Fables – Exploring Natural Ways of Expressing Behavior to Create Digital Simulations

Andrea Valente[1(✉)] and Emanuela Marchetti[2]

[1] Maersk Mc-Kinney Moller Institute, Embodied Systems for Robotics and Learning, University of Southern Denmark (SDU), Odense, Denmark
anva@mmmi.sdu.dk

[2] Media Studies, Department for the Study of Culture, University of Southern Denmark (SDU), Odense, Denmark
emanuela@sdu.dk

Abstract. We are interested in simplifying digital game design and programming for primary school teachers and their pupils. A central problem in this area is how to express knowledge about interactive digital systems in a simple yet powerful enough way, so that new digital games or interactive simulations can be generated automatically by teachers and pupils descriptions. We propose a novel approach that builds on Simon [4] and Schön [5] and the concepts of *simulation* and *repertoire of exemplars*. Instead of looking at programming concepts like conditionals and loops, we draw inspiration from soft methods like rich pictures, and formalisms like concept maps and mobile ambients. In this paper, we define the concept of fables, where a simple fable represents an exemplar and it can be interacted with digitally, as a simulation. A web-based prototype tool is under development, and we are conducting a series of workshops (last semester of 2017 and first semester of 2018) to discuss, co-develop and test our incremental prototype of the fable editor/player. Early tests and interviews indicate that fables are a viable concept with potential applicability in various domains, and that the current prototype is usable enough for further participatory development.

Keywords: Design thinking · Game development · Education
Knowledge management · Visualization

1 Introduction and Motivation

In past projects, we have worked at simplifying digital game design and programming for primary school children and their teachers [1, 2], to empower them and let them learn by creating their own digital games. In line with [13, 19], we found that teachers engage in forms of creative thinking, acting as finders or makers of multimedia applications and playful experience for their own pupils. Teachers tend to explore digital and non-digital solutions made available through the school system or simply the Internet, searching for materials that could be meaningfully contextualized within their class activities. Nevertheless, some teachers would like to address specific needs or visions for their class, and be able to create ad hoc playful experiences. Hence, as discussed in [1, 2] those teachers experiment with the design of paper-based board games or scripts for role play acts for the pupils, to engage more actively with their learning material. Interestingly, several teachers we interviewed have said that they would recur to designing paper-based games, also because the making of a digital game was seen as too complex for their IT competences. Therefore, in our study we are exploring the making of a system that would empower teachers to develop new games/simulations for their class, leveraging on creative thinking practice, which they seem to master already in their current exploration of paper-based games.

A central problem in this area is how to express knowledge about interactive digital systems in a simple yet powerful way, so that new digital games or interactive simulations can be generated automatically from descriptions created by teachers, pupils, and non-programmers in general. Usual approaches to solve this problem are: provide predefined customizable options (often via a game editor), or turn users into programmers. The latter requires the creation of a special programming language, typically coupled with highly visual and friendly development environments. Although very successful, this approach has problems, the main one being that it takes time for a pupil to be proficient enough to code satisfactory digital games, as discussed in [3]. In our experience the situation is even more difficult if we expect primary school teachers to learn to program digital games, especially teachers with non-technical backgrounds.

We propose to look at a third approach, and in particular we build on Simon [4] and Schön [5], and the concepts of *simulation* and *repertoire*:

- "Simulation, as a technique for achieving understanding and predicting the behavior of systems, predates of course the digital computer" [4].
- "When a practitioner makes sense of a situation he perceives to be unique, he sees it as something already present in his repertoire. […] The familiar situation functions as a precedent, or a metaphor, or… an exemplar for the unfamiliar one" [5].

The idea that simulation and understanding are related is also supported by research in bounded rationality and naturalistic decision making [6]. In naturalistic decision making a person is said to use a simulation to judge the consequences of possible choices: a mental simulation that acts very much like the exemplars in Schön's repertoire.

For us, the link between simulation and exemplars in a repertoire is the following: when a digital simulation of a real-world situation runs, it generates a sequential story about the situation changing over time. An interactive digital simulation (for example a digital game) will generate different stories each run, depending on the user's interaction. Therefore, a simulation can be said to generate or contain multiple stories, and running the simulation produces an interactive non-linear story. In our previous work [1, 2] we repeatedly encountered non-linearity, when studying how children and teachers express behavior of a system to be implemented digitally. Even with something as simple as digital multiple choice quizzes [9], describing alternatives is perceived as difficult and requiring precise definitions. On the other hand, according to Schön, it should come natural for practitioners to create and maintain their repertoire, and since exemplars must also deal with multiple options and outcomes, handling non-linearity should not be the source of the difficulties we observed. We decided, therefore, to explore natural ways of expressing behavior to create digital simulations, usable by teachers and pupils. Instead of looking at programming concepts like conditionals and loops (such as those in *Computational Thinking*), we draw inspiration from soft methods like *rich pictures* [11], and formalisms like *concept maps* and *mobile ambients* [7, 10]. In this paper, we define the concept of *fables*, where a simple fable represents one of Schön's exemplars. Multiple fables can be composed together to create what Schön calls repertoire.

We are currently implementing and testing a web-based tool to support creation and interactive playback of fables. The playback of fables will be an animated, interactive simulation of the changes that occur in the situation being described, and the user will be able to choose how the fable is progressing, always following one among all possible storylines.

We expect that fables will be used by teachers and pupils to express what they know about a situation or a domain, explicitly and in an objectified way, so that other pupils (and teachers eventually) can *explore* that knowledge by replaying the fables in multiple ways, i.e. interacting with the simulation automatically generated by those fables. If needed, a fable should also be used to generate concept maps or linearized and exported as a slideshow. Moreover, we want teachers to be able to create their own digital simulations within the domains of their pertinence by visually or textually define fables, without the need for them to become programmers.

