A Framework for Narration and Learning in Educational Multimedia

Jesper Mosegaard and Jens Bennedsen
Department of Computer Science
University of Aarhus, Denmark
{mosegard, jbb}@daimi.au.dk

Abstract: In this article we describe a multimedia adventure game framework for a learning environment to support the teaching and learning of introductory programming. In the framework we have conceptualized two important aspects of such an environment: narration and learning topics. We describe the interplay between these aspects and how the framework utilizes this to adapt the learning process to the individual student. The motivation for the separation is to help the teacher balance the two main driving forces of an edutainment product: entertainment and learning. It is the responsibility of the teacher to define the range of stories and topics using the framework. The framework provides a complete learning environment where the teacher merely needs to define the content.

The Lingoland Project

The overall goal of the Lingoland project is to develop a learning environment for teaching introductory programming. The learning environment is designed as a framework for an adventure game in which the teacher can instantiate the concrete learning environment to suit his students. In the prototype, (see Figure 1) Lingo is used as the programming language, but the ideas are not bound to any particular programming language. More information can be found at www.daimi.au.dk/lingoland.

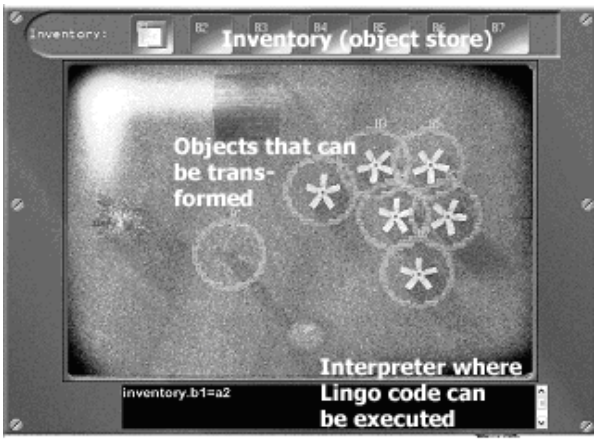

Figure 1: screenshot showing the prototype build on the framework.

The ideas behind Lingoland have evolved through a series of workshops and discussions with colleagues with a solid teaching experience as well as a theoretical background in pedagogy. The Lingoland project was launched in August 2001 with the participation of Peter Bøgh Andersen, Jens Bennedsen, Steffen Brandorff, Michael Caspersen, and Jesper Mosegaard.

We use the game and narration metaphors in order to support constructivist pedagogy (Ben-Ari 2000). We believe that today's students have a clear understanding of games and story, so their learning can build on top of their already established knowledge structures. The concrete implementation of the pedagogy in the game is left to the teacher. He can adapt the environment to the pedagogical approach he wants – a linear or a spiral approach for example (see Bergin 2000).

In the game, the students learn through example, reading or correcting already established code or writing new code. The code is presented in the context of the game as small pieces of code that control separated entities, e.g. an agent or a windmill. The reason the code is to be changed is explained through the game world, and the success of the change is directly determined by the behavior in the game world.

If one wants to learn a new foreign language, say German, one efficient approach is to spend some

time among native german speaking people. We are not arguing that learning a programming language is identical to learning a natural language, but we find the metaphor of spending time among native speaking inhabitants when learning a language useful also when it comes to learning a programming language.

Like all metaphors, the metaphor of viewing programming education as learning a new natural language breaks down at some point. In order to support the metaphor at the outset, the programming language is presented as the language of mechanical creatures in a fictive world and the language used to describe these creature's properties. Hopefully the students will then accept the primitive and strange way these machine-born creatures communicate.

Because the metaphor of game and story will break down at some point as the student learns to read and write programs in Lingo, we explicitly use the breakdown situations to make the student reflect on the current situation. This is done when showing the code behind some agent and through our interface that imitates a command prompt through which the student can interact with the agents (Andersen 2002).

