Children's use, interaction and learning in a video-game construction environment.

Jakob Tholander

Department of Computer and Systems Sciences

We analyse how a group of children learned to program their own video games during a period of two and a half years. Children from two schools participated in the project; one school for the complete time period and one for the last two semesters. In parallel with the children's work, the learning environment was further refined and developed. The children's learning cannot be viewed in isolation from this process since the tools the children used at the end of the project differed substantially from the initial stages of the project. Furthermore, the second group of children started working with the project when the tools were almost finalized. Such a complex situation of analysis provides a number of interesting foci. These include the kinds of games children constructed, the style of programming that children engaged in, and how they explained different aspects of their games.

This paper presents a high level analysis of the phases and activities of the project. Each year of the project is presented separately and divided into one section on the development of the programming environment and one section on the children's activities.

Based on the analysis a number of properties of importance to kids programming is discussed.

1 The first year

The animated programming environment ToonTalk, and the Playground project was introduced to the children in Swedish elementary school class during 'year 0' when the children were 6 and 7 years old.

1.1 The programming environment

As the project started the programming environment basically consisted of only ToonTalk itself with only some introductory video game examples available. During the first year a

number of example games and components were developed and introduced in the school. The most important games developed were a pong game and an adventure game. A number of puzzles that introduced elements in the programming environment were also developed and used by the children. Furthermore, ToonTalk itself was refined and extended in a number of ways. The most important extension allowed behaviours to be moved between objects in the environment.

1.2 The children's activities

During the first year the school activities were loosely structured focusing around the games and tools that was developed by the project team. The sessions did not have planned schedules. Instead children's interests led the work and the researchers guided the children into learning about how the games and tools were used, and how they could be changed.

The computer facilities where the software was installed were available in the children's classroom for the children to use during as well as after school hours. Project researchers were present once or twice a week to help and support the children in their work. This gave some children a high exposure to the environment and they also quickly developed impressing skills in using the ToonTalk programming environment. Most children also completed a number of puzzle quizzes which introduced ToonTalk programming concepts, such as boxes, sensors, birds, trucks, and robots in a playful fashion.

The children worked with the two games that were developed. They changed and extended the games by adding new, or by changing existing characters and objects in the games.

2 The second year

During the second year of the project development of the games and tool were mostly in focus. However, the school activities became, compared to the first year, more structured around specific issues.

2.1 The programming environment

A number of development projects were focused upon during the second year. The most ambitious task was to implement a new adventure game with a design based on the findings of children's use of the adventure game from the first year. In order for this to be completed major enhancements in the ToonTalk environment was required. Other projects completed during the second year included a number of game components such as mouse control, shooting, movement, and sound effects triggered by system events. These were designed for the children to construct games from scratch.

2.1.1 The children's activities

School activities were intensified, focused on a smaller number of children. A group of six children worked in weekly sessions with two researchers supporting them. All children worked with the components that were developed throughout the year. They also provided input to the new adventure game that were developed, and they evaluated preliminary design ideas.

All children constructed similar games by using these game components. Based on the games children constructed and the components they used, two task-based interviews were conducted to investigate how they understood and explained how the programs running the components worked. Furthermore, all children participated in an online games workshop with

children from England. During the workshop children shared and discussed their games over the Internet and through videoconference software. During the weekly sessions, the workshops were highly motivating for the children when completing their individual projects.

2.1.2 Game construction.

The majority of children's work involved several of the game components that were developed. The children were given extensive support during their game construction activities. A researcher guided them to complete the changes and extensions to the games that they had designed. Hence, game construction was a highly collaborative activity between the child and the researcher. A clear impression was that children could not have completed the changes on their own or with only minor support. However, the children were at all times highly motivated and engaged in the work. They did not simply follow instructions from the researcher, instead they generated ideas and pushed the work forward, even though the researcher helped them to kept focus on completing tasks and correcting mistakes that occurred. The degree to which the children understood what actually went on in the programming environment was therefore not always clear. This led us to conduct a number of task-based interviews which would clarify these issues.

2.1.3 Task based interviews.

Two series of task-based interviews was conducted with 5 children. One focused on investigating the children's understanding of a cat and mice game they had constructed. The children were given what we call *a reconstruction task* where they were supposed to rebuild the functionality in the game they had constructed. They were given two small games, one with the same surface features as the cat and mice game but some of the behaviours had been removed, and one with a character that could throw eggs. The two games had all necessary components to rebuild the cat and mice game in its original form. The second interview focused on the children's understanding of ToonTalk sensors. The children were given what we call a *thinking skills task,* which focused on the abilities children had acquired in explaining the behaviour of properties of objects. They were given a task to program a robot for a game with a cat that should meow if it touched food it liked and then analogously program a robot to growl if it touched food it did not like.

Results

For the *reconstruction task* the children succeeded in identifying the different behaviours required to reconstruct the original game. This included identifying the missing behaviours as well as locating where they could be found. Furthermore, they were also able to decompose the total behaviour of a component into appropriate sub-behaviours.

For the *thinking skills task* children succeeded in explaining how the properties of objects worked. However, it was difficult for them to complete the programming required to get the cat to behave as required. The children often got stuck on details and needed extensive support to complete all the necessary actions to program the robots.