In the following sections we discuss related work and theoretical background for this study (Sect. 2), present our latest prototype (Sect. 3) and initial findings from testing (Sect. 4); Sect. 5 presents conclusions and future work.

2 Related Work and Theoretical Background

Theoretically our study builds on Schön [5] and his notion of repertoire, and Simons' understanding of simulations [4].

In the book "The Reflective Practitioner", Donald Schön argues that: "a professional practitioner is a specialist who encounters certain types of situations again and again" [5, p. 60]. Different terms are then used in each profession to identify these situations, which could be called: "cases" or accounts, or projects or else depending on the specific profession. According to Schön, as practitioners experience different variations of these cases, they develop "a repertoire of expectations, images, and techniques" drawing from each individual case [5, p. 60]. Through this repertoire, practitioners become able to engage in a reflective conversation with the situation given by new cases, as they build on their repertoire, to reflect on the similarities between the old cases and the solutions applied then, which could inspire new solutions for a newly encountered case.

According to Simon [4] a simulation is a partial reproduction of a phenomenon, in which selected relations between the elements and their dynamics are preserved. So defined, simulations are techniques for "achieving understanding and predicting the behavior of systems" [4, p. 13]. Simulations are based on similarities between two different systems, as one system can be altered so to resemble the other. Computers have contributed to spreading the use of simulations and extended the range of phenomena that can be simulated. According to Simon, an interesting and awkward fact of simulations is that the makers of a simulation can learn something new of the phenomenon that they have reproduced in the simulation, even if they know in depth all the elements of the simulation and their relationships.

Combining the perspective of Schön and Simon, we see Schön's repertoire of cases as a simulative resource, as the construction of the repertoire enables the practitioners to acquire a set of expectations, images, and techniques that, in the terms of Simon, enable the practitioners to predict possible solutions for new cases. In the moment a practitioner engages in reflecting on the new cases, she in fact engages in reflecting on how old cases could be *similar* to the new case, and in simulating in her mind what could happen if one of the old solutions could be applied to the new case.

In our study, we aim at leveraging on the simulative potential of repertoires to create a tool that could enable teachers and pupils to create interactive stories (here called fables) and use them as simulations of specific topics. Hence, following Simon [4], by engaging in interacting and editing fables, pupils and teachers would be participating in the formation of a repertoire of simulated cases, which could foster learning and in-depth reflections, individually and in groups, on the selected topics.

A difficulty we are facing is how to express the behavior of the elements of simulations/games. We intend to face this issue by focusing on the design thinking activity in which teachers already engage when designing paper-based board games and role play scripts. Therefore, in our study we work across disciplines, taking inspiration from studies on design thinking and conceptual formalisms, to shape the logic and workflow for our tool. Regarding the learning aspect, we refer to Donald Schön's concept of reflection in action and repertoire [5], to define our scenario. By design thinking, we refer to "an analytic and creative process that engages a person in opportunities to experiment, create" prototypes, and gather feedback for further improvement [13]. In our scenario, designing thinking would be targeted the design and alteration of learning games and simulations, to be eventually improved with the help and active participation of the pupils. Following [13, p. 336], our scenario approaches design thinking as defined by seven main characteristics: human- and environment-centered concern, ability to visualize, predisposition toward multi-functionality, systemic vision, ability to use language as a tool, affinity for teamwork, avoiding the necessity of choice.

The first characteristic is "human- and environment-centered concern", meaning that designers must focus their intervention in addressing human needs, functional and aesthetic. In our study, look at the needs of teachers and their pupils. The second characteristic of design thinking includes the "ability to visualize", as design practice typically requires visualizations of information and/or of the interactive elements of the specific product. The third is "predisposition toward multifunctionality" and the fourth is "systemic vision", as designers should look at multiple solutions to an issue, shifting in between its overview and details, with the goal of providing *systemic* solutions that address complex issues for different groups of users and stakeholders within specific contexts. In the context of our project, we look at the design thinking in which the teachers continually engage when creating new learning activities for their class, while taking into account the multiple needs of the individual pupils and of the class as a whole. The fifth characteristic of design thinking in [13] is the "ability to use language as tool", as designers should be able to verbally articulate to users and stakeholders the creative process and outcome, where visual information is not obvious or perhaps inadequate. From previous studies (discussed in [1]) we know that language represents a central mode of expression for teachers to create their scripts role play and paper-based board games. Since we intend to leverage on current practices, we strive for a scenario in which language and visualization support each other in enabling teachers to express their creativity through digital media, as they do with paper-based media. The sixth characteristic, "affinity for teamwork", is defined by designers' interpersonal skills enabling them to communicate "across disciplines and work with other people" [13, p. 336]. This is another aspect that we take into consideration in the definition of our scenario, in relation to the role of the teachers as creators of playful experiences. The teachers we have observed and interviewed are engaged in designing playful interdisciplinary learning experiences for groups, leveraging on collaborative and competitive play. Finally, we are critical toward the last and seventh characteristic, called "avoiding the necessity of choice". According to this characteristic, designers should search for "competing alternatives" to land on a final solution which leads to "avoid decision and combine best possible choices" [13, p. 336]. We find this characteristic challenging, as it can be interpreted in an ambiguous way: avoiding decision could refer to how designers create solutions based on best practices, sparing the user from the confusion of having to take decisions among too many choices. This last characteristic can be then related to the principle of simplicity in good design [16], and, in this sense, it has been applied successfully to the design of digital media and games for learning [15]. However, in this project we attempt to empower teachers in becoming more independent in their choice of digital media, for their class, hence, we are more interested in offering more choices instead of reducing them. In this respect, we are rather following [14] in pursue of a democratization of design tools for creating games and simulations for learning: in this way, teachers will not have necessarily to recur to IT experts to create simple digital games and simulations, and retain more authorship and control, as is the case when they work with paper and scripts. In line with Schön [5], our scenario aims at enabling teachers to focus on their *reflective conversation* with learning content and potential game elements (mechanics, roles, possible characters, and tasks) that could enrich pupils' learning experience.