Learning Through Game and Story

The specific goals for the research as presented by this article were to devise a structure that can represent the story and the learning topics. We have successfully conceptualized the coexistence of plot objectives and learning topics in the context of Lingoland in such a way that these topics enable teachers to build a meaningful story and specify the sequence of topics to learn. The actual quests presented to the student can be dynamically chosen by the system to fit the competences of the student. Furthermore the difficulty of integrating the real world learning topics into the fictive game story has been addressed.

Many different definitions of narration exist; we have used the definition by (Garrand 1997, p.171):
"Briefly a narrative is a series of events that are linked together in a number of ways, including cause and effect, time and place. Something that happens in the first event causes the action in the second event, and so on, usually moving forward in time."

This definition of narration is clearly a very general one. We have intentionally chosen a definition that does not define what a *good* story is. This is completely in line with our wish that the responsibility of creating an entertaining story and planning the learning topics that support the teaching of the curriculum is that of the teacher. It is therefore also the responsibility of the teacher to create an experience that both entertains and educates the student in such a way that a symbiosis emerges.

We maintain information about the progress and abilities of the student, to enable Lingoland to create quests that fit the individual student. While the student is trying to solve the quests, the system collects information about his competence within different learning topics and uses this information to adapt the system to the individual student. The system uses the information to select quests that fit the individual student within the boundaries set by the teacher through the plot objectives and learning topics. The student can also make choices as to which quests he wants to try to solve and how. The discipline of letting the system adapt to the user or letting the user be able to adapt the system to his needs is realized through the concept of a User Model (Fischer, 2000)

The structures supporting story and learning are created within the general framework of Lingoland. In the current prototype we use it to teach Lingo with an object oriented perspective. The prototypical student is a non-major student who needs programming as part of his curriculum, but the framework can be adapted to other kind of students as well.

Related Work

Many people and projects have used games as the driving force for learning – and many people are sceptical about the learning outcome of the game. (Bang, 1997, p. 32) puts it like this (our translation): "Games are an advanced form of interaction that gets an increasingly strong position in multimedia productions aimed at learning. It is of course the enormous success of computer games that inspires for

imitation in applications with a pedagogical aim. It is the dream of any pedagogue to playfully stuff knowledge into the heads of pupils and students. No doubt certain kinds of information can be acquired this way just as skills can be learnt through simulation programs. I am more sceptical to the potential of bringing critical insight via games."

The aim of Lingoland is to give the students competencies in programming – a skill in which critical insight plays a minor role. It is therefore very possible that the use of edutainment for this kind of learning has good potential.

To the best of our knowledge there does not exist a framework with the same scope as Lingoland. As stated before many people have used games and multimedia applications as learning tools, but as far as we know they are not of the same generality as Lingoland. The closest that we know of is (Blank, 2002) and (Garner, 2002). Blank's CIMEL in not designed as a framework but it uses multimedia for learning. Therefore there is no direct representation of the Learning topics or linkage to a story. (Garner, 2002) defines a framework for a didactical model, but there is no multimedia aspect such game or narration.

The Lingoland Framework

In this section we describe the framework in more detail. We will focus on the story and learning structures, but the complete Lingoland framework also encompasses structures to handle the game world consisting of agents, items, interaction etc. See (Andersen 2002) for the general idea behind Lingoland. We will show how the framework structures the plot objectives and the learning topics to present this as a coherent game to the student.

Since Lingoland is designed as a framework in which we have separated narration and learning, it is possible to implement different types of adventure games for one curriculum. One example where this could be needed is in the case of gender differences as to what they consider important elements of games. "The majority of girls felt that the important elements were story line, characters, worth-while goals, social interactions, creative activities, and challenge. Most boys on the other hand, liked fast action and adventure, challenge, and violence." (Klawe et al, 1996, p.2).

The Quest Structure in Lingoland

Lingoland has two main goals. Primarily we have the goal of teaching the student a programming language - Lingo. Secondly we have the goals as defined in the context of the game world - that is, the story or plot.

A single mission or problem presented to the student through the game will be called a quest. A quest will often deal with both types of goals. It is imperative that the Learning topics and the plot of a quest have a strong connection. The student should not feel that the plot is only an excuse for presenting lingo. The plot must support the problem statement and the correct solution to the problem must be easily mapped to the game context.

