Chapter 5

The Story Maps Application

The exploratory study using a paper-and-card interface, reported in chapter 4, has confirmed the adequacy of Propp cards and story maps as interactive representations of Propp's functions. Based on this research a first computational version of the tool can now be developed for use in further studies. The development and verification of an initial story authoring application will lay the foundation for further development of the application, exploring more advanced features.

The design of the story maps application was based on the paper and card story writing tools used in the previous exploratory study and on the researcher's observations of participant's interactions with these tools. These design foundations are described in section 5.1. Section 5.2 specifies the formal requirements for the application. Section 5.3 describes the implementation and evaluation of a prototype that was designed to meet these requirements. A final version of the application was arrived at following several significant and many minor changes to the prototype in response to evaluations with users. This final version meets the requirements specified, and was used in the experimental study described in chapter 7. The final version of the story maps application is presented in section 5.4.

5.1 Foundations for Design

In the exploratory study participants constructed story maps by selecting the Propp cards they wanted to use from a set arranged on a table, taking their chosen cards back to their own writing table and arranging them as desired to form their story map. Participants were asked to keep their chosen Propp cards in the order determined by the overall sequence of the Propp cards. Participants wrote their stories by writing

the corresponding part of their story on each of their Propp cards. At any time during story writing participants were allowed to return Propp cards to the card table, take new Propp cards from the card table for their story map, or reorganise the Propp cards in their story map. Participants did not have to finish constructing their story map first and then write their story but could interleave the two processes.

The basic interactive objects in the story maps design are the *Propp cards*, which are analogous to the paper Propp cards used in the exploratory study; the *card store*, which is analogous to the card table; and the *story map* which is analogous to the story maps constructed by participants in the exploratory study. The design aims to recreate the process of constructing story maps and writing stories using paper Propp cards on a table. To this end, the *zooming interface paradigm* is adopted to tie together the three basic interactive objects.

5.1.1 The Zooming Interface Paradigm

The paper Propp cards arranged on a table that were used in the exploratory study (whether on the card table or in the user's story map on her own table) have some features that the story maps application aims to reproduce:

1. The user can easily take an overview of all of the cards available and their arrangement, but she can also choose to focus on a particular card or group of cards that she is currently working on by taking a closer look at them.

2. By moving her cards around on her table, the user can arrange them into formations and groupings with her own semantics. For example, arranging the cards that she intends to use in her story map in one group and those that she has already positioned in the ordered story map in another, or placing the cards she has finished with in one group and those she still has to do in another.

3. With enough table space the user can arrange a large number of Propp cards on the desk in front of her. It would not normally be possible to display the same number of Propp cards on a typical-sized computer screen, while still keeping each card large enough for its text and images to be legible.

The *zooming interface paradigm* (ZIP) (Raskin, 2000, p.152) was adopted for the story maps application because it best reproduces these features of the paper and desk interface. The ZIP is based on the idea that the user has access to "an infinite plane

of information having infinite resolution" (Raskin, 2000, p.153). Everything that the user can access is displayed on this plane, which is called the *zoom world*. Using the mouse and keyboard the user controls the viewport through which the zoom world is displayed on her screen by panning the viewport left, right, up or down, and zooming in and out. The navigation metaphor is one of "flying" by "climbing" to zoom out and "diving" to zoom in. To see more of the zoom world the user can fly higher and higher above it, while to look at a particular item in the zoom world she can dive down into it. Raskin (2000, p.153) describes a zoom world as being "conceptually similar to the walls of a project planning room ... covered with annotations, tacked-up sheets of paper, sticky notes, photos, or whatever else." Diving and climbing emulates what a person does in a planning room: "stepping back to view large areas, then walking up to just the one that is desired, and finally leaning forward to read the fine print or using a magnifying glass to see a detail in a photo" (Raskin, 2000, p.154). This form of navigation takes advantage of our inherent tendency to remember relative position and landmarks: the user can always see her goal (an object that she wants to get to) and the path to it (by diving into it), and continuously animated diving and climbing preserves the user's sense of position so she knows how to get back to where she has been (Raskin, 2000, p.154).

The zoom world metaphor fits nicely with the paper-and-desk interface. For the purposes of the story maps application the zoom world can be thought of as conceptually similar to the user's desk space, covered with her used and unused Propp cards, pieces of writing paper, etc. The zooming interface paradigm offers a simple and flexible foundation so that the finished application should have potential for expansion and modification in the future.

5.1.2 Propp Cards

A Propp card is a card that both represents a Propp function and, if the user has used the Propp card in her story map, contains the text for the part of the user's story that corresponds to that Propp function. At a minimum a Propp card displays a single word, the name of the function that it represents. A Propp card may also display an illustration of its Propp function, a longer description of its Propp function, the number of its Propp function in the overall sequence of functions, and further information such as a list of example realisations of its Propp function from Propp (1968) or elsewhere. Each Propp card contains an editable text area on which the user can write some notes

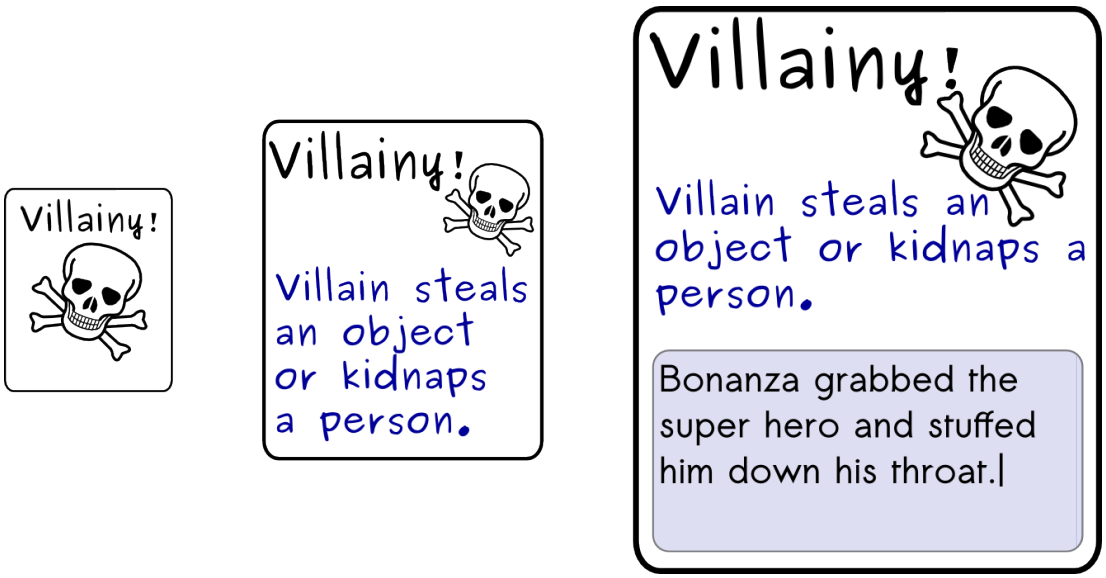


Figure 5.1: A mockup of a Propp card at three levels of semantic detail. From left to right: when viewed at a distance only the name and illustration can be seen on the Propp card; as the user dives in closer more descriptive text on the card is revealed; as the user dives still closer the Propp card's editable text area appears.

or part of the text of her story.

Semantic zooming is a common design pattern in ZIP interfaces: an object changes its visual representation according to the scale it is being viewed at. The object can remain legible and recognisable when viewed from afar, and as the user zooms in on the object more details are revealed as the screen area available to the object makes it possible for the object to display those details legibly. Figure 5.1 illustrates semantic zoom applied to a Propp card. Using a low level of semantic detail a large number of Propp cards can be displayed on screen at once while remaining legible and recognisable. When the user dives into a card or group of cards for a closer look more of their semantic detail is revealed.

The user can move Propp cards around in the zoom world by dragging and dropping using the mouse, emulating the ability of participants in the exploratory study to pick up paper Propp cards and move them around. When the user moves the mouse pointer over a Propp card the card should be highlighted (e.g. by brightness reversal, colour change, applying a border to the object, etc.) to indicate to the user that the application recognises that she is pointing to the card and that the card is an interactive object.

5.1.3 The Card Store

The card store is a container for the set of Propp cards available to the user for use in her story map. The card store initially contains a Propp card for each Propp function represented in the application. The Propp cards are displayed in their correct order according to Propp (1968). The user can remove Propp cards from the card store and place them into her story map, and can remove Propp cards from her story map and return them to the card store. The cards remaining in the card store retain their overall order as cards are removed and returned by the user.

5.1.4 The Story Map

The story map is an ordered sequence of Propp cards that represents the plot structure of the user's story. The story map is initially empty. The user can move Propp cards from the card store to the story map and vice-versa, and can move Propp cards around within the story map. The arrangement of Propp cards within the story map is determined by the user.