With respect to our goal of finding natural ways to express behavior, we have looked at various formalisms (within theoretical Computer Science, and formal methods) and we selected *mobile ambients* [10] and *rich pictures* [11]. We worked with both in our previous research (see [7, 12]); here we want to take advantage of the spatial metaphor and expressive power of ambients, while dropping most of their formal aspects like typing. In our work with rich pictures we found that when semi-structured data (such as ambients or objects in prototype-based languages) are provided with visual representation, they make for powerful and intuitive tools to represent knowledge, even for non-programmers.

For fables, we want a visual representation (also for direct manipulation in the editor of our prototype) that is spatial, but simpler than a bi-dimensional metric space. Informally, we want fables to be visualized like sticky notes of different sizes, where placing a note on top of another signifies inclusion or part-of (or *stacked* sticky notes). Therefore, diagrams representing fables could be composed of rectangular areas in a metric space, eventually placed one above another, or they could be represented by elements of a grid (with an added dimension for stacking elements, similar to what is called z-index in web pages). Finally, most of the geometrical information could be dropped, leaving mostly inclusion: this would result in tree-like structures, topological in nature since proximity and nesting would still remain. Figure 1 shows the spectrum of possibilities, and we decided to work with trees (Fig. 1 on the right). Interestingly, ambients have a tree-like structure and computation is expressed by creating, dissolving and moving boxes around in the tree.

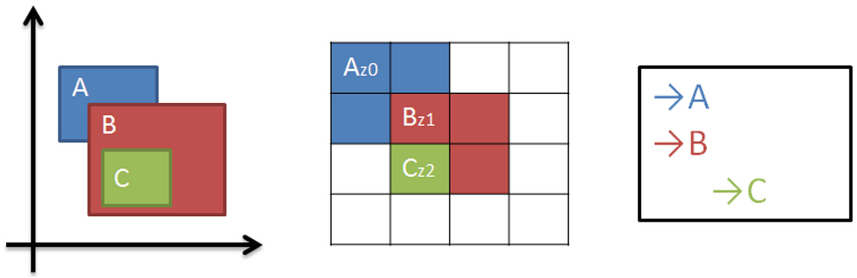


Fig. 1. Range of possibly visualizations: 2D metric space (left), grid with z-index (center), and topological (on the right).

3 The Prototype: An Example, Analysis, Design and Implementation

In this section, we want to define the concept of fables, then look into how a story can be expressed as a fable. The author of a fable should be able to play it back to explore all possible unfolding of the events, and that means that time and choices have to be easy to express in a fable, in a non-technical, natural way. The web-based prototype tool to create, edit and playback fables is called F4BL3s, and it is developed to support the authors by offering textual as well as visual ways to work with fables, leveraging on direct manipulation as much as possible.

3.1 Example of Use: A Ping Pong Match

Consider the following informally defined story. *Alice and Bob are in a room, inside a house, and are playing ping pong. In the room, there is a table and initially Alice has the ball. After Alice serves, Bob has the ball and it is his turn to send the ball back to Alice. Each of them could hit the ball too hard and send it out, which will conclude our story.*

A text analysis of this ping pong story suggests a spatial metaphor, where people and things inhabit rooms, and where a thing might belong to a person. The unfolding of the story can then be represented by defining the *state of the world* (i.e. rooms and their contents) at a certain time. Actions (such as "Alice serves") bring the world to a new state, where contents of the rooms have changed, and people and things have moved, appeared or disappeared. For us each state is represented by a *slide*, as in a presentation, and many slides constitute a fable. The ping pong story can be turned into a fable manually, following this work-flow (see Fig. 3, top part):

- Chose a title (here it could be "ping pong") for the new fable.
- Define the initial state of the world: i.e. draw a tree with nested rooms, then place Alice and Bob in the same room. Finally, add few things (like the table and ball) so that Alice owns the ball, and the table is in the same room as Alice and Bob (see Fig. 2).

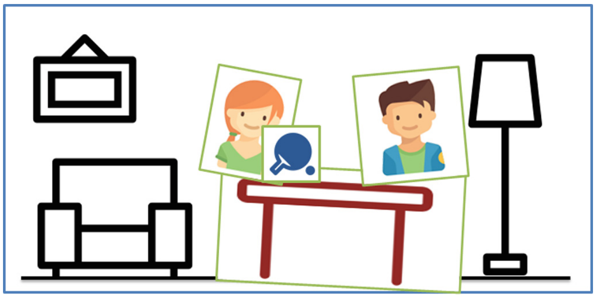


Fig. 2. Informal visualization (This figure uses free icons obtained from www.flaticon.com.) of the first slide of the ping pong story.

- Make a copy of the tree of rooms (i.e. the first slide of the fable) and move the ball from Alice to Bob. Draw an arrow to connect the first state of the world with this new one: the arrow represents an action with label "Alice serves". Draw another action (i.e. arrow) labeled "Bob's answer" that connects the second world tree to the first.

