Growing Up With Nell: A Narrative Interface for Literacy

C. Scott Ananian Chris J. Ball Michael Stone*
One Laptop Per Child Foundation
222 Third Street
Cambridge, MA 02142
{cscott,cjb,mstone}@laptop.org

ABSTRACT

Nell is a tablet-oriented education platform for children in the developing world. A novel modular narrative system guides learning, even for children far from educational infrastructure, and provides personalized instruction which grows with the child. Nell's design builds on experience with the Sugar Learning Platform [17], used by over two million children around the world.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*narrative interfaces*; K.3.1 [**Computers and Education**]: Computer Uses in Education—*computer-managed instruction*

General Terms

Design, Human Factors

Keywords

Narrative interfaces, tablet computing, education, Nell

1. A LESSON FROM NELL

Miles from the nearest school, a young Ethiopian girl named Rahel turns on her new tablet computer. The solar-powered machine speaks to her: *"Hello! Would you like to hear a story?"*

She nods and listens to a story about a princess. Later, when the girl has learned a little more, she will tell the machine that the princess is named "Rahel" like she is and that she likes to wear blue—but for now the green book draws pictures of the unnamed Princess for her and asks her to trace shapes on the screen.

"R is for Run. Can you trace the R?"

As she traces the R, it comes to life and gallops across the screen. *"Run starts with R. Roger the R runs across the Red Rug. Roger has a dog named Rover."* Rover barks: *"Ruff! Ruff!"* The Princess asks, *"Can you find something Red?"* and Rahel uses the camera to photograph a berry on a nearby bush.

"Good work! I see a little red here. Can you find something big and red?"

As Rahel grows, the book asks her to trace not just letters, but whole words. The book's responses are written on the screen as it speaks them, and eventually she doesn't need to leave the sound on all the time. Soon Rahel can write complete sentences in her special book, and sometimes the Princess will respond to them. New stories teach her about music (she unlocks a dungeon door by playing certain tunes) and programming with blocks (Princess Rahel helps a not-very-bright turtle to draw different shapes). Rahel writes her own stories about the Princess, which she shares with her friends. The book tells her that she is very good at music, and her lessons begin to encourage her to invent silly songs about what she's learning.

An older Rahel learns that the block language she used to talk with the turtle is also used to write all the software running inside her special book. Rahel uses the blocks to write a new sort of rhythm game. Her younger brother has just received his own green book, and Rahel writes him a story which uses her rhythm game to help him learn to count.

*Now at Akamai Technologies.

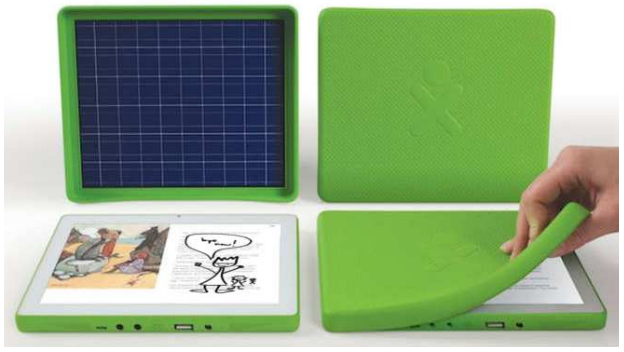


Figure 1: Nell's solar-powered hardware platform. The tablet design allows for direct interaction.

2. KEY IDEAS

The interaction design of the Nell system described above is inspired by the "Young Lady's Illustrated Primer" in the Neal Stephenson novel *The Diamond Age*, from whose protagonist Nell takes its name.[1] Nell's design embodies four key ideas: it is a **Narrative** interface using **Direct Interaction** which **Grows** with, and is **Personalized** for, the child. Nell uses these four key concepts to build a novel learning platform which addresses several challenges we've encountered in earlier work.

2.1 Narrative

Children (like all humans) are hard-wired for stories [4]. In our earlier learning software, we've seen some of our favorite pedagogical activities neglected because children had difficulty finding their way into the material without an enthusiastic teacher's guidance. In response, Nell uses a *Narrative Interface* [3, 6] to pave a path for the child user through the learning material. In the narrative interface, all system actions are shaped by a storybook metaphor, and interactions with the system are framed as interactions with one of the system protagonists.

Nell's overall story is a multi-character serial adventure, taking cues from *The Diamond Age* and from the UNIVERSE narrative-generation system [9]. Each of the several characters represents a specific skill or subject area, and each adventure represents about a year's curriculum. Adventures are further subdivided into story modules, which match the scope of a traditional lesson plan. The serialized multi-character design improves modularity and allows updates and decentralized authorship.

