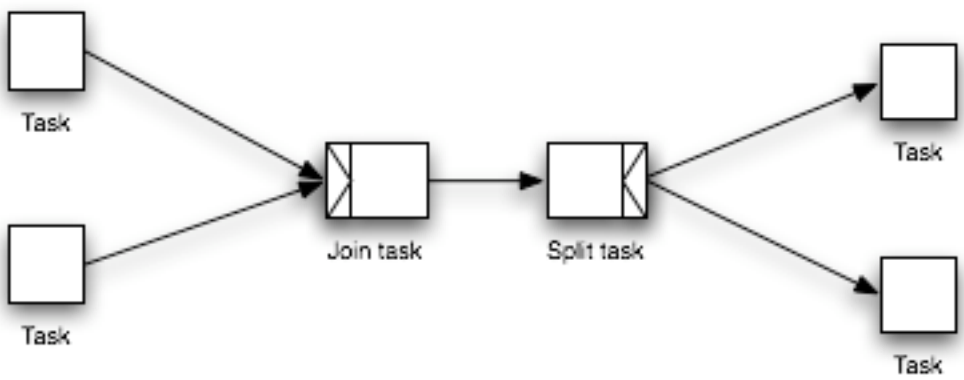


Figure 11: Join and split task

Every element of the model, except the user roles and swimlane separators, are connected to each other with direct graph, which is pictured as an arrow with a filled head. Arrows are usable only in one way, which indicates the direction of the process flow. Join and split task may connect more than one arrow at the same time, but the connection point of the different arrows should be the same in both cases.

Input and output conditions that are marking the starting point and the ending point of the process are situated as the first and the last symbols in the row. Those conditions do not have extra note about the previous or next activities. Input condition is visualised as white circle with green edge and a green triangle in the middle that is directed to the right. Output condition is a white circle with red edges and with a red square in the middle.

Figure 12 shows the symbols that are used in our visual representation.

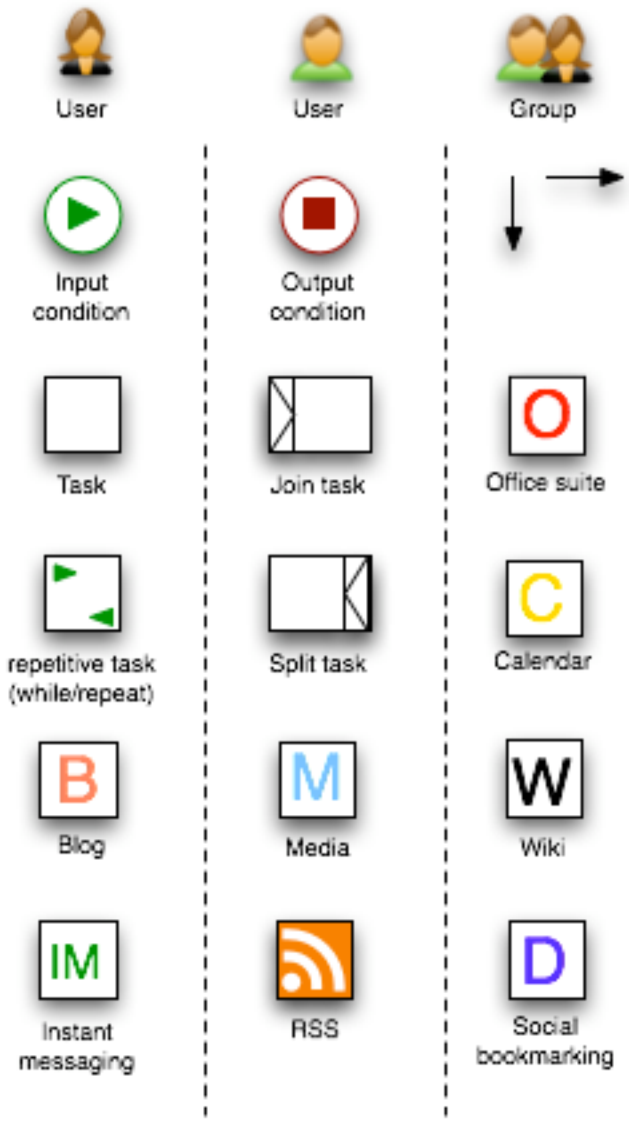


Figure 12: Symbols of visual modelling language

3.3. OmniGraffle stencil

OmniGraffle[18] is a powerful diagramming tool for Mac OS X, which is used for creating professional looking process charts, fast graphical design mock-ups and more. Via intuitive drag-and-drop interface and special shapes groups called stencils, users are able to create illustrations clean and quickly. Stencils are part of the program core but users are able to create and add their own or modify existing ones. They are consisting of small icon-size pictures which can be used for building a needed output for the current exercise. Many different pre-built and user generated stencils are being used for drawing UML diagrams, mind maps, creating photo albums, CD covers, garden layouts, family trees and etc.

Although OmniGraffle is originally designed for Mac OS X only, diagrams created with it can be exported to various formats, including vector, which makes its outcomes usable on other platforms (Windows, Linux) as well. After exporting the diagram to vector graphics, users are able to manipulate them almost as functional as in OmniGraffle.

Purposes of using the OmniGraffle as a software tool for representing the visual side of our outcome can be named here:

- simplicity of adopting the existing newYAWL stencil
- fast and rapid creation of modified stencil
- export feature to vector graphics format
- opportunity to share and get valuable feedback from the OmniGraffle users community

As author has been adopted newYAWL as core language for new appliance, one of the existing stencils has been taken for the starting point here. Original stencil, which can be used for drawing YAWL and workflow nets and is created by Frank Puhlman, is freely downloadable to all at OmniGraffle stencils library Graffletopia[19] .