After having defined the "ping pong" fable the author plays it back, deciding interactively step-by-step which actions she wants to perform (if multiple actions are possible for any of the slides), generating a linear sequence of world states. The author can now go back and add a new arrow for the action "Alice serves", expressing the idea that sometimes when Alice serves, she sends the ball out and the game ends. *Non-deterministic* choice (as this situation is called in the ambient calculus) can be easily represented here, simply by defining the playback of a fable in such a way that when 2 actions have the same label the user playing back the fable will not be able to control in which of 2 possibly states the world will end (see bottom part of Fig. 3, in particular the large "Alice serves" arrow going towards Slide2). Sometimes the player will send the user in a state where the ball is moved from Alice to Bob, and other times the ball will *randomly* end out, so neither Alice nor Bob own it anymore. So, in playback the 2 actions will simply be considered a single action with 2 possible, non-deterministic, outcomes.

As we are looking for natural ways of defining fables, incremental definition is very important, since the author should not have to worry about mentally constructing a

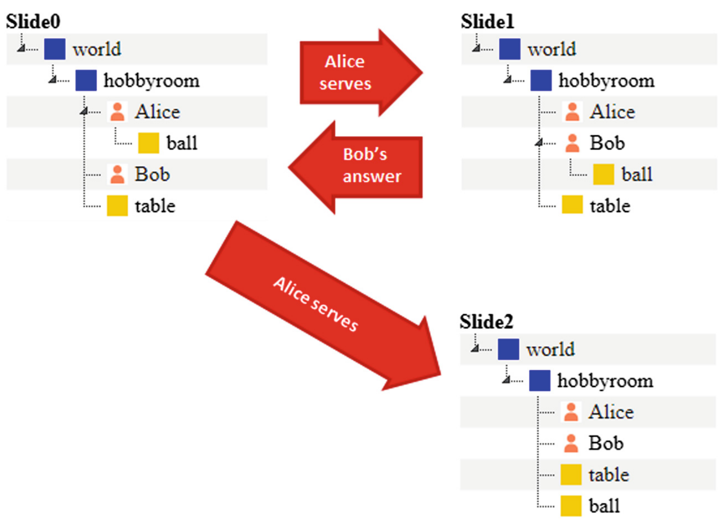


Fig. 3. The ping pong story expressed as a fable with 3 slides. Slide2 shows a world in which the ball is out and the game is over.

complete fable before starting to work on it digitally. We believe that a good digital tool should allow the author to change her mind as well as sketch partially correct fables, and only later go back and finish/fix them. So we could imagine in our example that the author of the "ping pong" fable realizes that she forgot to add a kitchen where Charlie is having tea, so she goes back to the drawings that define the fable, and draws one more room called "kitchen" in all world trees, adding a new person "Charlie" and a thing "tea", owned by Charlie. Playback of the new version of the fable results in a new sequence of world states, all of which also contain the kitchen with Charlie and the tea.

Finally, slides are temporal in nature, but from reading all people, things and rooms in all slides of a fable, we can infer a palette containing a-temporal information. In the case of the ping pong story the palette consists of Alice and Bob (as people), the table and the ball (which are things), and a hobby room (the only room defined in the fable). The palette can be seen as a summary of all elements that can appear in a slide of a fable, hence, the palette will be a valuable part of the F4BL3s editor, helping fable authors to maintain a coherent overview of people, things and rooms across slides in their fables.

3.2 Analysis

To better take advantage of natural ways of thinking and managing knowledge, we want fables to be constrained in various ways. Fables should offer a familiar metaphor, this is why we designed a fable to be a collection of slides, each slide representing the state of the world at a certain time in a story.

We want non-linearity to be present but it should not overwhelm the author, therefore a fable should be an *almost linear* story, with each slide connected to a maximum of 3 possible “next” slides. A slide with no outgoing connections is one of the endings of the fable, and loops are possible. This low degree of non-linearity is consistent with our analysis of other non-linear media, from games like Dragon’s Lair to Gamebooks [17], and cartoons like Kika & Bob[1]. Our analysis shows typical patters to describe non-linear storylines: sequence, choice, and end-of-story. Choices, i.e. points in the story when user input is required to continue, are sparse within the narrative and usually present a small number of options (usually 2 to 6, in case a dice is used to decide). In this sense, we are applying the principle of design thinking called “avoiding decision and combine best possible choices” [13, p. 336], as we are still trying to limit the number of possible options for our users. Hence, in many cases, there is a single *correct* option, while taking any of the others brings the story to an end; the end-of-story could be global, terminating the current line of narration, or local, simply bring the story back to the state preceding the choice, to allow the user to explore alternatives. In other cases, multiple options might allow the story to continue, however, the alternative paths will result in different levels of success for the user. Another constraint for a fable is that it should represent a short story. Fables should contain a small number of slides, 10 to 15 by design; with a possibility to define more complex fables by hierarchically composing simple fables together.

As mentioned in the previous section, we want to support incremental definition, so in our F4BL3s tool rooms, people and things can be defined and redefined at any moment during editing of a fable. This is more natural for authors than being forced to define everything at the beginning. However, changes that make the story inconsistent are automatically detected and signaled by the F4BL3s tool. Inconsistencies among slides (e.g. John having an apple in one slide, but there was no apple in any room, in any of the slides before) are allowed but automatically signaled, to support working with partially correct or incomplete fables (i.e. fable drafts).