Nell's characters are always-available agents layered above a particular system *activity* (application) which provides specialized functionality. The agents are not always foregrounded: constructionist learning occurs when the child plays freely to "make things" with the base activity. The handwriting tutor is fundamentally a drawing activity; the adventure involving the magical musical lock is also a music-making activity. The narrative system is hooked into each activity to provide passive guidance (congratulating the child when it notices they've drawn a letter), active guidance (Apple-Guide–style [11] contextual help), or system services (switching activities; jumping into a related story module). A system-wide *achievement* system provides goals and rewards.

2.2 Personalized

Children learn in different ways. Nell provides multiple story modules for its plot points/lessons to engage the child's interests and cater to multiple intelligences [8]. Selecting between the alternatives can depend on either explicit choices made by the child ("fractions in outer space," if the child chose a space-themed story or previously indicated an interest in space) or prior success with a given lesson style (a large number of accomplishments in musical-rhythmic tasks suggests a rhythmic approach to fractions). Similarly, several different achievements can coexist, rewarding different ways of accomplishing the same pedagogical goal.

The record of past choices and accomplishments is stored in a *Journal* and can be reviewed by the child. The contents of the Journal can be rearranged to create a Portfolio [16] demonstrating the child's progress. In order to encourage fearless play and experimentation, the Journal also supports pervasive undo; the child can rewind and replay the narrative starting at any past point in the journal.

Content can be remixed for further personalization. The child is encouraged to rename characters and change clothing, colors, and other superficial details. In the future we expect to accomplish further narrative customization by recombination of story elements. Lebowitz [9] and Riedl [13] show how a planner can be used to recombine and adapt story fragments. We have less ambition than the cited work: instead of attempting to generate thousands of stories from tens of templates, we hope to select and then modestly adapt from hundreds of story modules created in a decentralized manner by teachers—and eventually by the students themselves.

2.3 Growable

Children grow. Nell aims to provide a "low floor and no ceiling" to grow with them, along three main axes.

First, Nell tracks the child's growing intelligence and capability with increasingly challenging story material. As the child accumulates achievements and demonstrates proficiency, the serial story moves on to more advanced topics.

Second, Nell seeks to grow its authoring community. Decentralized authorship ensures that Nell's instructional content continues to increase in the number of topics and customizations. There are no ceilings between children and teachers: Nell contains a story editor and everything necessary to author and publish story modules.

To further promote collaboration, Nell is free and open source and implemented in standard web technologies (JavaScript, HTML5, and WebGL) with offline caching. Resources are named by URL, even when disconnected from the internet, which simplifies the distribution of updates to story modules and the Nell system. URL-based identifiers also allow third parties to manage their own namespaces when extending Nell.

Finally, Nell is designed to allow a capable child to learn about the construction of the system itself, eliminating the ceiling between the child and the authors of the Nell system. Nell can teach about itself, and its source code is open for exploration within Nell. Meaningful changes can be performed without external tools.

2.4 Direct interaction

The constructionist learning philosophy emphasizes creation and tangible interaction. Nell uses direct interaction on a tablet computer to maintain the child's connection to their work (Figure 1). In particular, Nell uses handwriting recognition as a primary interface.

The direct interaction model lowers the floor by eliminating the need to learn an abstract touchpad or mouse interface. It also supports our literacy goals by allowing direct handwriting instruction and development of motor skills.

3. TECHNOLOGY

Our implementation choices further our goals of modularity and decentralization. In this section we describe the technologies that enable Nell's four key ideas.

3.1 Rule-based story modules

To implement Nell's modular narrative interface, we use a production rule system with programmable conflict resolution. Figure 2 shows an excerpt of the story module

[1]In a nod to Seymour Papert, Nell can also be read as an acronym for "Narrative Environment for Learning Learning."

```
{ "world": { // id gives canonical URL for this module
    "id": "/nell.laptop.org/chapters/alphabet-book",
    "scene": "opening", // initial scene
    "inherit_from": "/nell.laptop.org/core/drawing_activity",
  }, "scenes": [
    { "id": "opening", subject: "actors/princess",
      "verb": "speak", object: "actors/user",
      "text": "Hello! Would (o/2) like to hear a story?",
      "group": "/nell.laptop.org/core/story-intro-group",
    },
    { "id": "story-1",
      "cond": rule‘At(opening) && Event(Chat, Yes)‘,
      "subject": "actors/princess",
      "verb": "speak", object: "actors/user",
      "text": "Once upon a time..." },
    ...
    { "id": "r-drawn",
      "cond": rule‘Event(Activity, DrawR)‘,
      "subject": "actors/princess",
      "verb": "speak", object: "actors/user",
      "text": "Very good, (o/3)!",
      "action": js‘startAnimation("r-runs")‘ }, ...
  ], ... }
```

Figure 2: Story module excerpt written in JSON (http://json.org) extended with quasiquote [14].

underlying the interaction described in Section 1.