Figure 5.2 on the following page is a mockup of some Propp cards and a story map in the zoom world. From the highest vantage point the user has an overview of all of the Propp cards available to her and their arrangement. The user has arranged the cards that she intends to use in her story map into a cluster in one part of the zoom world. By diving into it she can focus on her story map and inspect it more closely. By diving in further she can focus on a single Propp card at a time. The user can then climb out again to get an overview of her story map or of the zoom world as a whole. Diving and zooming emulates the user's ability when using paper Propp cards on a desk to take in an overview of her Propp cards or focus on a particular card or group of cards at a time.

5.2 Requirements & Constraints

This section specifies the formal requirements and constraints for the implementation of the story maps application. The application should be usable in a study that repeats the process used in the exploratory study in which participants aged 7-11 constructed story maps and then used the story maps to write stories, and the stories written by the participants were collected as data. The requirements for the application can be divided into three categories: functional requirements, data requirements, and usability

Figure 5.2: A mockup showing the story maps zoom world viewed at three different scales. From left to right: 1. An overview of the zoom world shows many Propp cards scattered around, a group near the bottom right are arranged into a grid. 2. The user dives into this grid, which contains the cards she has selected for her story map. 3. The user dives in further to inspect a single card from her story map.

requirements.

5.2.1 Functional Requirements

The functional requirements describe the tasks and sub-tasks that the user must be able to perform with the application. The functional requirements for the story maps application are as follows:

1. Construct a story map.
   (a) Get an overview of the Propp cards currently in the card store and in the story map.
   (b) Inspect any Propp card in the card store or story map in closer detail.
   (c) The user should be able to change from an overview of the Propp cards to a detailed view of a particular Propp card and back at any time.
   (d) Move a Propp card from the card store into the story map.
   (e) Remove a Propp card from the story map, putting it back into the card store.
   (f) Arrange Propp cards within the story map into the desired order.
2. Write a story for a story map.
   (a) Select a Propp card to write on.

   During the exploratory study most participants were observed writing the text for the first Propp card in their story map, and then moving onto the

second Propp card, and so on in a linear style. This style should be supported by allowing the user to easily move to the next or previous Propp card in her story map.

A linear approach to story writing should be enabled but not enforced, at any time the user should be able to move from the currently selected Propp card to any other Propp card in her story map and edit her text on that card.

(b) Add and edit text on the selected Propp card. The amount of text should be able to vary from one Propp card to the next, from a single line of text to multiple paragraphs.

Basic text editing functions should be supported: adding, deleting and editing text, selections, cut, copy, paste, and breaking text into paragraphs.

More advanced text editing functions such as a dictionary, spell checking and grammar checking, and style controls (headings, italics, bold text, lists, etc.) should *not* be present. Such features are undesirable because the story maps application must be as simple as possible so that children using the application for the first time can learn to use it very quickly, and because the intent of the application is to focus the user's attention on story structure and content and not distract them with details such as spelling and text styling.

3. Both linear and iterative approaches to story creation should be supported. The user should be able to take a linear approach, planning her story by constructing her story map first, and then proceeding to write her story once her story map is finished.

   The application should enable but not enforce a linear approach. The user should also be able to take an iterative approach, working a little bit on her story map, then doing some writing, then going back to modifying her story map, and so on.

4. Save a story map and open it later to continue working on it.

   It may happen that some participants are not able to finish the stories in the time available. If this happens they should be able to come back to their stories and finish them later. The complete state of the user's story map must be able to be saved and restored: the set of Propp cards in the card store, the set of Propp cards in the story map and the positions of the cards in the story map, any text that the

user has added to any of the cards in the story map or card store, and any title that the user has given to her story.

5. Export a story into a format suitable for editing in other common applications (e.g. a word processor).

   This will allow the application to be used as a planning and drafting tool for a story that can later be edited in a more traditional writing application such as a word processor, with support for the more advanced word processing features that are not provided by the story maps application.

6. Print a story.

   This will allow the application to be used as a planning and drafting stage for a story that will then be written out on paper, and will allow users to take home copies of their completed story maps.

5.2.2 Data Requirements

The data requirements define the meaning and structure of data that need to be represented within the application. The data requirements for the story maps application are as follows:

1. The set of Propp's functions.

   The application will create a Propp card for each Propp function that is represented in the application. For each Propp function there should be a textual name and description of the function and an illustration of the function. Additionally, the sequential ordering of the Propp functions should be defined.

2. The Propp cards.

   The set of Propp cards represented in the application is initially generated from the set of Propp functions. Each Propp card has an associated Propp function, an associated text field that contains any text that the user has added to the card, and a location in the card store or story map.

3. The card store.

   The application will keep track of the set of Propp cards currently present in the card store.

4. The story map.

   The application will keep track of the set of Propp cards currently present in the story map and the position of each card within the story map.

5. Logging.

   For research and development purposes the application will keep a log of application state (Propp cards, card store and story map) and the sequence of user interactions that have operated on the application state.

5.2.3 Usability Requirements

The usability requirements focus on the usability of the application, the characteristics of the intended users and the environment in which the application is intended to be used. The usability requirements for the story maps application are:

1. The application is intended to be used by school children aged 7-11. The users will have experience with computers from school and some of them will have computers at home. It can be assumed that the users will know how to use a mouse and keyboard, and will be familiar with common interactions such as selecting an object by clicking on it with the mouse, and moving an object by dragging and dropping with a mouse.

2. To be used by children the application should have a simple interface that is easy to operate, and any images or texts used in the application should be able to be understood by children.

3. The application is primarily intended to be used in a school environment, secondarily at home.

   The researcher should be able to bring the application into a school on a USB thumb drive and run it on the school's computers, with minimal requirements. Since school environments vary, the application should be usable with different operating systems (although most British primary schools use some version of Microsoft Windows), and at different screen resolutions (in the researcher's experience, primary school computers usually run at a much lower screen resolution than is common elsewhere). The application should not be demanding in terms of the software that a computer must support in order to run the application, most software packages are not likely to be installed on school computers

and the computers are usually locked down so that teachers or the researcher will not be able to install additional software. Software that runs over the internet should also be avoided as school computers usually have restricted internet access. Finally, the application must not require high hardware specifications of the computers that run it, such as memory, processor speed or graphics hardware, because school computers are unlikely to meet these specifications.

If the application meets the above demands of a school environment, then it is likely to be usable on the home computers of pupils, teachers and other users as well.

4. The application should be learnable by the child users with not more than fifteen minutes demonstration and practice. The purpose and operation of the application should be quickly recognisable, and the navigation and interactions within the application as simple as possible. Access to child participants and to computers in primary schools (where a single computer laboratory is usually time-shared between all classes) is likely to be time limited, whether for a research project or for lessons conducted by teachers.

5. The application should be efficient to use, with high throughput and few user errors, allowing the user to concentrate on the task at hand as directly as possible, keeping the time required to complete the task to a minimum.

   It is particularly important to prevent the possibility of any application or user errors resulting in loss of user content. It should not be easy for the user or the application to unintentionally destroy content created by the user, and changes to content should be revertible. Data should be collected and user content saved automatically by the application, without requiring the user to save their content and therefore allowing the user or a software error to close the application or destroy the content without saving it first.

6. The application should be pleasant to use, engendering a positive attitude in the user and motivating the user to complete the task.

5.3 Prototype Implementation & Evaluation

5.3.1 Implementation

The first prototype of the story maps application was implemented in the Python programming language using the graphics rendering engine Panda3D.[1] To create the prototype a zoomable interface framework was developed for Panda3D. The story maps prototype itself was then built on top of this framework. The code was designed in a generic and flexible way to allow new features and different variations of the prototype to be added quickly.

5.3.1.1 The Opening Screen

Figure 5.3 on the next page is a screenshot of the opening screen that greets the user on launching the prototype application. In the top half of the screen is the card store containing the set of available Propp cards. The title and illustration of each Propp card is visible, and these elements are scaled to fill the space available to the card so that they are as legible as possible. In the bottom half of the screen is the story map, initially empty, represented as a grid of boxes into which Propp cards from the card store can be dragged and dropped.

The user is welcomed with a passive, transparent overlay message that provides the minimal instructions necessary to begin using the application: “Left-click to zoom in. Middle-click to drag.” The message fades in, remains for five seconds, then fades out and is gone. Passive overlay messages of this sort are used whenever the application needs to inform the user of something.

From this opening screen the user can move the mouse pointer over any of the Propp cards, which will react by animating themselves, scaling to 110% of their original size as the mouse pointer moves over them and shrinking back down to their original size when the mouse pointer leaves.

5.3.1.2 Moving Propp cards

Using the middle mouse button, the user can drag-and-drop one Propp card at a time from the card store above to the story map below. The empty boxes of the story map do not react to normal mouse-overs because no action is available, but when a Propp

[1] Panda 3D is free and open source, available from `http://www.panda3d.org`.