We require the playback of fables to be digital and multimodal: both a visual animation and a natural-language textual representation should always be available for any fable. For example, the slides of the “ping pong” fable in Fig. 3 can easily be exported as a text in natural language, since all elements have names and actions have labels. In F3BL3s multimodality is supported also by allowing textual definition of fables. The typical work-flow in this case would be:

- Write textual description of a story, as a series of commands in a natural-language inspired DSL as in Table 1 (i.e. a Domain Specific Language [8])
- Compile the textual description and get an automatically generated sequence of slides (i.e. a fable), each with a snapshot of the state of the world at a certain step of the story
- Eventually the fable can be edited manually by redrawing slides or adding action and respective arrows to extend the storyline beyond its original textual description

The DSL language in Table 1 is very simple, and the first line is always the title of the fable. All other lines must start with a room where the story will develop (in analogy to comic books, where the narrating voice often explains where the action

[1] Official website: http://www.kikaandbob.com Last visited February 20th 2018.

Table 1. The left column describes the fable to be generated, and the right part shows the palette for the fable.

a ping pong match	**Inferred palette**
AT hobbyroom, THING table, Alice PICKUP ball	People: Alice, Bob
AT hobbyroom, Alice GIVES ball TO Bob	Things: ball
AT hobbyroom, Bob GIVES ball TO Alice	Rooms: hobbyroom

takes place), and after that a number of statements might follow. Some statements simply state the existence of a thing or a person (e.g. "THING table"), or they could relate people and things (as in "Alice PICKUP ball" or "Alice GIVES ball TO Bob"). Table 1 also shows that the compilation process turning the textual description into a fable automatically infers a palette for the fable. The decision to categorize elements of a fable into rooms, people and things, comes from our desire to base fables upon formal models like mobile ambients. In mobile ambients types are used to restrict the movements and interactions of processes [7]; here we use our 3 categories as a kind of *light typing*. The result is a set of restrictions for the elements of fable: people can be moved from room to room, but things should stay where they are, unless picked up by a person, in which case they can move with the person carrying them. Rooms cannot be moved and exist in the same relative place in the world in all slides, creating a perceptual and cognitive persistence to support authors and users of the fable to navigate (a phenomenon related to memorization techniques like the *method of loci*[2]). Moreover, in our experience we found that metaphorical naming of abstract entities often provides a way to **naturally attach roles** to those entities [20]; in the F4BL3s tool we expect that our categories will help authors and users to easily grasp which affordances are associated to the various elements of their fables.

3.3 Design

In order to implement the F4BL3s tool as a web-based application, we defined an ontology to clarify the relationship among the elements of a fable. The UML class diagram in Fig. 4 states that a fable is simply a collection of slides, and a slide contains a special room called world, and 0 to 3 actions, possibly linking the slide to other slides. The Slide and Action classes are arranged to define a graph data-structure, where slides are the nodes of the graph and actions the edges. The classes for Room, Thing and Person are connected according to the composite design pattern, suggesting that rooms act as containers for other rooms, things and people.

The composite pattern is also central in the definition of objects in object-based languages (like Javascript or Python) and ambients in mobile ambients. In fact, fables share many concepts with these two computational models, as visible in Table 2.

The F4BL3s web-based application has a modular architecture, centered on a single data-type. We started from the fable data-type (Fig. 4), and developed each part of the tool as a standalone web application manipulating the fable data-type, i.e. all webpages

[2] Mentioned by Cicero in his: "*De Oratore*", literally "On the Orator", written circa 55 B.C.

were interoperable by design. This architectural choice made it easy to experiment independently with the individual tools within the F4BL3s web application. For example, we could easily define the first fables by manually writing some data, and could immediately play them back, even if the editor page was still incomplete. The data-centric, modular architecture also made it easy to try out multiple versions of the same page/module, as needed for our user-centered development.

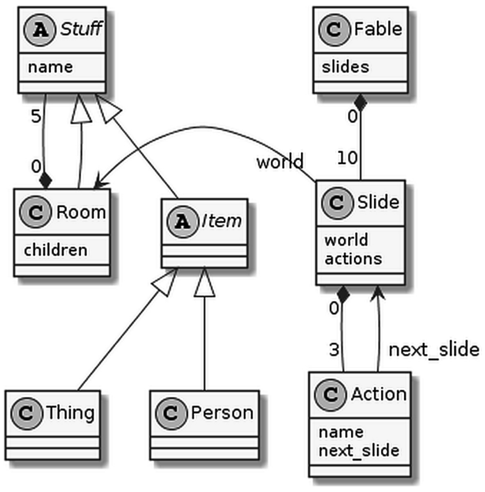


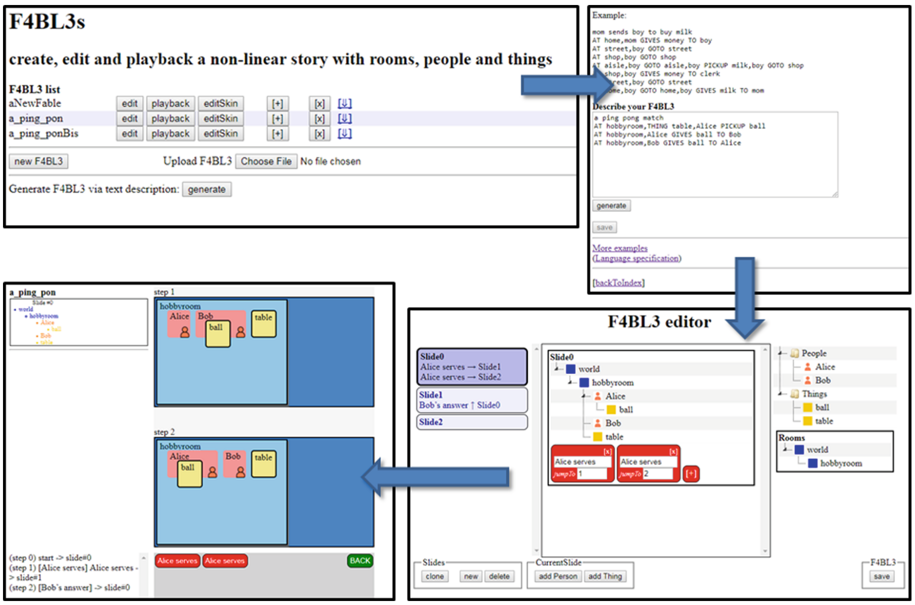
Fig. 4. UML class diagram for F4BL3s. Drawn using PlantText.