Collecting a feedback from the OmniGraffle users' community is not crucial aspect in terms of this thesis. It is more important to have possibility to upload our stencil to the

---

[18] http://www.omnigroup.com/applications/OmniGraffle/

[19] www.graffletopia.com/stencils/323

community website and analyse users' comments as the most direct opinions on the fly. Collected feedback will be gathered and suggestions are being adopted in the next stages and versions of the language.

In addition to presenting the stencil in Graffletopia, author has been set up the webpage with short introduction about the stencil and the modelling language behind that. Extra information about the research behind the development process of the visual language makes it more easily understandable for bigger audience and serves the outcome of the thesis in decent public way.

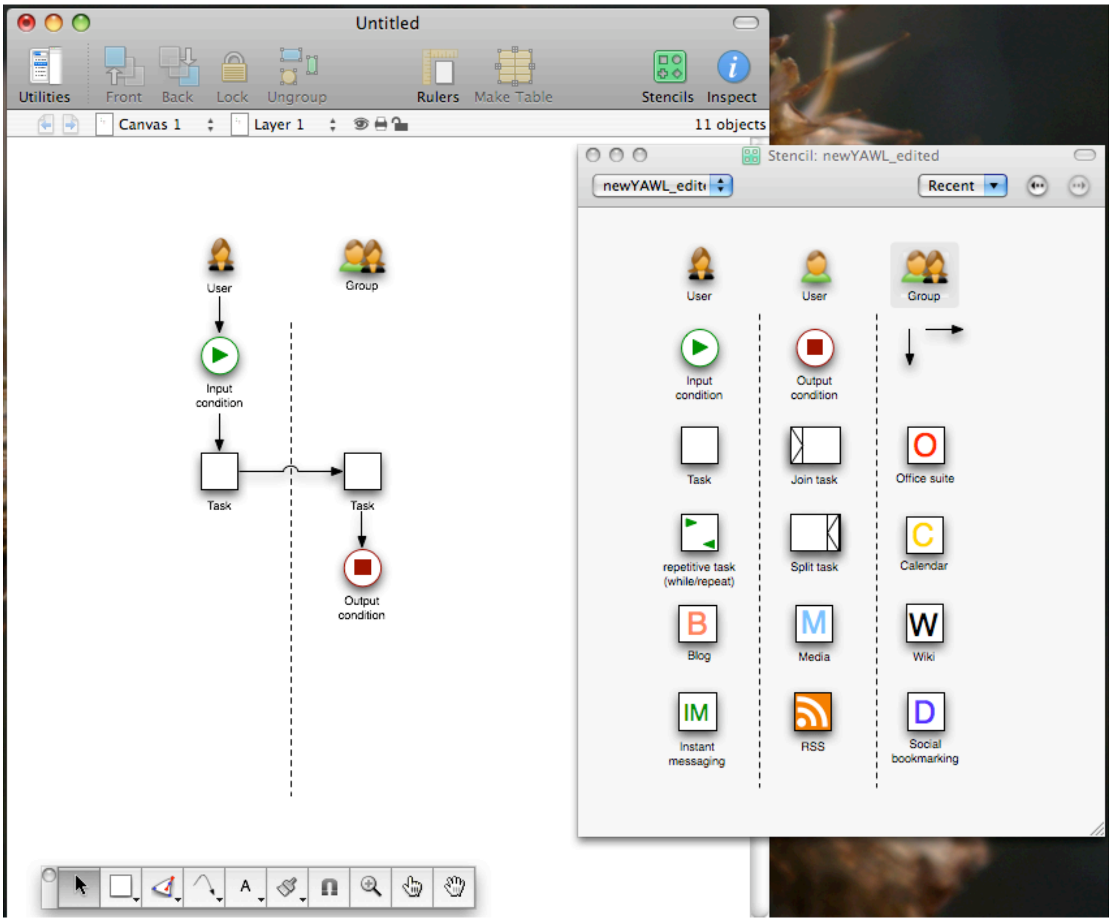


Figure 13: Screenshot of OmniGraffle stencil in use

4. Evaluation

As a research method, action research was conducted as part of the thesis. Creswell (2002) have claimed that action research is typically undertaken in a school setting and among educators and it can be described by five characteristics:

- A practical focus;
- The educator-researcher's own practice
- Collaboration
- A dynamic process
- A plan of action
- Sharing research

Action researchers are interested in examining their own practices rather than studying someone else's practices. As they study their own situation, they reflect on what they have learned as well as what they can do to improve their practices (Cresswell, 2002).

In the context of this thesis, action research was used so that author gathered information about existing visual modelling languages, analysed them, made adoptions and developed the new language, which can be used in describing learning patterns. Also the author tested the language, reflected the problems and offered solutions and ideas learnt from the testing ideas. The aspect of collaboration has been also presented in this process, as the author of the thesis is the member of CET research group and testing and evaluation of developed visual modelling language was part of the collaborative activities. In order to develop the final version of new visual modelling language, several evaluations should be done more, which means that the design has got dynamic and spiral process, which means that researcher "spirals" back and forth between reflection about a problem, data collection, and action. Reflecting, collecting data, trying a solution and spiraling back to reflection are all part of the process of action research (Cresswell, 2002).

Evaluation for the modelling language has been done using the evaluation method formative evaluation. To compare with summative evaluation, formative is judging the worth of a program while the program activities are forming or happening and it is focusing on the process when summative is focusing on the outcome (Bhola,1990). In our case, our outcome needs more internal testing in our research group.