To implement the structures defining the two goals in the Lingoland application we have to define and formalize the concepts concerning these subjects. The concepts are defined separately to model the two different goals in our learning environment. The two concepts are *learning topic* and *plot objective*.

We must also separate the notion of the planning process and the runtime process. The planning is the process in which a teacher builds up some structure narrowing down the possible span of plot objectives and learning topics. At runtime, this information is used to select and instantiate the actual quests that the individual student is to solve.

It is very importantly to notice that this method of presenting the learning process and the story is not bound to any specific form of story and more importantly not bound to any specific learning topics or subjects – it might not even deal with computer science specific things. As part of this work we have used the structures to represent a very specific group of learning topics namely aspects of programming Lingo.

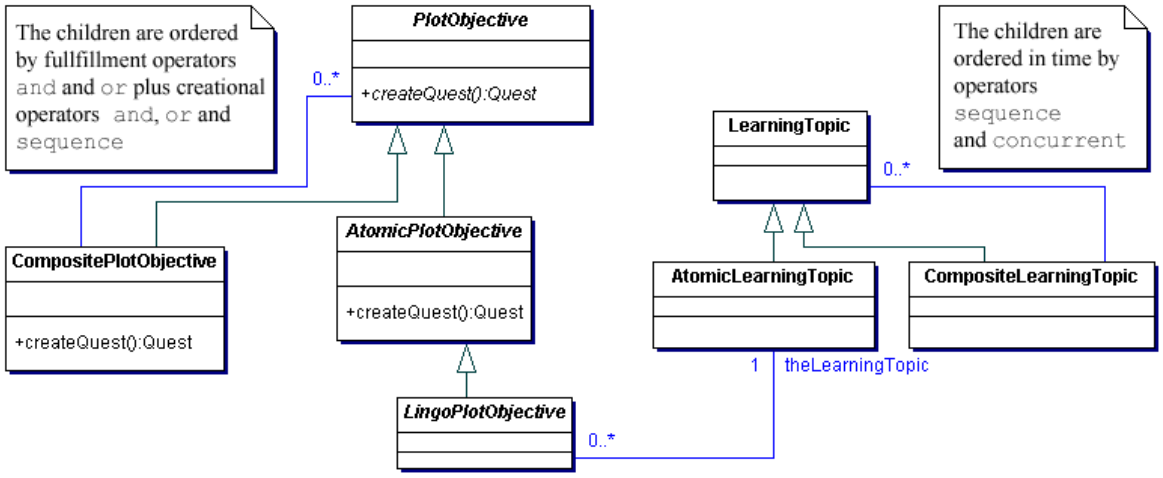

Figure 2: Partial class model for the framework.

Planning of the Learning Process.

As usual in designing a course you have to define the curriculum. Which topics should the student learn and when? Learning topics (see Figure 2) can be either atomic (e.g. rotation of a graphic element) or composite (That is, a group of learning topics). The total structure we call the *learning process tree* and it is a generalization of a traditional course plan.

The planning of the learning process can be done in the following steps:

1) Define the atomic learning topics (e.g. rotation of a graphic element).
2) Group the learning topics in a hierarchy. The composite type and the atomic type are generally called Learning topics.
3) Determine the ordering in time of the learning topics in a composite learning topic (either concurrent or sequential)
4) Express the learning process in the framework.

Implications on the planning.

In a traditional course plan the teacher uses only sequential ordering of topics but this is a dynamic environment, so parallelism is also an opportunity – the actual ordering is determined by the quests that the student finds first or which quests the system creates first to suite the individual student. The children of a composite learning objective can therefore be ordered in time either in sequence (;) or concurrently (‖) indicating when the learning topics can be presented to the student. This structure does not say what the student will actually be exposed to, this will be selected at runtime depending on the actual performance of the student – this structure defines a space in which Lingoland can select topics that creates a coherent story and fulfills the process of learning as defined in the learning process tree i.e. implicitly the pedagogy.

