Learning and Reuse in Visual Programming Environments: Teacher Simulation Creation Environment

Cheryl Denise Seals and Dr. Mary Beth Rosson
Virginia Tech Human Computer Interaction Group, Department of Computer Science
{cseals; rosson}@vt.edu

ABSTRACT

The use of computers in schools to promote active learning by students is not common. This research studies educational software simulations and utilizes the results to create a visual programming environment that will support the creation of simulations by teachers who are novice programmers. This will provide a tool that will empower them to create and modify their own software, and reuse software components.

1. Introduction

Seymour Papert described the design of Logo as taking the best ideas of computer science about programming language design and "child engineering" them [5]. Much has changed since the beginnings of Logo with much progress in programming language research and human computer interfaces. However in the area of end user programming, there is still need for improved usability: systems need to be easy to learn, easy to use, flexible, and pleasurable to use [4].

The main issue in end user programming is the relationship between ease of use and programming power. This has been studied with children as the target audience, but there has been almost no investigation of how end user programming should be introduced to someone who is mature and skilled, but is still a novice programmer [1, 3].

The target population of this work is secondary school teachers. We believe that empowering teachers would greatly improve educational software. Soloway points to the lack of money to create new educational software. [9] One way to get new software into the classroom is to create an environment that will allow teachers to create new software.

2. Research Process

Our research process has covered the complete software development life cycle. The Requirements Analysis stage consisted of initial requirements, intrinsic evaluations, and empirical evaluations. The evaluators began with a set of requirements acquired from intrinsic evaluations. We then utilized user surveys and empirical analysis to refine our requirements and initiate design. Implementation of the system has begun. Summative empirical analysis will be performed in both laboratory and field settings.

2.1 Intrinsic Evaluation with Scenarios and Claims

We investigated more than a dozen visual end user programming environments: simulation specific software that is procedure-based, mathematics-based or rule based (e.g. Star Logo, Toon Talk, Model-It, Agentsheets, Cocoa); viewable/observational software that are simulation environments with functionality that cannot be modified (e.g. SimCity, SimCalc); and observational software that are construction kits (e.g. ActivChemistry, Geometer's Sketchpad) [7].

An intrinsic evaluation is an analysis of an artifact directed at its functionality or features. Scriven contrasts this with empirical evaluations that involve observing an artifact in use [8]. The intrinsic analysis utilizes scenarios and claims analysis to identify the strengths and weaknesses of existing systems. [2] Assessing system features with claims analysis in a realistic context provided guidance for our system design. We examined usage features for four types of usage scenarios: Exploration and Learning (Environment Focus), Experienced Use (Environment Focus), Mapping Real World to Simulation (Modeling Focus), and Adapting of Existing Projects (Reuse Focus). Based on the result of this evaluation, we selected an environment for empirical analysis.

2.2 Empirical Analysis

After completing, intrinsic analyses, we performed an empirical evaluation of Agentsheets$_{TM}$, and are planning a second study with Stagecast Creator$_{TM}$. The empirical analysis was performed in a controlled laboratory setting with two evaluators. The participants were trained using a minimalist instructional manual, and were instructed to use the "think aloud" protocol. There were trained in the basic use of the environment, how to build simulations in this environment, how to reuse simulations, and were also given an interaction guide to aid with error recovery and to review of helpful shortcuts. One evaluator was in the proximity of the participant in case of serious breakdown. The second evaluator was located in a separate observation room to record critical incidents without distracting the participant, and to control the audio-visual equipment recording the session. The goal of the experiment was to establish whether teachers would create real simulations quickly, relying on their domain expertise, and new skills they have learned about simulation creation.

3. Visual Programming Environment

Our results will support the development of an environment aimed at teacher creation of science simulations, while mitigating identified usability problems. The rationale for building simulations as educational material is very practical. As Kuyper argues [4], simulations are independent of time and place, which makes them more readily available for real experience. Simulations can also provide better conceptual model of a situation, and can be used to create virtual environments, taking learning into the realm of imagination [4]. The emphasis on reuse is tied to the need for teachers to benefit from one another's work. The current systems that we have explored have some low-level components that can be reused. We would like to explore the reuse of higher-level components, e.g. entire simulations. Our initial plan is to support reuse of simulations and their components at various levels.

4. Research Status

At present intrinsic evaluations have been performed on two environments and empirical studies have been carried out with area middle and high school science teachers. The majority of users were able to create simulation with the environment we studied. “General reactions were positive; all agreed that they could imagine using this software in their own classes”[6]. They had little trouble with the visual aspects of the system. However a number of problems were observed in the following areas: variety of drawing tools provided, icons, rules/actions, general environment (e.g. ease of direct manipulation), and general usability for novices. We have conducted a visual language workshop with area teachers and community members to brainstorm directions and practical uses of visual languages for novices. We are currently working on a prototype environment Teacher Simulation Creation Environment, being built in Squeak, a descendent of Smalltalk.

Our plan is to have the prototype system ready to introduce in early fall. We will test the system with our teachers through comparative and/or summative evaluation to determine future research outcomes.

5. Acknowledgements

This work was supported by a grant from the National Science Foundation (NSF DGE-9553458). We would like to thank the Virginia Tech HCI group, and Alexander Repenning and the Center for Lifelong Learning at the University of Colorado Boulder.

6. References

1. Brand, C., Radar, C., Carlone, H. Lewis, C. (1998). Prospects and Challenges for Children Creating Science Models. Presented at NARST 1998 Annual Meeting, San Diego CA, 4/98.
2. Carroll, J.M., and Rosson, M.B. "Managing Evaluation Goals for Training" Communications of the ACM. July 1995, Vol.38, No.7, 40-48.
3. Gilmore, D.D., Pheasey, K., Underwood, J., and Underwood, G. "Learning graphical programming: An evaluation of KidSim," ESRC Centre for Learning Research Psychology Dept. University of Nottingham.
4. Kuyper, M. (Dissertation). Knowledge engineering for usability: Model-mediated interaction Design of Authoring Instructional Simulations. University of Amsterdam, Department of Psychology, 1998.
5. Papert, Seymour. Mindstorms: children, computers, and powerful ideas. New York: Basic Books, 1980.
6. Rosson, M. B. and Seals, C. (2000). Learning and Reuse of a Visual Programming Language, IEEE, Visual Languages Journal. September 2000.
7. Schmucker, K., A Taxonomy of Simulation Software: A work in progress. Learning Technology Review, Spring 99, 40-75.
8. Scriven, M. The methodology of evaluation. In Perspectives of Curriculum Evaluation, Rand McNally, Chicago, 1967, 39-83.
9. Soloway, E. (1998). Log on Education: No One is Making Money in Educational Software. Communications of the ACM. Vol.41, No.2. 1998,11-15.