For evaluating the outcome, it is best to use it on the real life situation and conditions that are actually missing that kind of visual representation during the course design. Our aim was to find a target, that meets on few requirements and have as many listed properties as possible:

- uses distributed learning environment as a main platform during the course
- uses as many social media tools as possible
- has different user roles
- has parallel working tasks, repetitive tasks

Online course Designing e-learning[20] was a part of European Master's Programme in Interactive Media (EMIM) and was provided by Tallinn University in spring 2008 and it is a part of the iCamp[21] projects third trial. The EU funded IST research project iCamp deals with formal higher-education settings in which actors are distributed geographically, culturally and over disciplines and in which communication and interaction is mediated through increasingly heterogeneous landscapes of networked tools and services (Fiedler, 2006).

Course objectives were to introduce to students the theory and practice of designing e-learning environments, elements and activities in the era of Web 2.0. Through the practical exercises and discussions were participants able to familiarise with e-learning concepts, issues, methods and more. Various social software tools were used during the course, which enabled students to learn them through the practical tasks.

Roles of the actors in the course is slightly different, there are teachers or facilitators and students, who are acting as an individuals and also as part of the group during the course. A huge number (around 80) of the participants defined course structure a bit different than the classroom one, for example giving feedback for the students cannot be as fast than in the face-to-face lectures. That dictates use of larger number of

---

[20] http://www.htk.tlu.ee/elearning

[21] http://icamp.eu

facilitators to be used and also giving more freedom for solving the tasks given in the lectures.

Facilitator's role in the course was to observe, offer support, to encourage and to keep eye on the students' activities. Facilitators give the task to introduce you among the other participants, which included forming the groups in the learning management system Moodle and using the wiki solution within it. His or her task is also to collect all the students' blogs from the social bookmarking system Scuttle, which are findable via common keyword or tag and to add those blogs into the blogroll of the course main blog. During the course, facilitator supports and encourages students through their weblogs by commenting them actively.

Students who take part or the course were not originally from the one country, international group consisted of students from Estonia, Finland, Bulgaria, Spain and more. Working language in the course was English. Participants were divided into smaller groups which were consisting of not more than 10 persons, which helped to work more intensively than in the one huge one.

4.1. Usability inspection

The course, which is lasting from March to June 2008, is intensive and capacious. According to this and other reasons the initial text based course design was hardly graspable for strangers and for those who are not familiar with the concept of that kinds of course design. For that reason we choose only one's week activities for usability testing of our visual language. The first week's topic was to introduce e-learning and course design and tasks were to introduce their self, assembling students' personal social environment to participate at the course, introducing different e-learning trends and their influence on the course design principles. Activities helmed several web-based tools like blog, videowiki, social bookmarking tool Scuttle[22], Moodle LMS and other. No complicated or grading tasks were presented.

We started our testing from the rough paper prototypes (see Annexes 1-4),which is the easiest and less time consuming way for creating fast mock-ups on a fly. Usability inspection involved general design principles that should be followed when using the

[22] http://sourceforge.net/projects/scuttle/

certain modelling language. Also, guidelines and requirements for the new visual modelling language were part of the inspection process as well; like we have defined that the final outcome must be usable even without the computer and should be possible to draw the whole model with paper and pencil.

The second noticeable extension to the first prototype is overall visual representation, precisely orientation of the diagram. Those, who are familiar with project management and diagrams used there for visualising the time flow and activities, can imagine long graphical tables that are visualising the timetable. Longer lasting projects may require amount of space, that is finally non graspable and finally almost unusable. Instead of horizontal placement we are using vertical layout, like often is used in different flowcharts, for example.

As we decided to use vertically placed model (see Annex 1), one of the negative side of it popped up in the very beginning – using descriptions within the task is necessary but the way newYAWL is using them (bottom of the task icon) is not the best way. Drawing the vertical model will cause descriptions and connections to cross and the overall picture will be turning out quite messy. One possibility is to move descriptions next to the task icon, which is working as far there are only two types of actors, for example facilitator and a student. But adding the one extra will turn the concept useless again. Here we remained to the structure provided by the newYAWL, except connectors should be visualised with starting from the description, as figure in Annex 3 demonstrates.

One problematic issue, which appeared during the visualising the part of the course, was related with group of actions that are consisting of many different tools. For example learner may use many different tools for performing one action and it might be difficult to describe it in terms of tool based approach, we have been chosen here. Another issue will pop up describing individual and collaborative activities, when it is not sure when individual activity becomes collaborative activity. Or description of the group work, where teacher will not define the tools and share the activities between participants. Those issues need to be considered more in next design sessions.

Although our modified method should be usable with only one colour as well, sample sketches proved, that when teacher is dealing with more complicated course design or pattern, it might be hard to read the model afterword as fast as it might. Bold lines are useful features that can be used and at least A4 size paper that can be scanned and modified in the computer afterwards, for example. Or using post-it notes as re-usable task bases, which can be helpful while designing the course or analysing the patterns on the whiteboard.