Table 2. Comparison among mobile ambients, object-oriented programming and fables' concepts. Concepts on the same line behave similarly.

Mobile ambients	OOP	Fables
ambient literal, type	class	palette of people and things
ambient	object	slide
name	attribute	room
ambient, process	value	people, thing
operation (and reduction semantic)	(short/simple) method	action, going from slide to slide

3.4 Implementation

The F4BL3s prototype is developed as a collection of dynamic web-pages, using Javascript, jQuery, and jsTree to visualize and edit the nested rooms with people and things. The jsTree library is a jQuery plugin especially designed to represent and manipulate trees, it provides many features such as drag and drop, types for the nodes of the trees and even constraints circa which types of nodes can be nested in which other types. Its adoption allowed for a rapid development of the UI in the editor



F4BL3s

create, edit and playback a non-linear story with rooms, people and things

F4BL3 list

aNewFable	edit	playback	editSkin	[+]	[x]	[⇓]
a_ping_pon	edit	playback	editSkin	[+]	[x]	[⇓]
a_ping_ponBis	edit	playback	editSkin	[+]	[x]	[⇓]

new F4BL3   Upload F4BL3 Choose File No file chosen

Generate F4BL3 via text description: generate

Example:

```
mom sends boy to buy milk
AT home,mom GIVES money TO boy
AT street,boy GOTO street
AT shop,boy GOTO shop
AT aisle,boy GOTO aisle,boy PICKUP milk,boy GOTO shop
  shop,boy GIVES money TO clerk
    reet,boy GOTO street
     me,boy GOTO home,boy GIVES milk TO mom
```

Describe your F4BL3

```
a ping pong match
AT hobbyroom,THING table,Alice PICKUP ball
AT hobbyroom,Alice GIVES ball TO Bob
AT hobbyroom,Bob GIVES ball TO Alice
```

generate

save

More examples
(Language specification)

[backToIndex]

F4BL3 editor

Slide0
Alice serves → Slide1
Alice serves → Slide2

Slide1
Bob's answer ↑ Slide0

Slide2

a_ping_pon

(step 0) start -> slide#0
(step 1) [Alice serves] Alice serves -> slide#1
(step 2) [Bob's answer] -> slide#0

Fig. 5. Navigation diagram of the F4BLEs web application. The blue arrows show a typical workflow: generation of a fable, editing and playback. (Color figure online)

(bottom-right screenshot in Fig. 5), which required the most of our development time so far.

Figure 5 shows a navigation diagram of the 4 main pages of the F4BL3s web application. On the top-left, the main page that works as a dashboard, listing all fables created up to now. Fables can be created, cloned, deleted, downloaded and uploaded from this main page. The current prototype does not allow multiple authors/users, and stores all data locally (using the browser's local storage), however, it should be easy adding a simple user login page and switching to a web-hosted database (like for example mLab's MongoDB) in a few iterations. The screenshot on the top-right of Fig. 5 shows the page for generating a fable from a textual description. Once a fable is created (directly from the main page or via textual generation) it can be opened in the editor (bottom-right part of Fig. 5). The editor page, the most complex page of F4BL3s, is implemented around the MVC pattern, where each fable is a tree of Javascript objects modeled according to the UML diagram in Fig. 4. Operations on the fable data are implemented separately from the data-structure, to simplify JSON-serialization across web-pages. The playback page is visible in the screenshot in the bottom-left part of Fig. 5. The visualization of the slides is different from that used in the editor page: in playback slides look as much as possible like sticky notes, with generic icons and color-coding for rooms, people and things. The playback page lets the user navigate through the slides, by choosing among the (potentially) multiple actions available from each slide. It also offers a rewind button, labeled "back", that allows the user to go back one step and chose an alternative action.

We explored various visualization styles for the slides, which are shown as trees and as sticky notes in Fig. 5, and for the final version of the tool we would like the editor and playback page to have a more uniform visualization, and that the overall visual style is more close to stacked sticky notes than trees.

4 Testing

4.1 Organization of the Tests

We plan to test our concept and the web-based tool with teachers and primary school pupils during the spring semester of 2018. However, we believe that having a minimum viable product is necessary in this project, mainly because fables are rather abstract and theoretically based, so they could be hard to present and discuss in participatory workshops. Hence, the idea of developing an early prototype, designed with the information gathered in previous experiences with school cooperation (such as [2]).

We organized a task-based testing of the F4BL3s prototype in the fall semester of 2017, involving 8 of our bachelor students, all males. They are part of the engineering education in Learning and Experience Technology at the University of Southern Denmark (SDU), hence they know programming and they are experienced in developing e-learning applications together with local schools, teachers and pupils. For us it was a good opportunity to perform usability tests and interview them as experienced users, to collect early impressions and constructive feedback about the prototype and on the idea of fables itself. These students represent a significant sample of our user population, as we aim at developing fables as a metaphor for programming games.