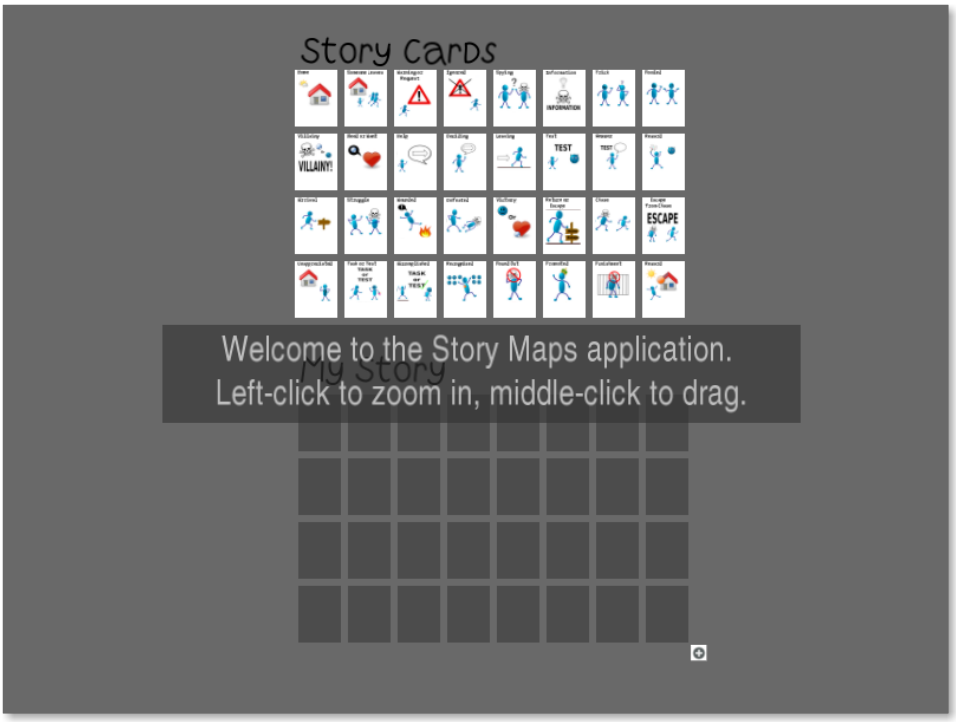


Figure 5.3: The opening screen of the prototype story maps application.

card is dragged over an empty box the box highlights itself to indicate that the Propp card can be dropped onto the box.

If a Propp card is dropped onto a box it snaps to the location of that box, if it is dropped onto nothing it snaps back to its original location in the card store. When a Propp card is moved from the card store to the story map a passive, faded image of the card is left behind in the card's original location in the card store (see figure 5.4 on the facing page). This means that the user can see where a card she has moved came from, and when inspecting the remaining cards in the card store she can which cards used to be between them so that the sequence of the Propp cards (which contributes to their meaning) is never broken in the card store.

A Propp card can be dropped onto any box in the story map, and once in the story map cards can be rearranged by dragging and dropping them between boxes to achieve any arrangement the user wants (see figure 5.5 on the next page). As before, if a card is dragged out of the story map and dropped onto nothing, it simply snaps back to its original position in the story map.

Propp cards are never deleted, they always return to their original location in the card store from where they can be retrieved again if wanted.

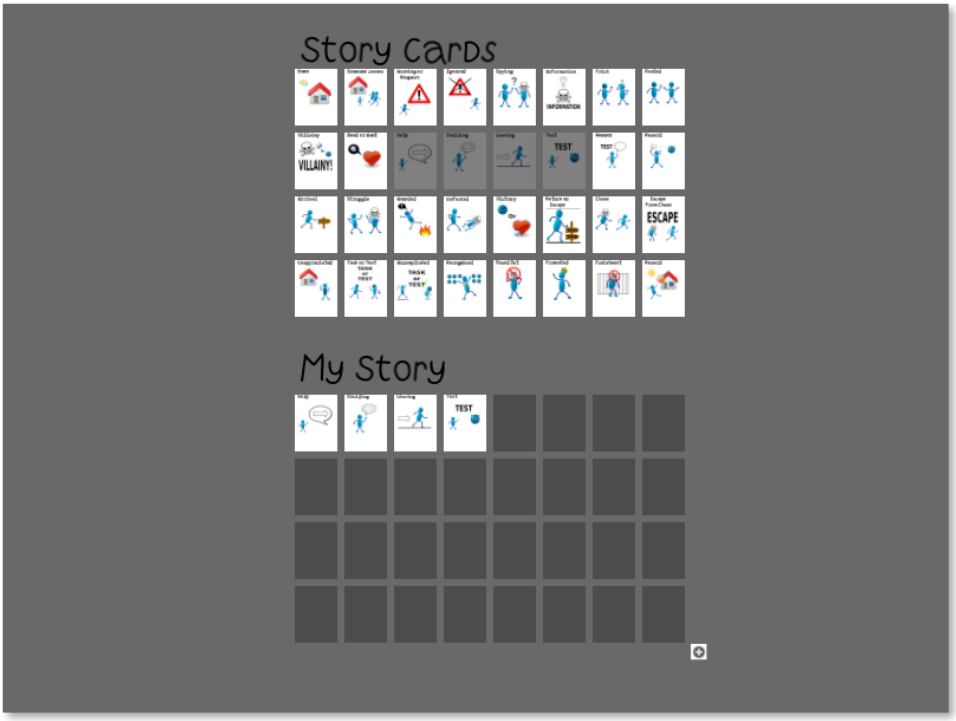


Figure 5.4: The opening screen of the prototype after the user has dragged some Propp cards from the card store (above) into the story map (below). A faded image of each card remains at the card's original location in the card store.

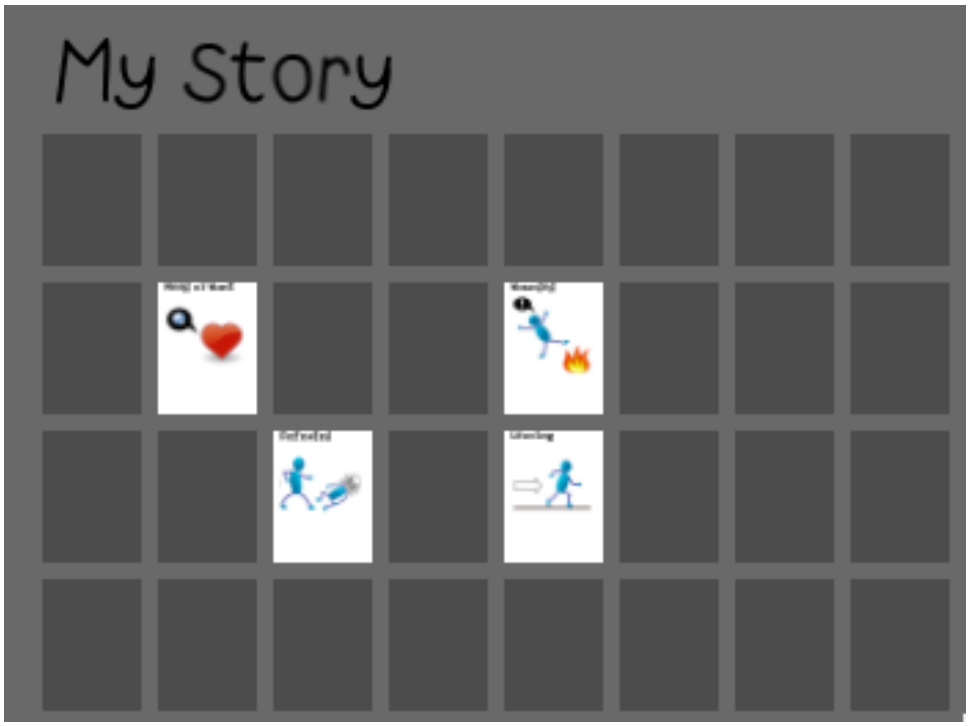


Figure 5.5: A screenshot of the prototype story maps application showing that within the story map Propp cards can be arranged into any formation that the user chooses.

5.3.1.3 Entering the Story Title

By left-clicking on the text ‘My Story’ above the story map and then using the keyboard, the user can replace the title with the name of her story.

5.3.1.4 Zooming In to Focus on a Propp Card

By positioning the mouse pointer over any Propp card in the card store or story map and left-clicking the user can zoom in on that Propp card for a closer look. Clicking the right mouse button will cause the view to zoom out again (the first time a user zooms in on a Propp card an overlay message reading “Right click to zoom back out” appears for five seconds, then fades out). All zooms and pans are smoothly animated so that the user’s sense of position is preserved.