Final result of the named course is displayed on the following figure:

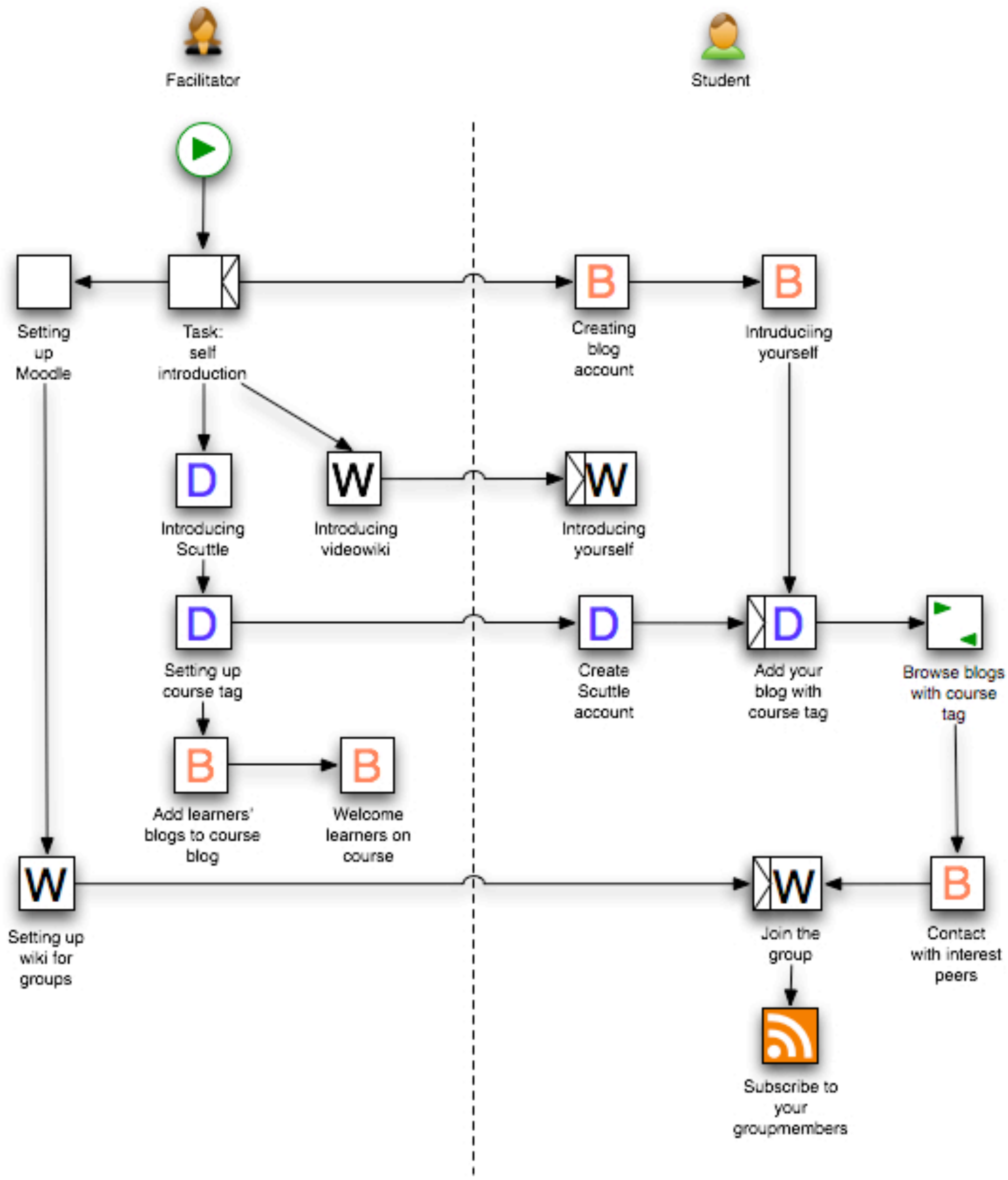


Figure 14: iCamp trial 3, introduction part, modified newYAWL

Modelling the course design with a modelling language should always start with the list of activities that are planned for each user role. By separating them and using bullets instead of the full story will definitely help course designer to keep him- or herself on track.

It is also crucial to have an introduction sheet along the modelling language that would provide quick help about the icon and visual use and with some example designs, that can be adopt into real case. Those, who can use a digital version of the language, see the descriptions already inside of the elements.

Conclusion

This thesis concentrated on the search and development of the visual modelling language, which should be usable as well as designing a single course or bigger learning environment. Nowadays more and more teachers are dropping the use of the closed and outside invisible learning management systems and are starting to use tools that are provided in the Internet. All kinds of Web 2.0 tools have been adopted by the students already, although not for pedagogical purposes. It is possible for teachers to take a leap forward, closer to the students by starting to aggregate those possibilities in their everyday work.

One of the hardest parts during this process, that may occur, is the design process of the course. So many different tools and methods are suddenly available. Our solution for helping to start with the new wave is a visual modelling language that will help teachers to formalise their ideas, analyse them and even compile patterns that are usable for others as well. As sharing has become popular in the Internet lately, why not to try sharing your experience of teaching through the formalised graphical tool?

New solution is not a language to help programmers to design their work processes, it is a visual method for communicating between people, who are not sharing their ideas due to lack of common vocabulary and having different backgrounds on their subject fields.

Action research was used as a methodology in this thesis. The author of the thesis (and the research group of CET) gathered data about existing visual modelling languages, analysed them, developed new visual modelling language and tested it on group of learners. The evaluation was done together with research group and author reflected the problems of new language and also suggested some new ideas and solutions for developed language.

Aim of the thesis was to find and compile an appropriate visual modelling language, which is based on existing visual modelling language, in order to support the design process of the different pedagogical patterns and activities.

Development process of designing visual modelling language was based on Pervasive Usability Process model (Brinck et al, 2001), which was modified to serve our goals. Results of the thesis are:

- Existing workflow languages are not the most appropriate to be used without the modifications and adaptations for describing pedagogical patterns in distributed learning environments, because they do not follow the principles of pedagogical design elements;
- Some certain elements should be adopted or presented in existing visual modelling languages, for example users, events, tool based tasks in order to support the better visualisation of the distributed learning environment;
- For supporting the activity theory based learning design model, visual modelling language should involve the following parts: object, subject, activity (consisting of task, the user role and rules) and instrument.