The test was organized at our campus and we divided the students in 3 groups, of 4, 3 and 1 students respectively. The student who took the test alone decided to perform the test from home and write reflections by email. The students who took the test in class were gathered as a focus-group [21] and were sitting with their laptop. The test was conducted as a qualitative evaluation, supported by ethnographic observations and semi-structured interviews, supported by content analysis on video recording gathered during the whole test [22]. In analyzing the videos, we focused our attention on the students' facial expressions and interaction with the system, at the same time we looked at what they were doing on their laptops and took notes on the actions that we considered more interesting in relation to how they were editing their fables. The test started with a 5-min presentation of the fables project, the concept of developing a tool to support non-programmers (as for instance teachers and pupils) in creating their own digital games/simulations, and how fables relate to theories from Schön and Simon. After that the participants were given a few tasks to try out with the tool, starting with generating a fable from a textual description, followed by editing and playing back of the same fable; the last of the tasks was to freely expand the fable, customize it, and eventually alter the sequence of the slides. About thirty to forty minutes were used for the tasks. The last step in the test was to conduct a focus-group semi-structured interview, aimed at discussing problems with the current prototype and collect opinions on the idea of fables in general, as well as suggestions circa the areas of applicability of our tool. The interview took ten to fifteen more minutes. Altogether the tests took roughly one hour for each group.

The first group went through the test exactly as describe above, and none of the participants talked until they finished all tasks, and we got very positive feedback from them afterwards. The second group was less focused on the tasks, and the participants were very quick in getting the point of fables, so they immediately started discussing pros and cons of the prototype, as well as brainstorming on the domains in which they would have like to use fables, or in which they could imagine pupils and teachers using them. The last student, who participated asynchronously and remotely, focused on the usability of the editor, but reported having used circa the same time as the participants physically present at the test session.

The test sessions of the first 2 groups were recorded on camera, and the following interviews were also audio recorded, so we could later perform content analysis and cluster the topics of discussion. The student who participated asynchronously also documented his reflections and sent the data to us via email. In such case, data were gathered applying content analysis on their emails.

4.2 Results and Discussion

Analysis of the test results showed 3 categories in the discussion and feedback: domains of applicability, UI issues, and more general reflections circa the concept of fables.

The first domain suggested was English teaching in primary school, as a foreign language; the argument for that was that F4BL3s are multimodal, and the use of visual and text together should support learning associations between words in another language and images representing the real-world objects. Moreover, fables can easily express typical situations, such as a family at the dinner table, talking about food and kitchenware, which are often used when teaching English to pupils. Another domain that was mentioned was natural sciences. Our students imagined a scenario where a pupil shows what she knows about a forest (for example) by making one as a fable, with animals eating, living, reproducing and dying, plants of various kinds, different rooms representing habitats. The playback of the forest fable can be shown to other pupils for reflection or to a teacher for assessment. In this scenario F4BL3s could be used in class as an interactive alternative to sequential presentations with slides. Some of our students from the second group proposed using fables to represent historical events, or everyday situations from history, like for instance Viking raids. In fact, one of the students started making an example of a story in a Viking village with F4BLEs and that in turn led to interesting reflections on the relation among fables and OOP, where classes and objects are often used to represent the state and evolution of real-world situations. Finally, a student thought of groups playing RPG: fables could help creating the plot for a session of an RPG game, the plot can then be explored digitally using the playback page in F4BL3s.

With respect to the user interface, we received various comments. A few student lamented problems with deleting/inserting slides (in the editor page). We had an interesting discussion about rooms and how changes in them are always automatically reflected in all slides, while adding or changing a thing only affects a single slide at the time. We are happy to see these issues emerging, since fables can be seen as semi-structured data, and when reasoning formally on such structures it is common to use *temporal logic*. For instance, in our ping pong example, the author might decide that Alice had a bag with something inside the whole time as she played ping pong with Bob, and it should be possible to add a bag *thing* to the *person* Alice in all slides, i.e. globally, at all times. To us that suggests that fables could offer a visual way to simplify understanding and thinking in terms of temporal logic, for instance in high-school or university Computer Science courses. We were also glad to observe that none of our participants had problems understanding and taking advantage of the palette.

We collected general reflections on fables in a specific category. A student stated that F4BL3s "feels like OOP in many ways, where slides are a bit like objects, but more visual". Another said: "Nice that is all very visual, but it could actually be even more like a mobile app, with everything drag and drop". There was a request to add an export functionality. The argument was that it would be useful to export the fable as a Scratch program, so that children could use it at school; many primary schools in Denmark have sessions of Scratch programming, or similar visual programming languages. A participant in the first group remarked that if fables could be exported towards other programming languages (e.g. C#) our prototype could effectively be used as a prototyping tool for programmers. Interestingly, these comments seem to point at the need to bridge between the activity of designing and programming games, hence we interpret this comment as suggesting that the students grasped the design thinking aspect of our tool. We find this feedback very encouraging; especially in light of the fact that we are considering an application of fables to the description of game mechanics, for rapid development of simple games to be used in local schools. One of the participants in the second group has relatives who are primary school teachers, he confirmed that most teachers (and pupils too) in his experience are familiar with web and android tools like google slides and Microsoft PowerPoint. This is in line with our data showing that in Denmark many primary schools have laptops, tablets, and often iPads, that can be lent to pupils during classes. Therefore, he argued that the F4BL3s web tool might work better for schools if the editor was re-implemented and made more similar existing presentation tools.