The conclusion from this study was that given that the programming tools have an appropriate level of granularity, children are able to work with interesting programming concepts. Children should not be given tools that require them to interact with the most detailed elements of the programming language. Rather an appropriate granularity level includes game objects such as players and tools, and high-level behaviours such as move left

and right with mouse. Properties of the behaviours should be specifically designed to be open for manipulation by the children.

2.2 The third year

The initial third year work focused on stabilizing the programming environment and setting up a number of well-structured activities that could be carried out with children in two different schools.

2.2.1 The programming environment

The focus of the development activities during the third year was on finalizing the new adventure game into a version that could be used by all children. The game included a set of elements that were designed based on the findings from the work during two previous years with the games. These elements let children change and extend the game in a powerful way. Furthermore, the team in London extended the game components to consistently include descriptive textual comments with speech for children who could not read, as well as visual descriptions of the behaviours. These let children design games on their own by combining game components with graphical elements the way they wanted

2.2.2 The children's activities

During the third year a series of activities were conducted with children at two different schools. These activities included game construction with the adventure game, task based interviews based on the adventure game, and online games workshops with children in London. In total, 10 children completed all of these activities throughout the year. Furthermore, a number of supplementary activities were conducted with the children at the two schools. First, the involvement of children at a new school required introductions to the project activities with children as well as teachers. Second, the experienced children at the old school constructed games from scratch by using a limited number of game components.

Introducing the new school

During the first semester children at the new school primarily worked with the pong game and the old adventure game. They were also given tasks that introduced programming in ToonTalk and other elements in the environment.

From game construction, via games workshop, to interviews of game functionality

Two groups of children from two different schools all completed three consecutive activities, which evolved around the new adventure game. The three activities involved construction of an adventure game, participation in an online games workshop with children in London, and a task based interview regarding the functionality of the adventure game. The first group of children (below called the experienced group) came from the old school and all had about two years of experience of working with the project. The second group (below called the beginners group) came from the new school and were introduced to the project during the fall semester.

Constructing adventure games

The children worked in pairs (except for one child) with support from a researcher. At the first two sessions all children were given worksheet with tasks introducing the adventure game how it could be changed and extended. During these sessions children also started generating

ideas for their games and started to implement these. At the following two sessions the children were given individualized worksheets based on the game ideas they had generated.

All children successfully built games of different extent and sophistication. A number of dimensions of these games were of particular interest and some were also investigated in more detailed analyses:

Styles of programming and designing for programming

- All constructed games involved manipulations of elements in the program environment. A central goal which had been pursued throughout the project was that children should be able to construct games that were meaningful to them and which also required what we have called "real programming" (see case study article). Real programming involves interaction and manipulation of elements in the program environment as compared to manipulations of surface features of pictures.

  Conclusion → Design processes where developers are closely collaborating with children allow the final product to stand up to initial design goals.

View of learning objects

- The children in the experienced group constructed games that involved a wider range of functionality, often outside the scope of the adventure games. They often brought in functionality from other games and from other game components. The children in the beginners group seldom brought in functionality from other games and game components even though had worked with such games. Their games mostly involved manipulations of functionality originally in the adventure game. However, the changes and games of the experienced group were not more sophisticated than the beginners group. The experienced children could clearly separate games and game components from the language elements that implement these. They could clearly see distinguish elements to program with (e.g. robots), from elements built out of programming components (e.g. games). The beginners group on the other hand, saw games and game components as isolated pieces of software. They were not able to distinguish the role played by the different elements in the environment.

  Conclusion → Given an adequately designed (see conclusion above) environment children can program at a rather high degree of sophistication. However, in order for children to develop an understanding where they can distinguish the programming language elements from the programmed artefacts, experience with the environment including pure programming level elements as well as pre-built artefacts, is necessary.

Advanced game construction at old school

A limited number of game components were selected to be used by the experienced children to construct a game from scratch. During the first two sessions the children were given worksheets designed to focus on the program mechanisms of the game components, and how to use these in the their games. We were concerned that the limited number of game components and the structured session would restrict the ideas that children would generate. However, this was not the case, instead this led to highly creative game design processes. The children generated ideas and pursued the programming of these ideas into the system to a larger extent than we had experienced with their work with the pre-built games. Even though they were not able to implement all ideas into their games they often complemented the lack of programmed behaviour with their own imagination. Furthermore, the children were also able to investigate programming elements at a deeper level than had been the case with the pre-built games.

Other interesting events that indicate how children *make use of different resources to participate and communicate in a collaborative situation* also occurred during these sessions. Children with limited conceptual understanding of a task used the concrete metaphors of the programming environment to communicate their understanding to each other and to adults This was not found until detailed investigation of video data was conducted, which suggests the importance of doing micro level analyses of children's learning. Furthermore, children of different ability showed collaborative skills where one more experienced child guided a friend to program something they had mutually decided upon. This rarely occurred among less experienced children, whom mostly took turns in their use of mouse and keyboard. The turn taking was problematic as it was often a source of argumentation between the children, and the child not in control the mouse often lost attention of the task. A more productive kind of collaboration require longer experience with the computer environment and the kind of tasks involved.

Important properties in children's programming environments

From the analysis what can one say about what kind of programming that is mostly feasible for children. Tensions:

1. object oriented versus rules
2. ready made games versus bottom up construction

Questions

→ How classify children's programming activities?

→ How describe the development from these heterogeneous data?

→ How relate field experiments to observations from weekly sessions?