The developed and described visual modelling language is recommended to use in a learning environment design process, where no massive course is involved. Its graphical representation does not allow use of multiple user roles on one graph, which makes it hard to be practiced on more complicated cases. Those shortages should be taken into account in the further development process of visual modelling language developed in this master thesis.

References

Flowcharting Techniques, an IBM manual (1969) Available at http://www.fh-jena.de/~kleine/history/software/IBM-FlowchartingTechniques-GC20-8152-1.pdf [2008 May 04]

Garrett, J. J. A visual vocabulary for describing information architecture and interaction design, 2002. Available at http://Jjg.net [2008 May 04]

Garrett, J. J. (2002). The Elements of User Experience: User-Centered Design for the Web. New Riders Press.

Nielsen, J. (2008). Bridging the Designer–User Gap. Available at http://www.useit.com/alertbox/designer-user-differences.html [2008 May 04]

Mcluhan, M. & Fiore, Q. (2001). Medium is the Massage. Gingko Press.

Dumas, J. S. and Redish, J. C. (1999). A Practical Guide to Usability Testing. Intellect Ltd.

Krug, S. (2000). Don't Make Me Think: A Common Sense Approach to Web Usability. New Riders Press.

O'Docherty, M. (2005) Object Oriented Analysis & Design. Understanding the system development with UML2.0. John Wiley & Sons Ltd

Graham, I & Wills, A. (2007) UML tutorial. Available at http://www.scribd.com/doc/2244370/uml-tutorial [2008 May 04]

Gaines, B.R. (1991). An interactive visual language for term subsumption visual languages. Proceedings of the Twelfth International Joint Conference on Artificial Intelligence. Morgan Kaufmann. p. 817-823.

Saint-Martin, F. (1990) Semiotics of Visual Language. Indiana University Press.

Novak, J. D. & Cañas, A. J. (2008) The Theory Underlying Concept Maps and How to Construct and Use Them, Technical Report IHMC CmapTools 2006-01 Rev 01-2008, Florida Institute for Human and Machine Cognition. Available at:

http://cmap.ihmc.us/Publications/ResearchPapers/TheoryUnderlyingConceptMaps.pdf [2008 May 04]

Brinck, T., Gergle, D., and Wood, S. D. (2001). Usability for the Web: Designing Web Sites that Work (The Morgan Kaufmann Series in Interactive Technologies). Morgan Kaufmann.

Snyder, C. (2003). Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces (The Morgan Kaufmann Series in Interactive Technologies). Morgan Kaufmann.

Russell, N, ter Hofstede, A.H.M, van der Aalst, W.M.P, Edmond, E. (2005). newYAWL: Achieving Comprehensive Patterns Support in Workflow for the Control-Flow, Data and Resource Perspectives. Available at http://is.tm.tue.nl/staff/wvdaalst/BPMcenter/reports/2007/BPM-07-05.pdf [2008 May 04]

Waardenburg, M & van Emmerik, M. (2004) Workflow Management in an Internet Environment. Axxerion. Available at http://www.axxerion.com/download/axxerion-workflow.pdf [2008 May 04]

YAWL (2007) Frequently Asked Questions. Available at http://www.yawl-system.com/services/faq/faq5.php [2008 May 04]

Accueill (2007) Activity diagram. Available at http://www-inf.int-evry.fr/COURS/UML/notation/notation10.html [2008 May 04]

Carroll, J. M. (2000). Five reasons for scenario-based design. Interacting with Computers, 13(1):43-60.

Budd, J.W. (2004) Mind Maps as Classroom Exercises. Journal of Economic Education, vol. 35, no. 1. p: 35-49

Davidmann, M. (1998). How the human brain developed and how the human mind works

Avgeriou, P., Papasalouros, A., Retalis, S., & Skordalakis, M. (2003). Towards a Pattern Language for Learning Management Systems. Educational Technology & Society, 6(2), 11-24.

Leinonen, T., Toikkanen, T., Purma, J., & Põldoja, H. (2008). LeMill – uuden sukupolven opettajayhteisö tuottaa oppimisaihioita yhdessä. In L. Ilomäki (Ed.), Sähköä opetukseen!Digitaaliset oppimateriaalit osana oppimisympäristöä (pp. 114-125). Vammala: Opetushallitus.

van der Aalst, W. v. d. (1998). The application of petri nets to workflow management. The Journal of Circuits, Systems and Computers, 8(1):21-66.

SeungSoon. P & YoungIn, K. (2008) Applying Petri Nets to Model Customized Learning and Cooperative Learning with Competency, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.2

Goldstine, H. H. (1980). The Computer from Pascal to von Neumann. Princeton University Press.

Cohen, P. (1998). Flowcharting sessions. Available at http://www.hci.com.au/hcisite2/toolkit/flowchar.htm [2008 May 04]

Bowman, M. (1999) What is Distributed Learning? Available at http://techcollab.csumb.edu/techsheet2.1/distributed.html [2008 May 04]

Bergin, J. What are Pedagogical Patterns. Available at http://www.pedagogicalpatterns.org/ [2008 May 04]

O'Reilly, T. (2006) Web 2.0 Compact Definition: Trying Again. Available at http://radar.oreilly.com/archives/2006/12/web-20-compact-definition-tryi.html [2008 May 04]

Eckstein, J. & Voelter, M. (2001) Learning to Teach and Learning to learn. Patterns for Pedagogy. In: Proceedings of netObjectDays 2001. Erfurt, Germany

Bhola, H. S. (1990) Evaluating "Literacy for development" projects, programs and campaigns: Evaluation planning, design and implementation, and utilization of evaluation results. Hamburg, Germany: UNESCO Institute for Education