The concepts are implemented in the framework as classes and the teacher can create subclasses or instances of these classes for the concrete instantiation of the framework. By using the different sequence operators it is possible to describe different kind of learning trails for the student. If one teacher wants his

students to follow a very specific trail, he will only use sequence – a strict linear approach to the learning topics. If on the other hand the teacher decides that there is more than one starting point, but the student needs to master the basic level of all these subjects and then move on to a more advanced level, he will create a learning process tree that look like this:

```
Learning ::= BasicLevel ; AdvancedLevel
BasicLevel ::= BasicSubject1 || BasicSubject2 || ... || BasicSubjectN
AdvancedLevel ::= AdvSubject1 || AdvSubject2 || ... || AdvSubjectN
```

Planning of the plot

When planning the plot there are several things to consider. First of all it is done hand in hand with the learning topics described in the previous section. The game and story is situated within some static game world with flowers, trees, buildings etc. This part of the plot has no explicit connection to a learning subject and can be thought of as the stage. The stage is built through a game editor. The narrative drive in the story is delivered through the plot objectives, which are abstract classes that can be specialized and then instantiated to enable a specific range of stories.

A Plot Objective (see Figure 2) represents a piece of story in the game. The hierarchy of Plot Objectives represents the possible stories. The teacher defines this structure when planning the possible plots and thereby the possible Lingo subjects that are to be covered. This structure is called the *story tree.*

The Plot Objective is *formulated* somehow, has some notion of when it is *solved*, and should be solved by some user-built *solution*. The formulation of the objective is in most cases done by one of the associated agents. The solution is often expressed in Lingo.

Plot Objectives can be atomic or composite (see Ryan, 1991, Chapter 7 and 10). The plot objective as presented to the student through story and/or problems is a quest. An atomic Plot Objective is acted out somehow through an agent. A Composite Plot Objective is a quest that depends on other quests to be fulfilled.

A specific atomic Plot Objective called Lingo Plot Objective is what binds (and the only connection between) the plot objective and the learning topics. A Lingo Plot Objective has a reference to a Lingo learning subject defining what the student can learn by solving the problem.

The children of a Composite Plot Objective are grouped with two different kinds of operators, creational and fulfillment. The operators represent constraints on the order of creation of Plot Objectives and when a Composite Plot Objective can be said to be fulfilled.

The creational operators indicate which Plot Objectives actually create quests at runtime. The operators are $\texttt{or}_c$, $\texttt{and}_c$ and $\texttt{sequence}_c$. The statement `p1` $\texttt{sequence}_c$ `p2` indicates that plot `p2` is created when `p1` is solved. `p1 and p2` indicates that `p1` and `p2` are created at the same time. `p1` $\texttt{or}_c$ `p2` indicates that either `p1` or `p2` is created.

The $\texttt{and}_c$ and $\texttt{sequence}_c$ operators are very simple to evaluate but the $\texttt{or}_c$ operator (together with the `concurrent` operator) is where the system can make a choice to create a better learning experience within the possibilities of the story tree and the learning process tree. The reason for this is that the `concurrent` operator ensures that we can have several learning subjects active at any time, and the $\texttt{or}_c$ operator ensures that the system can make a selection of traces through the story tree. This selection is made by an empirically based algorithm based on the level of expertise that the student has reached in the different learning subjects. It is of course possible to define story trees and learning process trees for which it is not possible to create quests that fulfill both trees, but this is not a realistic problem as the designer of the two trees will create trees that can easily work together.

The fulfillment operators, $\texttt{and}_f$ and $\texttt{or}_f$ decide when a given plot objective is solved. The $\texttt{and}_f$ operator indicates that a given plot objective is solved if both children are solved. $\texttt{or}_f$ indicates that only one of the plots needs to be solved.

Associated with each Plot Objective are objects that are necessary for the Plot to have a physical extension in the game. This can for example be a specialized agent with specific text pieces (manuscripts) used

for interaction.