Figure 5.6 on the facing page is a screenshot of the prototype when the viewport is focused on a Propp card in the card store. When focused a Propp card is revealed in greater semantic detail, the title and image no longer fill the card but take up only a small area, and a longer description of the card’s Propp function and a button labelled ‘+’ appear on the card. The eight cards surrounding the focused Propp card remain active, they respond to mouse-overs and if the user left-clicks on one of them the camera pans to focus on that card. In this way the user can move from card to card within the card store or story map without having to zoom out and zoom in again.

5.3.1.5 The Stack

Figure 5.7 on page 158 is a screenshot of the story maps application when ‘the stack’ is in view to the right of the screen. The stack is a temporary holding area for Propp cards that moves with the viewport as it zooms in and out and pans around. When the viewport is focused on a card in the card store the user can pan the viewport from card to card within the card store, but she cannot drag a card to the story map without first zooming out to bring the story map into view. The stack was added as a convenient place for the user to store several cards that she is interested in from the card store without having to zoom out to drag each card to the story map. The ‘+’ button on a Propp card in the card store adds the card to the stack. Cards can also be dragged to and from the stack. The stack is always available whether the viewport is zoomed out or is focused on a card in the card store or story map. The stack slides into view whenever the user moves the mouse pointer to the right edge of the screen, and slides out of view again when the pointer moves away.

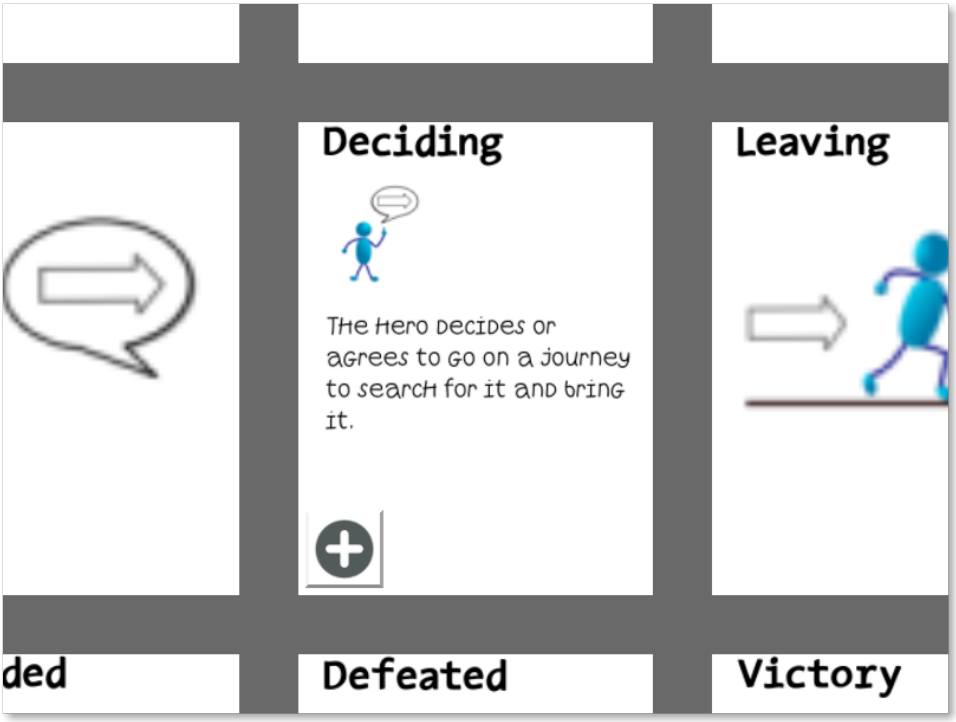


Figure 5.6: A screenshot of the prototype story maps application with the viewport focused on a Propp card in the card store. When focused a Propp card is revealed in greater semantic detail, with a longer text description of the card's Propp function and a row of buttons for acting on the card. The user can click on any of the eight adjacent cards and the viewport will pan to focus on that card instead.

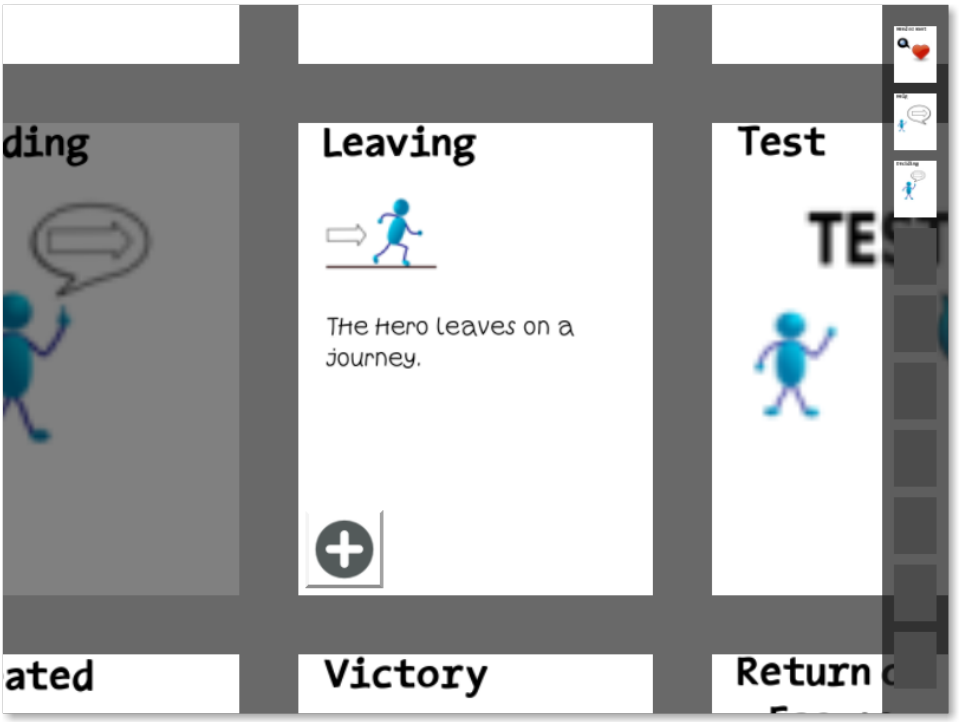


Figure 5.7: A screenshot of the prototype story maps application with the stack visible on the right side of the screen, holding three Propp cards.

The stack operates similarly to the clipboard in a traditional desktop environment, but instead of holding only one object at a time (with each new object that is added overwriting any currently held object) the stack can hold up to ten Propp cards at once. If the user tries to add a card to the stack when the stack is already full an overlay message asks the user to drag some cards out of the stack first. This approach avoids the possibility of a Propp card in the stack being unintentionally deleted when another Propp card replaces it in the stack, as can happen with a traditional clipboard.

5.3.1.6 Writing on Propp cards in the Story Map

Figure 5.8 on the facing page is a screenshot of the prototype application with the viewport focused on a Propp card in the story map. When the viewport focuses on a Propp card in the story map the view is slightly different to when focused on a Propp card in the card store. The text on cards in the story map can be edited with the keyboard, so that the user can fill in her own story by replacing the text on each card in her story map. A focused card in the story map has a row of three buttons along its bottom edge, from left to right: *copy* and *paste* (the common text editing functions) and *remove*. *Copy* and *paste* operate on the editable text of the card and allow text to

Figure 5.8: A screenshot of the prototype story maps application with the viewport focused on a Propp card in the story map. The Propp card contains an editable text area, copy and paste buttons, and a '-' button that removes the card from the story map (sending it back to the card store).

be easily moved from one Propp card to another, *remove* is a quick way to remove the card from the story map (sending it back to the card store).

When a Propp card is removed from the story map, either by clicking its remove button or by dragging it to the stack or card store, the card retains any modifications that the user has made to its text. Propp cards are never deleted, only removed from the story map. They can always be retrieved again from the stack or card store with their text intact. This makes it difficult for the user to unintentionally destroy part of her story without being able to retrieve it again.