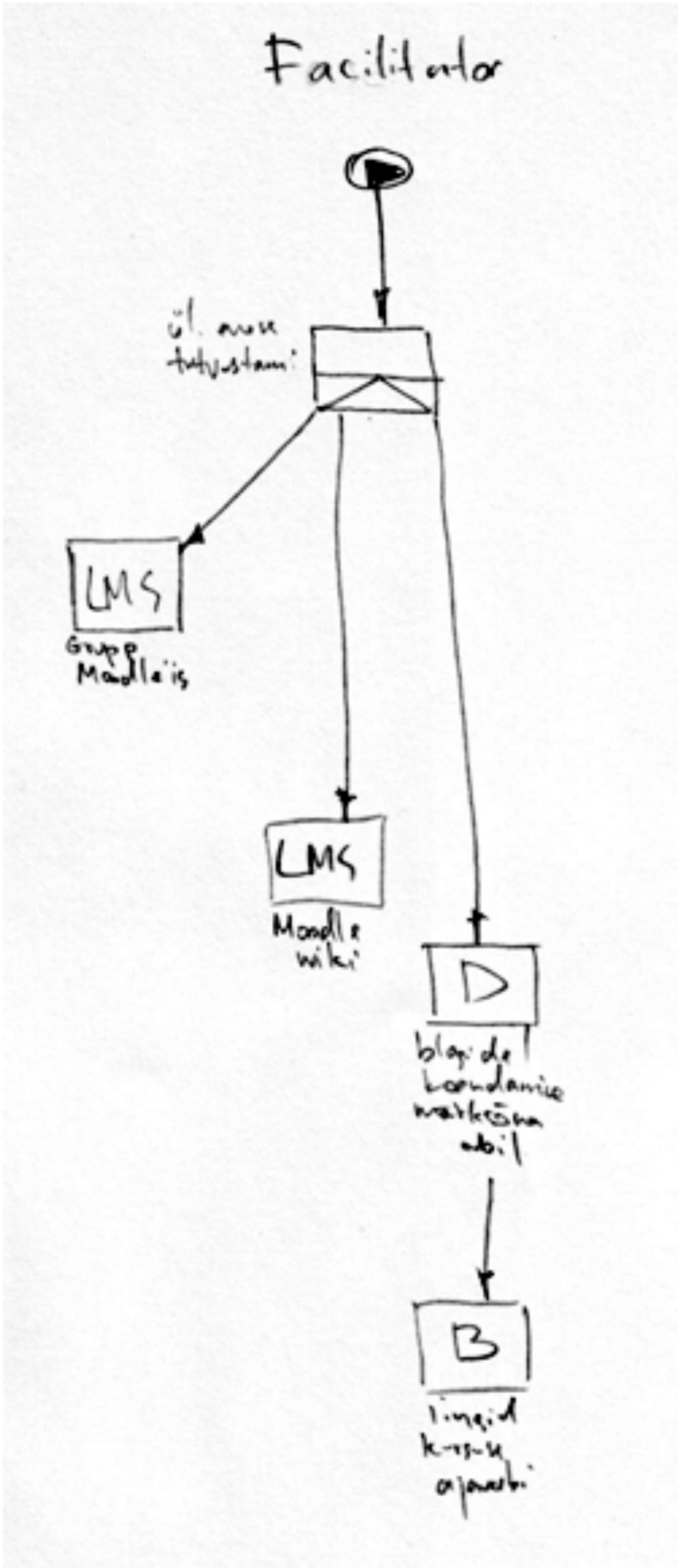
Fiedler, S. (2006) Landscapes of tools and services reconsidered: New directions for informal learning support in higher education? Informal learning and digital media (DREAM conference). Odense, Denmark

Creswell, J.W. (2002). Educational Research. Planning, Conducting and Evaluating Quantitative and Qualitative Research. New Jersey: Pearson Education, Inc.