An example of a story tree could be the following simple story (see Figure 3). The town is going to have a party, but there is no power and no disco light, which are indispensable for a party in Lingoland. The student must reestablish power, either by making a windmill rotate or by removing an obstacle from the river which is used to generate power. The student is presented with one of these quests; this choice is made by the system depending on the student's level of competencies in sprite rotation and sprite blending. In the Disco Light Quests the student must find the disco light and bring it back to the town. These two quests are solved using the command prompt (See Figure 2) to assign object references, so no script is edited. But in the third disco light quest the object must be made to blink and this is done by editing the script.

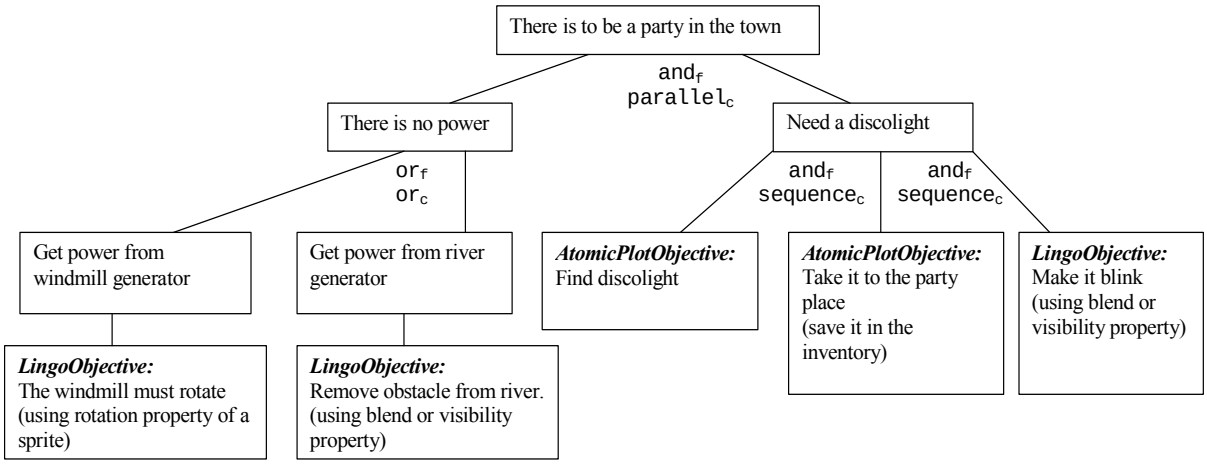

Figure 3: An example of a story tree.

Quest Tree at runtime

A Plot Objective can be in states completed, active or notcreated. The responsibility for changing state from notcreated to active is that of the `createQuest` method (see Figure 2). The state change causes creation of the actual quest that the student is supposed to solve. If the Plot Objective is a composite it will recursively call `CreateQuest` on its children depending on the creational operators. In case of a `sequence`$_c$ of Plot Objectives the method will be called on the first not-completed plot. In the `and`$_c$ case, the method will be called on both operands. In the `or`$_c$ case, the system has the possibility of choosing what plot to create. When the quest is solved the state of the corresponding plot is changed from active to solved.

Dynamic Creation of Plot Objectives

Besides the quests, which lie within the boundaries of the story tree, we have the possibility of dynamically inserting Plot Objectives into the story tree. Theoretically we can create the whole story tree dynamically to fit the individual student if we can define the concept of a meaningful or good story precisely enough.

Architecture

As the game progresses the student will get more and more aware of the game architecture. The climax of the plot and the training in programming will be the point in time when the student is fully aware of his absolute control of the game. The student will be made aware of the game world as nothing but a number of scripts, and will also understand his power.

While the Lingoland prototype helps the student learn about programming and specifically about multimedia aspects of programming, the prototype itself is also an example of how a multimedia program can be structured. To quote (Andersen 2002):"The best teaching environment is one that itself demonstrates what is being taught, i.e. the environment should itself be a good media product that stages the learning process. There are two aspects of being a good multimedia product, an external and an internal. From an external point of view, the product must be understandable, entertaining, aesthetically satisfying, educational, provocative, etc. From an internal point of view the product must posses traditional software engineering qualities such as modularity, low coupling and high cohesion, etc. Good quality is hopefully inspiring to the students, but more importantly it is necessary when demonstrating the system architecture: how interface functionality and model works together."