5.3.1.7 Saving and Loading from the Toolbar

Finally, figure 5.9 on the next page is a screenshot of the prototype application with the toolbar in view. When the user moves the mouse pointer to the bottom edge of the screen the transparent toolbar slides into view. The toolbar slides out of view again when the user moves the mouse pointer away. From the toolbar the user can access buttons for saving and loading story maps. Story maps are also saved automatically

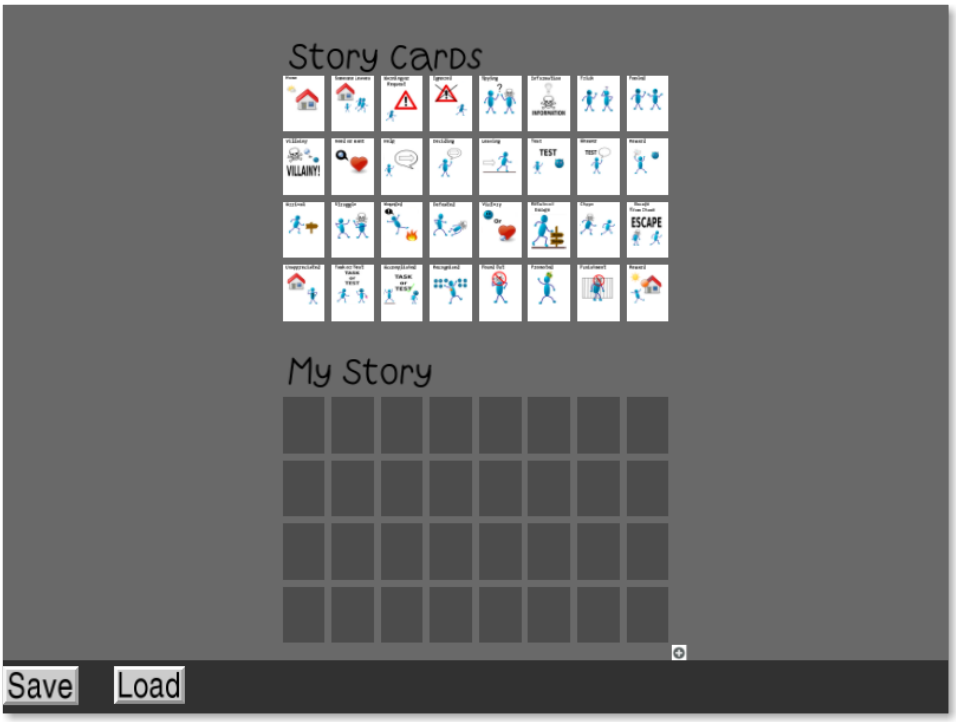


Figure 5.9: A screenshot of the prototype story maps application with the toolbar in view at the bottom of the screen. The toolbar slides into view when the user moves the mouse pointer to the bottom edge of the screen, and slides out of view again when the user moves the mouse pointer away.

every five minutes and whenever the application is closed.

5.3.2 Evaluation

A series of usability evaluations were carried out with the prototype implementation, the prototype being modified in response to the results of each evaluation before the next evaluation, eventually arriving at the final implementation presented in section 5.4 on page 164.

The usability evaluations began with usage simulations carried out by the researchers who, as expert users, attempted to simulate the behaviour of less experienced users in order to predict any problems that users would be likely to encounter, generating a list of usability problems to be addressed.

Next, four separate cooperative evaluations were conducted with two primary school teachers, one creative writing teacher and one pair of children as users. Each user was given the task of constructing and writing a story using the prototype, and asked to verbalise any problems that they encountered while carrying out the task. The researcher

observed the users and made notes, and asked the users questions or asked the users for suggestions in response to the researcher's observations or to the user's verbalisations.

Next, a formative evaluation was conducted in which four children aged 8-11 constructed and wrote stories using the prototype. A primary school teacher assisted.[2] The procedure used for this evaluation was a prototype of the procedure that would be used for the experimental study reported in chapter 7. The procedure was as follows: the teacher read a sample Proppian tale (*The Black Geese of Baba Yaga*) to the participants. The researcher showed the participants a story map for the sample tale in the story maps application on a projector, and stepped through the story map card-by-card with the participants. The researcher then demonstrated how to use the story maps application. Participants were then asked to construct and write their own stories using the story maps application. When the participants finished their stories there was a break for juice and biscuits, and then the teacher and researcher led a brief discussion about the positives and negatives of the prototype.

Finally, the prototype was tested in school conditions in a pilot of the experimental study that is reported in chapter 7.

5.3.2.1 Changes to the Prototype Implementation

Many minor modifications were made as the application evolved, tweaking the layout of user interface elements, changing button labels or making buttons larger, fixing bugs that were revealed, etc. For brevity this section will present only the main design changes that were made in response to the evaluations between the first prototype described above and the final implementation presented in section 5.4.

The *stack* (see section 5.3.1.5 on page 156) was removed. The stack was found to confuse users because it is not visible on screen until the user moves the mouse to the right edge of the screen. When a Propp card is added to the stack by clicking a button on the Propp card it disappears from screen and users do not realise where the card has gone. Also, some users tried to zoom in on cards in the stack in the same way as they could zoom in on cards elsewhere. This does not work because the stack is not part of the zoom world but is overlaid on the zoom world. Ideas were generated to mitigate these issues, but it was observed that users were easily able to move Propp cards by zooming in and out and drag-dropping them. The stack appears to be an unnecessary complication, so it was removed.

[2]Thanks to Miss Fleet from Maddiston Primary School for her help.

A *story editor* was integrated into the application, taking over the function of writing the story. Users no longer write their story by writing on the Propp cards themselves. The story editor is closer to a traditional linear text editor or word processor, but integrates closely with the Propp cards and story map. The final implementation of the story editor is described in section 5.4.3. The story editor solves a number of problems that were encountered with the prototype approach:

1. The instructions on each Propp card (the name of its Propp function, the illustration, and the longer description of the Propp function) were fighting for space with the user's text for her story which was also to be written on the Propp card. In the first prototype implementation the user erased the instructional text on the Propp card to replace it with the text for her story. This caused problems when users forgot what the instructions had originally read. The story editor takes over the function of holding the user's story text, allowing the Propp cards to be reserved for the text that describes the Propp functions.

2. A number of users ran out of space while writing their text for some Propp cards. Users pointed out that the amount of text that they are likely to write varies from card to card depending on the Propp function represented, some functions demand more than others. The story editor solves this problem by allowing the text area for each Propp card to expand as necessary, the text no longer needs to be contained on the Propp card itself.

3. One teacher got lost while writing her story, forgetting which Propp cards she had placed next and so making a mistake in her story. This problem was also observed with children using paper Propp cards in the exploratory study. If the writer does not keep her story map as a whole in view but instead views only one Propp card at a time then she may make a mistake of this sort. One teacher pointed out that because the user has to zoom in and focus on each card one by one to write the text on that card, the story map as a whole is not on screen when the user is writing her story. Only the card currently at hand is on screen. With the story editor the entire story map remains on screen above the story editor while the editor is used to edit the text associated with each card.

The buttons on the Propp cards were removed. The *move to stack* ('+') button is no longer needed because the stack has been removed. The copy and paste buttons have been moved to the story editor, which has taken over the text editing function from the

Propp cards. The *remove from story map* ('-') button was a potential source of error when the user accidentally removes a card by clicking the button, it was removed in favour of requiring users to zoom out and drag-drop Propp cards back to the card store to remove them from the story map.

Although most users quickly discovered and used the ability to zoom into and out of Propp cards, in some cases users appeared to forget or not to realise that they could do this and constructed their story maps without zooming in to inspect cards first. They decided which cards to use based on the illustration and (primarily) the name of the card, without seeing the more detailed descriptions of each card's function. The result was a story that did not accurately follow the story map, misinterpreting many of the Propp functions. Rather than requiring users to left-click on a Propp card and have the viewport zoom in to focus on the card to see it in more detail, in the final implementation Propp cards were made to scale up and show themselves in greater semantic detail when the user simply moves the mouse over them. Scaling on mouse-over is more easily discoverable than left-clicking to zoom in, and allows one Propp card to be viewed in detail while the other cards are still on screen.

Teachers expressed the desire to see a 'final product' when the user has finished writing their story, in which the Propp cards were sorted neatly or in which the story is presented in a more traditional linear format (instead of having to pan from Propp card to Propp card in the story map to read the story by reading the text on each card). A preview window was added, along with the ability to export the story in HTML format and to print the story, see section 5.4.4 on page 173. The addition of these features also meets functional requirements 5 and 6 (export a story for further editing in a word processor and print a story, see section 5.2 on page 147).

Users suggested that children might need some assistance with sorting their Propp cards into the correct order in the story map. A 'Sort' button was added to the story map which users can use to automatically sort their cards into the correct order.

The interface gave some users the impression that they were required to fill all of the empty boxes in the story map with Propp cards, and teachers agreed that children might get this impression. The appearance of the story map was changed so that the underlying grid to which Propp cards are snapped is not visible, and the story map area was made smaller than the card store area so that not all Propp cards can be used in one story map.

The texts for the Propp cards were rewritten with advice from teachers. The original texts, intended to be child-friendly descriptions of Propp's functions, were consid-

ered by teachers to be too long and complicated and were described as 'intimidating' for children. Teachers pointed out that faced with such long and difficult sentences, children would be likely to give up on reading them and refer to the illustrations instead. Teachers suggested breaking the text up into short phrases and using structured text layouts (e.g. bullet-point lists) to 'allow children to be systematic' when considering what they have to do for each Propp card. New texts for the Propp cards were developed based on the clear and concise definitions of Propp's functions that were developed in chapter 3. As well as its illustration each Propp card was given a one or two word name, a brief description derived from the definition of the Propp function, and a longer instructional text in the form of a bullet-point list, derived from the definitions of the function's species.