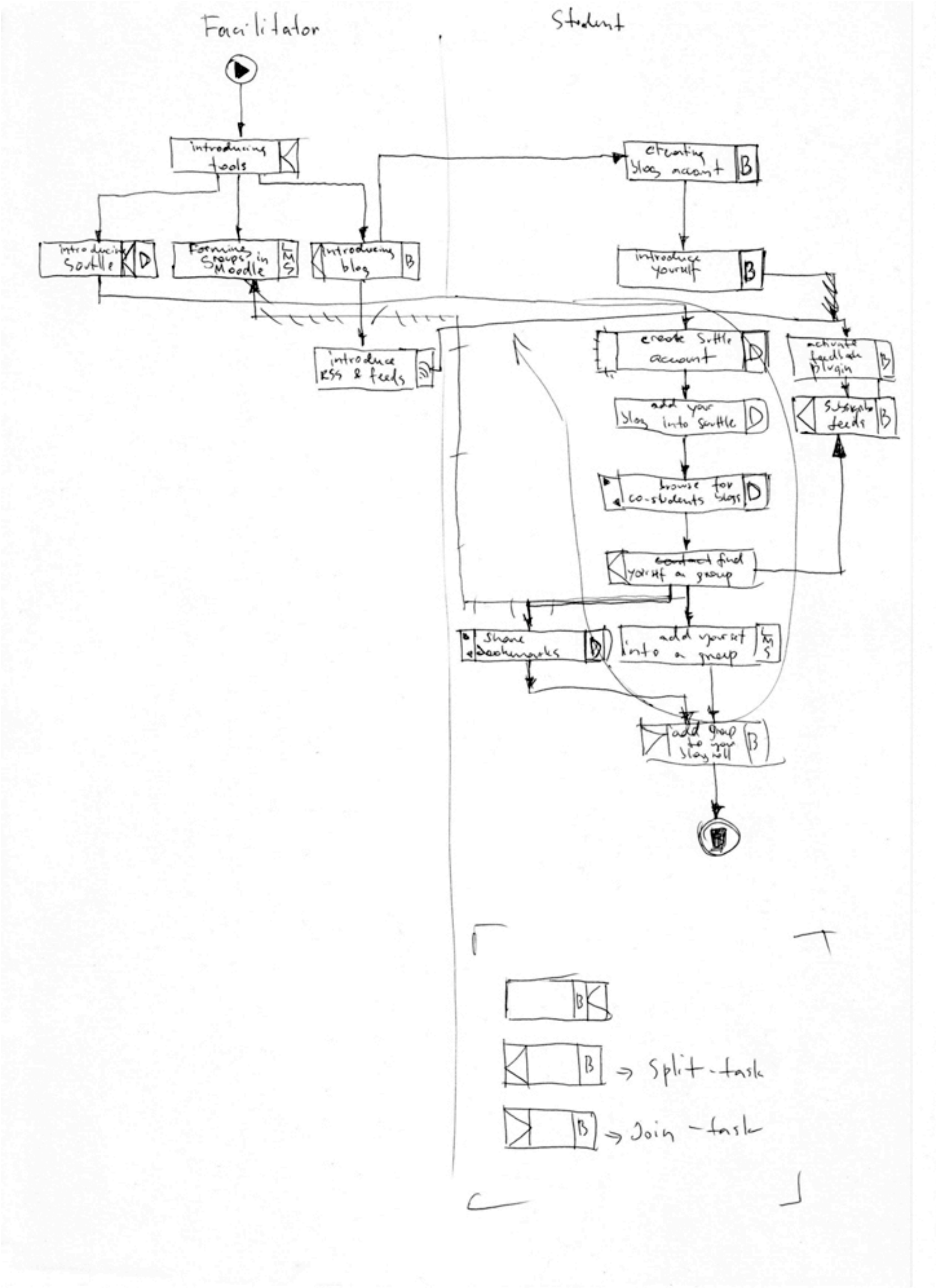
Engeström, Y. (1987). Learning by expanding. Helsinki: Orienta-konsultit.

Väljataga, T.; Pata, K.; Laanpere, M.; Kaipainen, M. (2007). Theoretical Framework of the iCampFolio – New Approach to Evaluation and Comparison of Systems and Tools for Learning Purposes. Duval, E.; Klamma, R.; Wolpers, M. (Eds.). Creating New Learning Experiences on a Global Scale (349 - 363). Berlin/Heidelberg: Springer

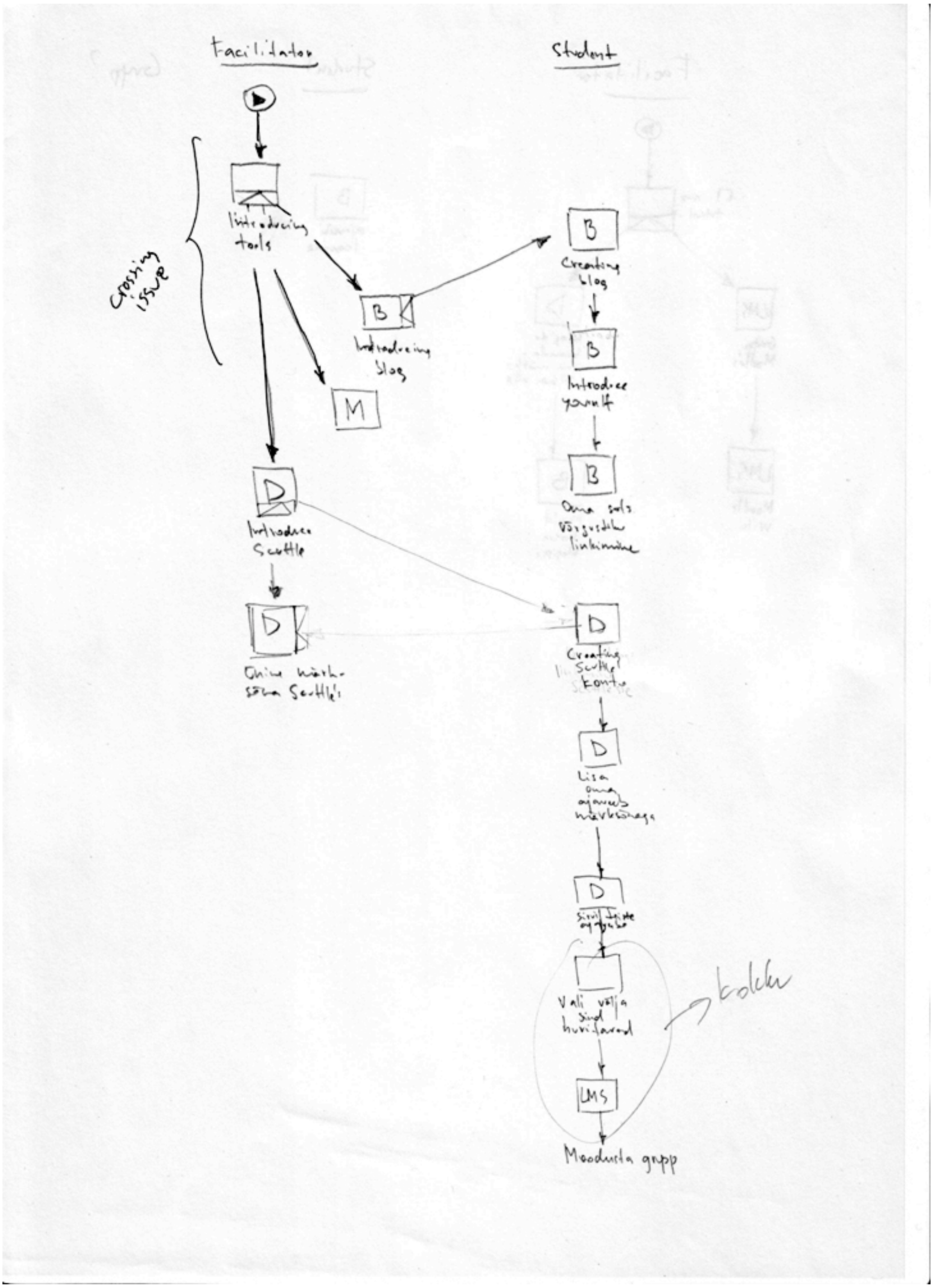
Annex 1, Facilitators' activities, vertical arrangement