Because the system in itself is a good example of the construction of a multimedia product, the framework can act as a "Lay of the Land" (Bergin, 2000) so that the students at the beginning of the course can see what they will be able to do when they have completed the programming course.

Evaluation of the prototype

There are two groups of users for the Lingoland system: Teachers and students. The prototype has not yet reached a state where an evaluation with students makes sense. To get an initial response on the ideas from the students we have made a preliminary evaluation of the prototype. The students commented more on the obvious shortcomings rather then the underlying concepts of the prototype. We are planning a formal test to be done in the spring of 2003 if the development goes according to plan.

Conclusion

In this article we have described a framework for a learning environment to support the learning and teaching of introductory programming. The process of learning is motivated through the progress in the adventure game. The game delivers quests dealing with aspects of the multimedia environment. The individual quest can be both pure entertainment and based on the learning topics that the student is supposed to solve. In all cases the motivation for solving the quest is based both in the game world as well as in the aim of learning to program.

We have defined the concepts of plot objective and learning topic as well as their interplay to deliver a learning experience adapted to the individual student. In practice these concepts are used to define the learning process tree and story tree that represent the possible span of stories and learning paths.

The framework provides the teacher with a complete learning environment where concepts and functionality are available to easily build a working learning environment. We have built a prototype based on the framework for a small part of an introductory course in programming.

Future work

The next step in our research is to construct a set of Plot Objectives and Learning Subjects and formally investigate what effects the prototype has on students. Furthermore we will look into the way our system can be integrated into courses

In this paper we have worked with a basic definition of plot, but this might be extended to handle more complex models of the concept of narration. A more detailed model might enable us to dynamically create stories from even smaller pre-fabricated bits of story, and thereby create a story better suited to support the learning process of the individual student.

The framework is a very general description of the relationship between plot and lingo subjects and the two kinds of problems are very loosely coupled. This means that we could create categories of plot and lingo problems that can be paired and thereby dynamically creating actual Plot Objectives in which lingo problems must be solved.

The place where the computer can select a path through the learning process tree and the story tree is clearly defined in the above article. In our future research we will look deeper into how to define the selection operator. This might not be a single operator suitable for all needs but might be a collection of selection functions that can selected by the teacher for a given situation.

References

Andersen, P. B. et al. (2002). Teaching programming to liberal arts students - a narrative approach. Submitted for *ITiCSE 2003* Thessaloniki, Greece.

Bang, J. (1997) Multimedier, interaktion og narrativitet. *Læring og Multimedier*. Aalborg universitetsforlag (in Danish).

Ben-Ari, M. (2001). Constructivism in Computer Science Education. *Journal of Computers in Mathematics and Science Teaching* 20(1), 2001, 45–73.

Bergin, J. (2000). Fourteen Pedagogical Patterns. *Proceedings of the Fifth European Conference on Pattern* Languages of Programs, July 5-9, 2000, Irsee, Germany.

Blank, G. D. (2001). CIMEL: Constructive, collaborative Inquiry-based Multimedia E-Learning. *the 6th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE)*. Canterbury, UK.

Fisher, G. (2000). User Modeling and User-Adapted Interaction (UMUAI). *User Modeling in Human-Computer-Interaction*, 10th Anniversary Issue of the Journal.

Garner, S. (2002). COLORS for Programming: A System to Support the Learning of Programming. *Proceeding of the IS2002.* Pori, Finland.

Garrand, T. (1997). *Writing for multimedia: entertainment, education, training, advertising, and the world wide web*. Boston: Focal Press.

Klawe, M. et al. (1996). *Phoenix Quest: lessons in developing an educational computer game for girls . . . and boys.* Vancouver: Department of Computer Science, University of British Columbia.

Ryan, M. (1991). *Possible worlds, Artificial intelligence and Narrative theory*, Indiana University Press.