The Propp cards (their fonts and illustrations), the layout and the colours of the application were redesigned with assistance from a designer and illustrator, helping to meet usability requirement 6: that the application should be attractive, pleasant to use and motivating (see section 5.2 on page 147).

Finally, a number of users and teachers expressed the desire for some sort of drafting area in which Propp cards could be arranged before moving them into the final story map area. A number of ideas were generated but because it would greatly complicate the design and raise new questions that would require further evaluations with users, this feature was *not* added.

Following these usability evaluations, the final implementation of the story maps application described in the next section is considered to meet the functional, data and usability requirements specified in section 5.2.

5.4 Final Implementation

The final version the story maps application was implemented in the Java programming language using the structured 2D graphics framework Piccolo 2D.[3] The overall design of the code was based on that of the prototype implementation. Java and Piccolo offer a number of technical advantages. Piccolo supports the development of zoomable user interfaces, taking care of many of the unique features of such interfaces, so using the Piccolo framework allowed for rapid development and modification of the application. Piccolo also supports integration with Java's comprehensive user interface framework *Swing*, which supports the more common user interface elements of the application

[3] http://www.piccolo2d.org/

(buttons, menus, scrollbars, text editing, etc). Java applications are easy to distribute to users. A Java application can be published on the researcher's website as a JAR file for users to download and launch by double-clicking, or as a Java Web Start application that users can launch by clicking a link in their web browser. The researcher can also carry a JAR file containing the application on a USB thumb drive, which is useful when entering a school environment that is likely to have restricted or unreliable internet access. The tool must be usable on a number of different operating systems: those used by the researcher's computer, the home computers of teachers who helped test the application, the computers in the university computer lab where usability tests were conducted, and the computers in the different schools in which the pilot and main experimental studies were conducted. A JAR file or Java Web Start application will run on any operating system as long as a recent version of the Java Runtime Environment is installed. Finally, like the story maps application, both Java and Piccolo 2D are available as free software.

5.4.1 Propp Cards

New Propp cards for each of Propp's functions were developed for the application. Figure 5.10 on page 167 shows some Propp cards from the story maps application, appendix C contains the complete set of Propp cards used by the application. The illustrations for each Propp card were created for the story maps application by designer and illustrator Raymond Yuen[4] based on the researcher's texts for the cards.

A Propp card in the story maps application is an interactive object that can be dragged and dropped, and to which text can be added using the story editor (see section 5.4.3 on page 171). As well as an illustration each Propp card has a one or two word name derived from the name of the Propp function it represents, a one sentence description derived from the definition of the Propp function, and a longer instructional text derived from the species of the Propp function, where relevant.

Each Propp card has three levels of semantic detail. By default only the card's illustration and one or two word name are shown. When the user inspects the card by moving the mouse pointer over it the card scales up and the illustration and one-sentence description are shown, when the user moves the pointer away the card scales down again. Scaling up and down on mouse-over also serves to indicate that the Propp card is an interactive object and make the action of dragging the card with the left-

[4] http://rayuen.com

mouse button discoverable. When the card is viewed in the story editor the illustration, name, description and longer instructions are all shown. As suggested by teachers, the instructions are broken up into short phrases and structured using bullet-points, allowing children to be systematic when considering what they must do for a card and what options they have. Figure 5.11 on page 168 shows a Propp card at each level of semantic detail.

The Propp cards, their order and the name, description and instructions for each are specified in a JSON-formatted text file[5] that is read by the story maps application. This allows the text of the Propp cards to be tweaked, illustrations replaced, or the set of Propp cards available in the application to be changed or reordered easily without modifying or recompiling the application source code.

5.4.2 The Planning View

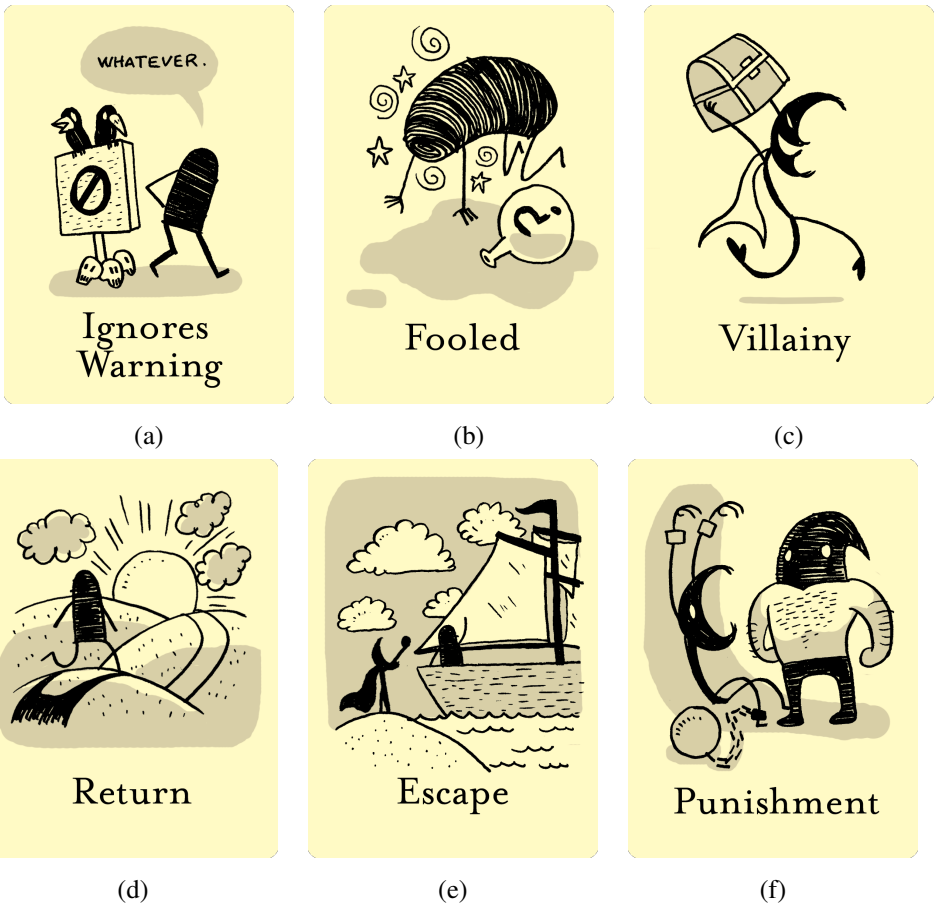
Figure 5.12 on page 169 is a screenshot of the opening screen of the application, the planning view. The user is presented with a selection of Propp cards to choose from in the green area at the top of the screen (the card store). The user can inspect each Propp card by moving her pointer over it to see it in more detail. By dragging and dropping Propp cards the user arranges the cards that she wants to use for her story in the initially empty grey area at the bottom of the screen (the story map).

Propp cards can be dragged from the card store to the story map, from the story map back into the card store, or rearranged within the story map. The position of Propp cards in the story map is determined by the user by drag-dropping. In the card store the cards always maintain Propp's overall sequence, when a card is dropped anywhere in the card store it snaps to its starting position in the card store. Figure 5.13 on page 170 shows the story map view after some Propp cards have been dragged from the card store into the story map. When a Propp card has been removed from the card store a faded copy of the card remains visible in the card store. This means that the user can see where a card she has moved came from, and when inspecting the remaining cards in the card store she can see which cards used to be between them so that the sequence of the Propp cards (which contributes to their meaning) is never broken in the card store.

Attached to the story map at the bottom of the window is the story map toolbar. The toolbar allows the user to enter a title for her story. The *Sort* button on the right of

[5] JSON is an open standard data-interchange format that is easy for humans to read and write: `http://www.json.org`.

Figure 5.10: Some Propp cards from the final story maps application, shown at their lowest level of semantic detail. The Propp cards were illustrated by designer and illustrator Raymond Yuen (`http://rayuen.com`).



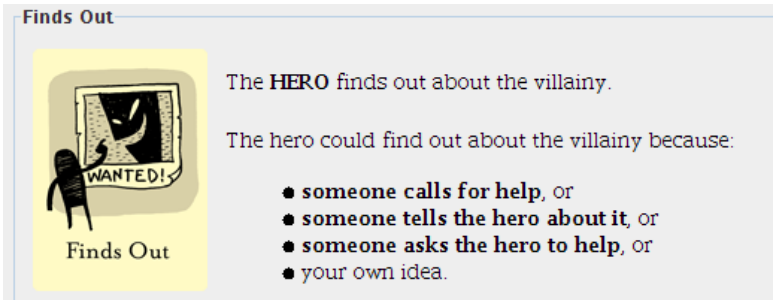
(a) (b) (c)

(d) (e) (f)

Figure 5.11: One of the Propp cards from the final story maps application shown at three levels of semantic detail.