Stories consist of a sequence of *scenes* plus information about characters, locations, and activities. Scenes are "goals" in the STRIPS formalism [7] and consist of action descriptions, preconditions, and effects. In principle, all preconditions for all scenes of all story modules are evaluated continuously to determine the actions of the narrative agents. This evaluation is made computationally efficient using the Rete algorithm. Dialog is expressed using Curveship-style templates, allowing Nell to restructure expository tense, style, and point-of-view [10].

New story modules can be downloaded from the Internet in connected deployments. In disconnected deployments, new stories might be distributed once a year on USB sticks or shared among friends. In a classroom environment, a teacher can share stories for the day's lesson at the start of class.

Story modules can be inherited and extended. This makes it easy to take an existing story and modify it to better suit particular interests, a particular type of learner—or just to add variety. This can lead to conflicts: multiple versions of a story, or scenes within a story, may have their preconditions satisfied simultaneously.

Conflicts are resolved by consulting a *group* associated with each scene. Each group has a default priority and names a scene to be invoked when a conflict exists; the actions associated with that scene will (eventually) alter the priorities of the conflicting scenes to resolve the conflict. The resolution scene may include any number of actions. It may select randomly among the conflicting scenes or base its selection on the child's preferences, curriculum progress, or inferred learning style. The resolution scene may even initiate a new multi-scene story.

For example, the *story-intro-group* used in Figure 2 defaults to a low priority so that continuing a story in progress is preferred above starting a new story. When a story module is completed, several new story openings will be in conflict. The resolution scene for *story-intro-group* may bring

```
{ "aiml": {
  "version": "1.0",
  "categories": [
    { "pattern": "GO *",
      "template": "{*}" },
    { "pattern": "N",
      "template": "{NORTH}" },
    { "pattern": "NORTH",
      "template": js‘FireEvent(Chat, "NORTH")‘ },
    { "pattern": "* THE DOOR",
      "template": "{*} DOOR" },
    { "pattern": "* SQUEAKY DOOR",
      "template": "{*} DOOR" },
    { "pattern": "OPEN DOOR",
      "template": js‘FireEvent(Chat, "OPEN DOOR")‘ }
  ] } }
```

Figure 3: An AIML fragment which recognizes the commands `North`, `Go North`, `Open the squeaky door` and variants. AIML's XML syntax has been translated to JSON. Template contents in braces invoke the recognizer recursively (what AIML calls `<srai>`).

us to the controls of an airship which flies between locations corresponding to the various story continuations. In that set of scenes, the child may eventually decide to fly to Mathland to hear the next episode in her agent Mathis' adventure, which would bump up the appropriate scene's priority to make the transition without further conflict.

Nell will include an integrated story editor. Our exemplar is the Wide Ruled authoring tool, which was successfully used by non-technical story creators [15].

3.2 Extensible dialog

Nell's use as an interactive diary and *portfolio* [16] is enhanced by the child's collaboration in the fiction that Nell is intelligent—what Turkle [18] calls the "ELIZA effect." Convincing discourse is a hard problem, but reasonable approximations do not have to be difficult; even the rudimentary conversational abilities of ELIZA elicited hours of conversation.We've chosen to allow the child to directly converse with the Princess and other agents within Nell.

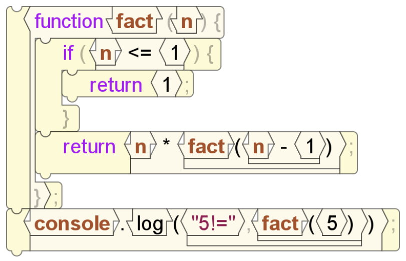
Our implementation extends AIML [1], the markup language developed for the Loebner Prize–winning AliceBot.[2] Each story module can include AIML fragments extending the conversational capabilities of Nell's agents.

Figure 3 shows an AIML fragment augmenting an agent with commands suitable for a text adventure story module. In such a story the child might write instructions such as `go north`, `go east`, or `open door` in addition to the usual dialog with the agent. These new commands fire events which are referenced by scene preconditions. For example, `north` may trigger a scene which causes the activity to draw a new location on the page, causes the agent to write and speak a description of the new location, and finally causes a shift in the AIML *topic* to enable additional vocabulary suitable for the new location.

3.3 TurtleScript

We have developed a block-oriented view of JavaScript,

[2]Wilcox identifies several shortcomings with the expressive power of AIML, proposing ChatScript as a replacement [19]. AIML's community and diversity of third-party implementations make it our choice at present.



```
function fact(n) {
  if (n <= 1) {
    return 1;
  }
  return n * fact(n - 1);
};
console.log("5!=", fact(5));
```

Figure 4: Factorial function in TurtleScript.

which we call TurtleScript [2]. The TurtleScript dialect of JavaScript is the single implementation language for Nell.[3] TurtleScript scales from Logo-like pedagogical tasks (in subsets of the language) up to the implementation of TurtleScript and the Nell system itself. As Figure 4 shows, TurtleScript's block view is isomorphic to JavaScript's standard textual representation to ensure the readability of and compatibility with existing JavaScript libraries. We have begun implementation of a tablet programming environment for TurtleScript which uses drag-and-drop, direct interaction, and intelligent prompting to reduce text entry requirements.