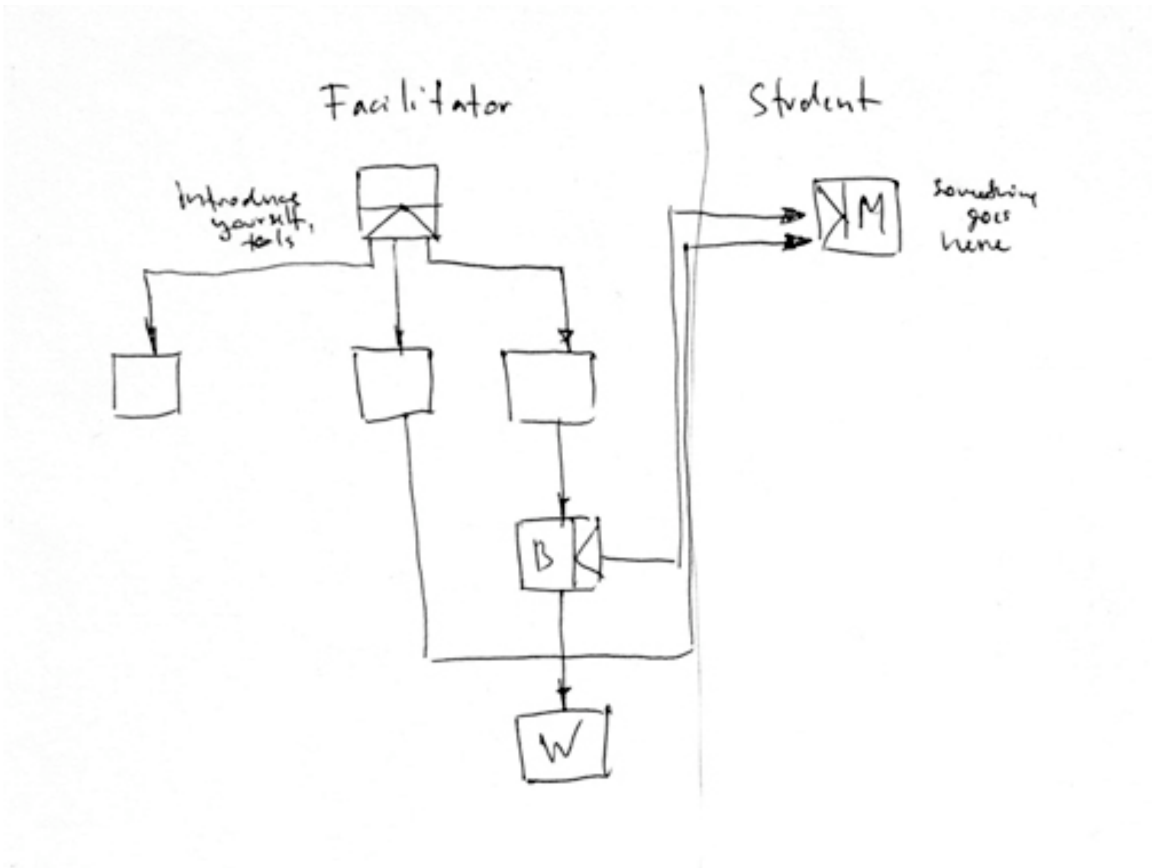
Annex 2, descriptions inside of the icons



Annex 3, Crossing lines issue



Annex 4, Different types of connections



Kokkuvõte

Märksõnad: visuaalne modelleerimiskeel, pedagoogilised mustrid, newYAWL

Käesolevas magistritöös kirjeldab autor visuaalsete modelleerimiskeelte kasutamist pedagoogliste mustrite kirjeldamiseks. Olles ise osa Tallinna Ülikoolis tegutsevas uurimisrühmast, kelle tegevusalaks on erinevate haridustehnoloogiliste vahendite välja töötamine ja juurutamine, peab autor vajalikuks antud magistritöös ilmnevate teemade vajalikust:

- Kuidas ja mil viisil on võimalik kasutada visuaalseid modelleerimiskeeli pedagoogiliste mustrite kirjeldamiseks
- Millistest osadest peaks koosnema modelleerimiskeel, mida saaks kasutada hajutatud struktuuriga õpikeskkonna analüüsimiseks ja disainimiseks

Magistritöö raames valiti uurimise aluseks modelleerimiskeel newYAWL, mida adopteeriti ning rakendati koos lisadega reaalse kursuse mudeli kirjeldamiseks. Hinnang keele kasutatavusele on antud magistritöö autori poolt ning keele arendamine ning edasine evalveerimine jätkub ka peale magistritöö valmimist.

Magistritöö pikkus on 50 lehekülge, sisaldab 18 joonist ja ühte tabelit. 36 kirjanduse ja veebiviidet on lisatud kirjanduse loetellu. Magistritöö on kirjutatud inglise keeles.