(a) Normal.

(b) On mouse over.



(c) In the editor view.

Figure 5.12: A screenshot of the planning view in the final story maps application.

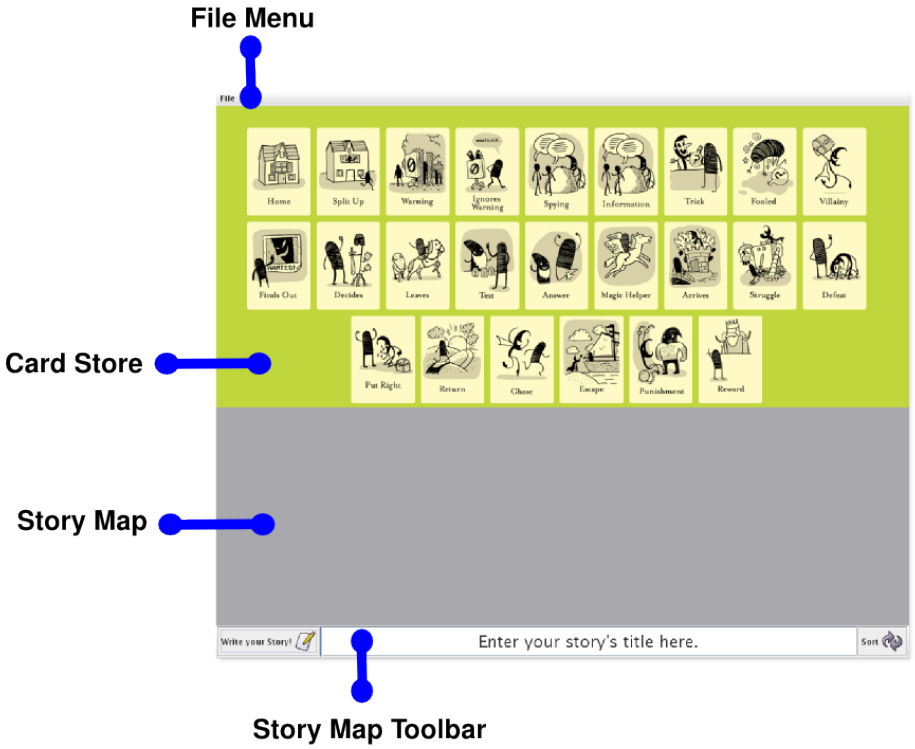
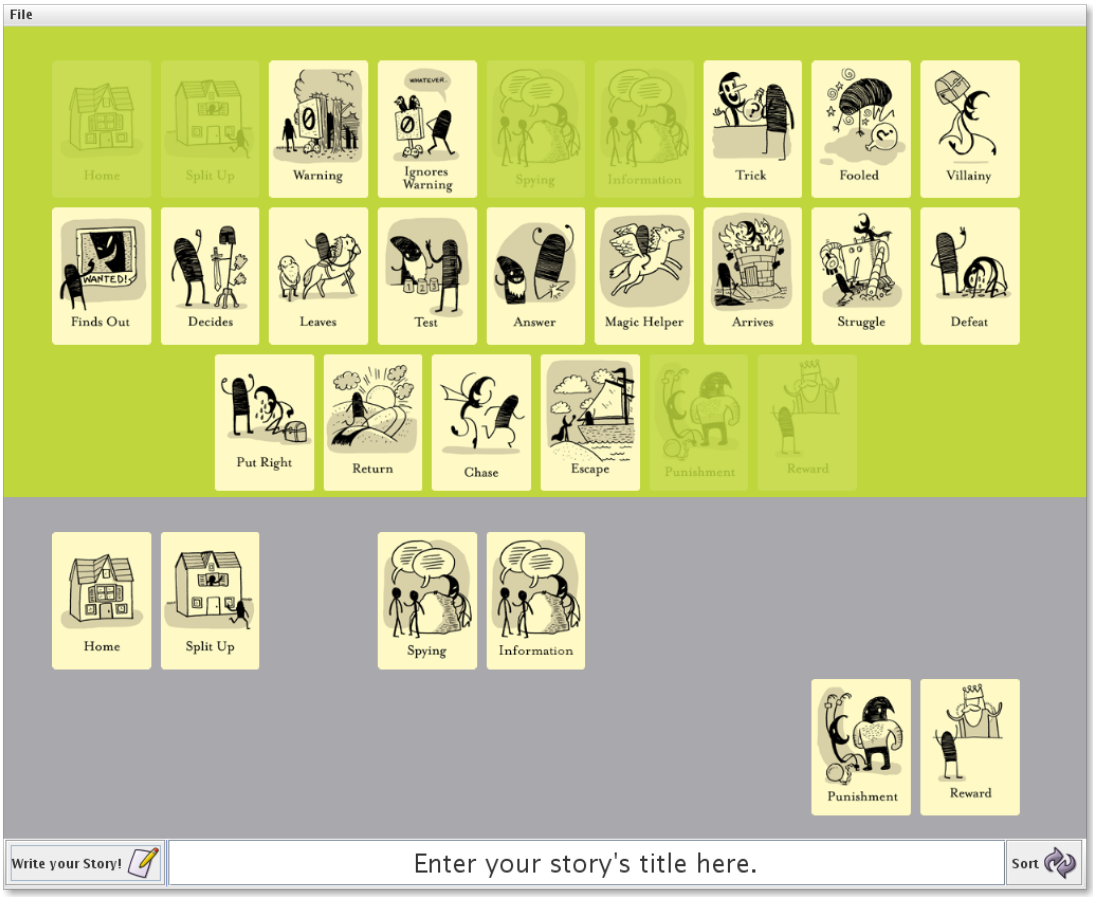


Figure 5.13: A screenshot of the planning view of the final story maps application after some Propp cards have been dragged from the card store (above) into the story map (below). A faded, non-interactive image of each moved Propp card is left behind in the card store.



the toolbar sorts the cards in the story map into Propp's overall sequence. The *Write your Story!* button on the left of the toolbar activates the story editor.

5.4.3 The Story Editor

Figure 5.14 on the following page is a screenshot of the story editor view. The story editor is activated when the user clicks the *Write your Story!* button. Figure 5.15 on page 173 illustrates the transition from the planning view to the editor view. In an animated transition the card store slides out of view above the top of the screen and the story editor slides into view from below. The story map moves from the bottom half of the screen to the top and remains in view. The toolbar attached to the bottom of the story map, previously at the bottom of the screen, is now in the middle of the screen. The label of the *Write your Story!* button on the story map toolbar has changed to *Go back to planning*. This button can be used to move between the planning and writing views at any time. As figure 5.15 on page 173 shows, conceptually the card store, story map and story editor are arranged in a column in the navigation space and moving between the two views consists of panning the camera up or down within this space. The use of animation makes the spatial relationships of the components intuitive to the user. The file menu is outside of the animation and remains at the top of the screen in both views.

In the top half of the story editor view is the user's story map. As in the planning view the user can see all of the cards in her story map and can use the mouse to inspect the cards or rearrange the cards within the story map by drag-dropping. If the user wants to remove a Propp card from her story map and put it back into the card store or wants to move a new card from the card store into her story map she must go back to the planning view to do so.

In the bottom half of the story editor view is the story editor itself. The story editor displays a single Propp card from the story map at a time. The card is displayed in the story editor at its highest level of semantic detail, with the full textual description of the card's Propp function. Below the card is a text area in which the user can enter her text for that card. The *Next* and *Prev* buttons to the right and left of the story editor bring up the next or previous card from the story map into the editor. The user can also click on a Propp card in her story map to show that card in the editor.

At the bottom of the story editor view is the editor toolbar. On the left hand side of the toolbar are buttons for the common text-editing functions cut, copy and paste.

Figure 5.14: A screenshot of the story editor view in the final story maps application.