TurtleScript allows the child to dig as far into Nell's implementation as their capabilities allow. Although the operating system and the browser implementation remain inaccessible, incorporating a meta-circular interpreter for JavaScript[4] allows us to expose the system all the way down to the programming language implementation.

4. RELATED WORK

The TinkRBook system [5] shares a similar focus on personalization and literacy. Nell attempts to scale the system up with a modular extensible narrative system and a broader range of pedagogic material.

The Sugar Learning Platform [17] is an earlier effort at child education in which we've been directly involved. Nell's focus is on children further from educational infrastructure than those targeted by Sugar. The narrative at Nell's core attempts to provide students pedagogical guidance missing from Sugar, and the direct interaction model avoids keyboard problems which have plagued Sugar's initial deployments. Nell also attempts to provide a greater sense of ownership through pervasive customization.

Automatic narrative generation has seen attention in gaming, simulation, and training contexts [12]. We believe Nell's synthesis of these ideas in a pedagogical context is unique.

5. CONCLUSIONS

We have described the design of a novel narrative direct-interface system for education, starting with literacy, which is personalized for and grows with its child owner. Personalized direct interface engages the child, and narrative guides them through pedagogic material. Low-cost solar-powered hardware allows Nell to reach further into the least-developed areas of the world to help those without traditional educational infrastructure.

We are in the process of implementing the Nell system, and expect to field initial deployments of subsets of Nell in Africa later this year. Volunteer assistance is welcome.

[3]Since JSON is the data-oriented subset of JavaScript, our examples in Figures 2 and 3 are valid TurtleScript.

[4]For example, `https://github.com/mozilla/narcissus/`

6. REFERENCES

[1] A.L.I.C.E. AI Foundation. Artificial intelligence markup language (AIML) version 1.0.1. `http://www.alicebot.org/TR/2005/WD-aiml/`, 2005.

[2] C. S. Ananian. `http://cscott.net/Projects/TurtleScript/`, 2011.

[3] J. Bizzocchi. Games and narrative: An analytical framework. *Loading—the Journal of the Canadian Games Studies Association*, 1(1), 2007.

[4] B. Boyd. *On the Origin of Stories: Evolution, Cognition, and Fiction*. Belknap Press of Harvard University Press, 2009.

[5] A. Chang and C. Breazeal. TinkRBook: Shared reading interfaces for storytelling. In *Proc. of the 10th Int'l Conf. on Interaction Design and Children (IDC '11)*, pages 145–148. ACM, June 2011.

[6] A. Don. Narrative and the interface. In B. Laurel, editor, *The Art of Human-Computer Interface Design*, pages 383–391. Addison-Wesley, Reading, MA, 1990.

[7] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4):189–208, 1971.

[8] H. E. Gardner. *Frames Of Mind: The Theory of Multiple Intelligences*. Basic Books, Dec. 1983.

[9] M. Lebowitz. Story-telling as planning and learning. *Poetics*, 14(6):483–502, Dec. 1985.

[10] N. Montfort. Curveship's automatic narrative variation. In *Proc. of the 6th Int'l Conf. on the Foundations of Digital Games (FDG '11)*, pages 211–218, June 2011.

[11] J. Powers. Giving users help with Apple Guide. *develop*, 18, June 1994. Apple Computer, Inc.

[12] M. O. Riedl, A. Stern, D. Dini, and J. Alderman. Dynamic experience management in virtual worlds for entertainment, education, and training. *Int'l Trans. on Systems Science and Applications*, 3(1), 2008.

[13] M. O. Riedl and R. M. Young. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39:217–267, 2010.

[14] M. Samuel. Ecmascript quasi-literals. `http://wiki.ecmascript.org/doku.php?id=harmony:quasis`, 2012.

[15] J. Skorupski and M. Mateas. Interactive story generation for writers: Lessons learned from the Wide Ruled authoring tool. In *Proc. of the 8th Digital Art and Culture Conf. (DAC)*, Irvine, CA, Dec. 2009.

[16] E. Stefanakis. *Multiple Intelligences and Portfolios: A Window Into The Learner's Mind*. Heinemann, 2002.

[17] Sugar Labs. `http://sugarlabs.org`.

[18] S. Turkle. *Alone Together: Why We Expect More From Technology and Less From Each Other*. Basic Books, 2011.

[19] B. Wilcox. Beyond Façade: Pattern matching for natural language applications. `http://chatscript.sourceforge.net/Documentation/Pattern_Matching_for_Natural_Language_Applications.pdf`, Feb. 2011.