5 Conclusion

We are looking at natural ways of expressing behavior to create digital simulations, for primary school teachers and their pupils. Looking at Simon's simulations and Schön's repertoire of exemplars, we propose a novel approach where authors define a particular kind of non-linear story, using both a text description and/or direct manipulation of people, things and rooms. These stories, called fables, represent Schön exemplars. In line with our previous research, teachers and pupils can *transpose* knowledge from a domain by defining a fable, as well as store, share and demonstrate their fables digitally. In fact, without the need of coding, a fable automatically plays back as a digital interactive simulation of a domain. The concept of *transposition* is discussed in [18].

Results from early tests with a web-based prototype (called F4BL3s) are encouraging. Our students could quickly grasp the idea of fables and could reflect on the implications of defining, editing and simulating real-world situations in primary schools. We consider the test results mainly from a usability standpoint, because the participants are engineers, with programming skills and are used to work and even develop e-learning tools. However, we wanted some validation of our tool before starting the actual user-centered development targeted at teachers and pupils.

We expect fables to be usable in various domains, but the participants to our early test spontaneously proposed even more domains and scenarios of use. Similarities between fables and OOP were highlighted, and we were happy to observe that fables enabled participants to engage in temporal thinking, providing a terminology and visual way to discuss changes in people, things and rooms over time.

We concluded that despite presenting issues with the UI and some functionalities, our prototype is already usable enough to be presented to teachers and school children in coming participatory workshops; moreover, we expect the fables should make sense to them, especially if presented starting from examples grounded in a concrete subject or domain. We are currently organizing a series of participatory workshops to be held in spring 2018, involving local school teachers and their classes, to discuss, co-develop and test the next iterations of our prototype of the F4BL3s web application.

References

1. Marchetti, E., Valente, A.: It takes three - re-contextualizing game-based learning among teachers, developers and learners. In: Connoly, T., Boyle, L. (eds.) Proceedings of the European Conference on Games Based Learning, pp. 399–406. Academic Conferences International (2016)
2. Valente, A., Marchetti, E.: Digital game development for kids, mediated by board game inspired paper prototypes. Accepted for publication, IEE: Information Engineering Express, IIAI (2017)
3. Aivaloglou, E., Hermans, F.: How kids code and how we know: an exploratory study on the Scratch repository. In: Proceedings of the 2016 ACM Conference on International Computing Education Research, pp. 53–61. ACM (2016)
4. Simon, H.A.: The sciences of the Artificial. MIT press, Cambridge (1996)
5. Schon, D.A.: The Reflective Practitioner: How Professionals Think in Action. Ashgate, Aldershot (1986)
6. Klein, G.: The fiction of optimization. Gigerenzer, G., Selten, R. (eds.) Bounded Rationality: The Adaptive Toolbox, pp. 103–121, Dahlem Workshop Reports (2001)
7. Sangiorgi, D., Valente, A.: A Distributed Abstract Machine for Safe Ambients. In: Orejas, F., Spirakis, Paul G., van Leeuwen, J. (eds.) ICALP 2001. LNCS, vol. 2076, pp. 408–420. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-48224-5_34
8. Fowler, M.: Domain-Specific Languages. Pearson Education, Boston (2010)
9. Marchetti, E., Valente, A.: Quiz-R-Us – Re-conceptualizing quizzes to enrich blended learning in occupational therapy study lines. In: The International Conference of Human Computer Interaction, Lecture Notes in Computer Science, Springer (2018, accepted for publications)
10. Cardelli, L.: Abstractions for mobile computation. In: Vitek, J., Jensen, Christian D. (eds.) Secure Internet Programming. LNCS, vol. 1603, pp. 51–94. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48749-2_4

11. Love, S., Gkatzidou, V., Conti, A.: Using a rich pictures approach for gathering students and teachers digital education requirements. In: Little, L., Fitton, D., Bell, Beth T., Toth, N. (eds.) Perspectives on HCI Research with Teenagers. HIS, pp. 133–149. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-33450-9_6
12. Valente, A., Marchetti, E.: Development of a Rich Picture editor: a user-centered approach. Int. J. Adv. Intell. Syst. **3**(3, 4), 187–199 (2010)
13. Razzouk, R., Shute, V.: What is design thinking and why is it important? Rev. Educ. Res. **82** (3), 330–348 (2012)
14. Björgvinsson, E., Ehn, P. Hillgren, P.A.: Participatory design and "democratizing innovation." In: Proceedings of the Participatory Design Conference, pp. 41–50. ACM (2010)
15. Könings, K.D., McKenney, S.: Participatory design of (built) learning environments. Euro. J. Educ. Editor. **52**, 247–252 (2017)
16. Möllerup, P.: Simplicity: A Matter of Design. BIS Publishers, Amsterdam (2015)
17. Costikyan, G.: Where stories end and games begin. Game Dev. **7**(9), 44–53 (2000)
18. Marchetti, E., Valente, A.: Learning via game design: from digital to card games and back again. Electron. J. E-learn. **13**(3), 167–180 (2015)
19. Henriksen, D., Mehta, R.: A beautiful mindset: Creative teaching practices in mathematics. J. Math. Educ. **9**(2), 81–89 (2016)
20. Marchetti, E.: If it looks like a duck. Names as shared signifiers for discussing "cuteness" in healthcare robotics. In: The 9th International Conference on Multimodality, University of Southern Denmark, Odense, Denmark (2018, submitted)
21. Preece, J., Sharp, H., Rogers, Y.: Interaction Design. Beyond Human Computer Interaction. Wiley, New York (2015)
22. Krippendorf, K.: Content Analysis. An Introduction to its Methodology. Sage, Thousand Oaks (2004)