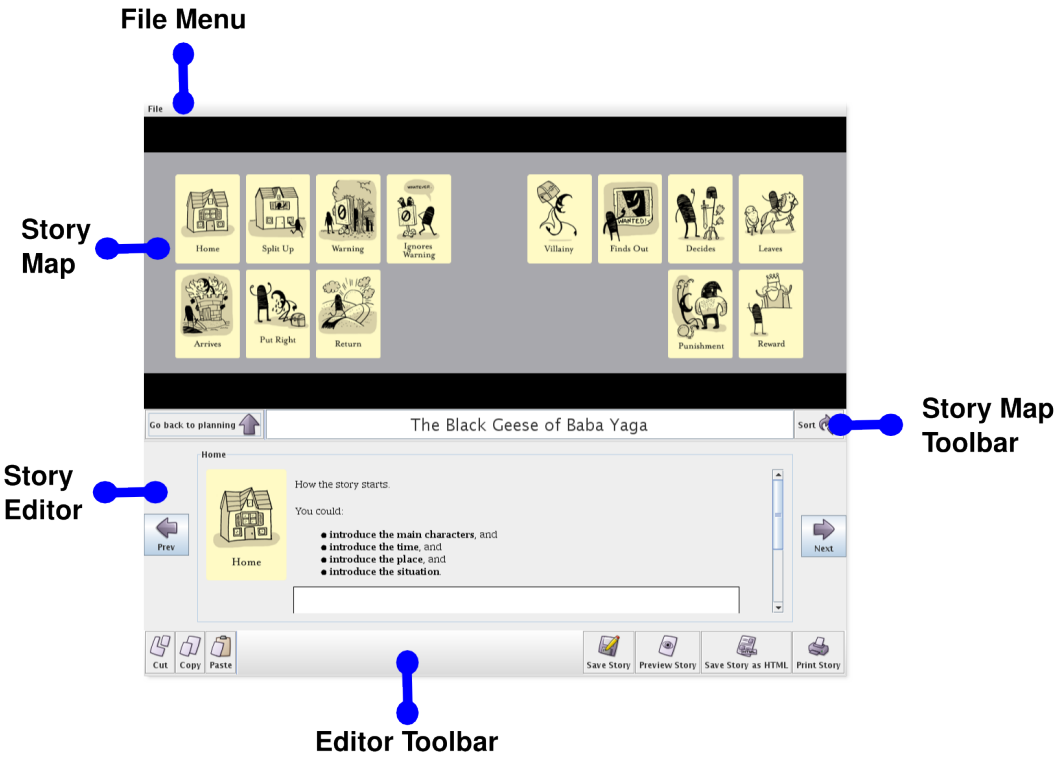
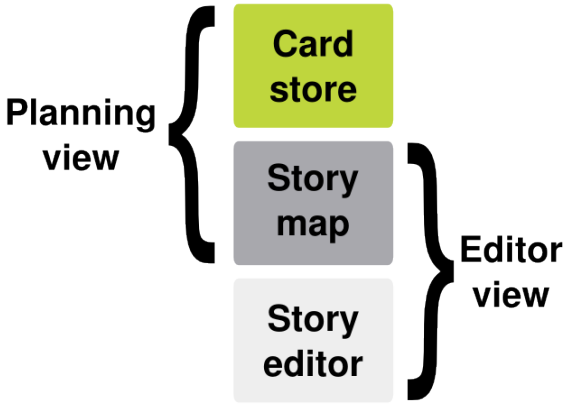


Figure 5.15: An illustration of the navigation space within which the card store, story map and story editor are contained. In the planning view the card store occupies the upper half of the user's viewport, the story map occupies the lower half, and the story editor is out of view. In the editor view the card store is out of view, the story map occupies the upper half of the viewport, and the story editor occupies the lower half of the viewport. The *Write your Story!/Go back to planning* button (located on a toolbar at the bottom of the story map) triggers an animated transition from the planning view to the editor view or back.

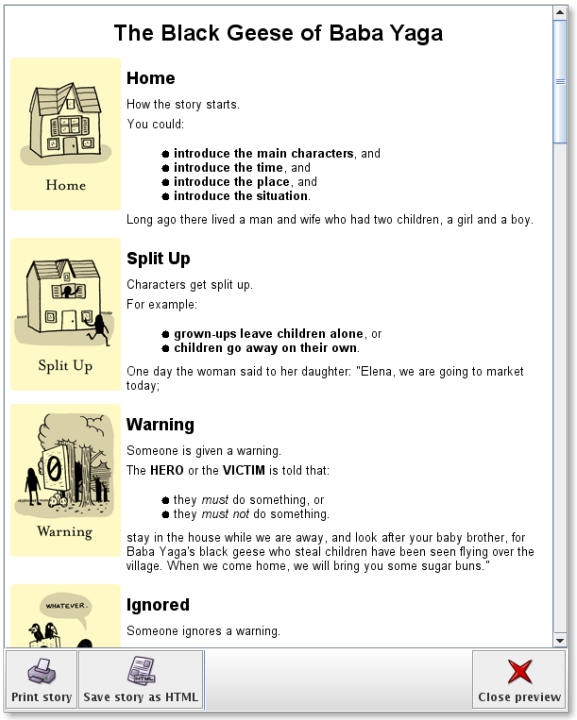


On the right hand side of the toolbar are four buttons: *Save Story* (clicking this button activates a file chooser dialogue for the user to select a location and file name to save a copy of her story to), *Preview Story* (clicking this button activates the preview window, see section 5.4.4), *Save Story as HTML* (clicking this button activates a file chooser dialogue for the user to select a location and file name to save an HTML copy of her story to) and *Print Story* (clicking this button actives a standard print dialogue).

5.4.4 The Preview Window

Figure 5.16 on the following page is a screenshot of the preview window, a popup window that appears when the user clicks the *Preview Your Story* button in the *File* menu or on the story editor's toolbar. The window contains a preview of how the story will look when exported to HTML or printed. Story maps are printed with their title at the top and the Propp cards arranged vertically. To the right of each Propp card is the detailed description of that Propp card's function, followed by any text that the user has added to the Propp card. At the bottom of the preview window is a toolbar from which the user can print her story, export it to HTML, or close the preview window

Figure 5.16: A screenshot of the preview window in the final story maps application.



and return to her story map.

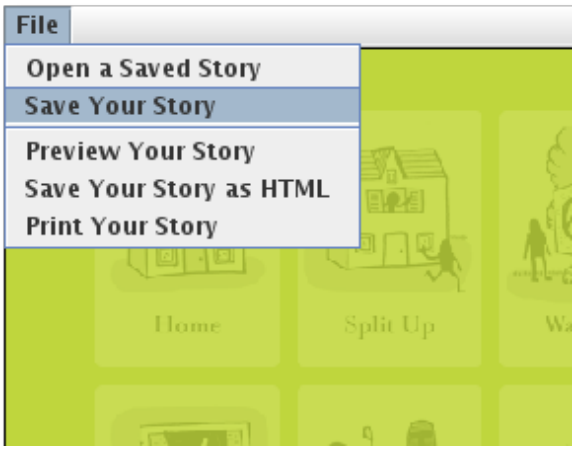
5.4.5 The File Menu

The *File* menu (figure 5.17 on the next page) is always present in the top right corner of the application window. From the *File* menu the user can open a saved story map (a file chooser dialogue is activated for the user to locate her saved story map file), save her story map (a file chooser dialogue is activated for the user to select a location and file name to save a copy of her story to), preview her story (the preview window is activated), export her story to HTML (a file chooser dialogue is activated for the user to select a location and file name to save an HTML copy of her story to), or print her story (a standard print dialogue is activated).

5.4.6 Saving and Loading

The user can save a copy of her story to a location of her choice by using the *Save your Story* button from the file menu or the story editor toolbar. However manual saving is *not* required because the application regularly saves its state automatically as the user works and when the application closes. Stories are saved using the story maps

Figure 5.17: A screenshot of the file menu in the final story maps application.



application's file format, and can be opened again using the *Open a Saved Story* item from the *File* menu. All aspects of the story maps application's state are saved: the Propp cards that the user has in her story map and their positions within the story map, any text that the user has added to any of the Propp cards, etc. The default location for saved stories that the user is presented with is the StoryMaps folder, which is created by the story maps application in the user's starting directory (in the user's home directory on Linux, in the user's My Documents folder on Microsoft Windows, etc.).

Primarily for research and development purposes, the story maps application automatically saves copies of story maps in an autosaved_storymaps folder within the StoryMaps folder. Each time the story maps application is launched a subdirectory for the session is created in the autosaved_storymaps folder, and copies of the user's story map are regularly saved in this directory as the user works and when the application closes. The names of the automatically saved files and their directories indicate the time and date at which they were created. A log file containing timestamped logs of the user's interactions in XML format is also saved in each session's directory alongside the automatically saved story maps.

5.5 Conclusion

The chapter has described the design, implementation and evaluation of a computational story authoring application based on Propp's morphology. The design of the application was based on the paper and card story writing tools that proved successful in the previous exploratory study, these foundations were described in section 5.1.

Formal requirements for the application were specified in section 5.2, and section 5.3 presented the implementation and evaluation of an initial prototype. Several significant modifications and many minor modifications were made to the prototype in response to evaluations with users, eventually arriving at the final implementation that was presented in section 5.4. This final implementation meets the requirements specified, and was used in the experimental study of children's story authoring that is reported in chapter 7.

Now that an initial story authoring application based on Propp's morphology has been designed, implemented and verified, it can form the basis of further work exploring the integration of more advanced features into the application.