Using Game Mechanics to Measure What Students Learn from Programming Games

Jill Denner, ETR, Scotts Valley, CA, USA
Linda Werner, University of California, Santa Cruz, CA, USA
Shannon Campe, ETR, Scotts Valley, CA, USA
Eloy Ortiz, ETR, Scotts Valley, CA, USA

ABSTRACT

Despite the growing popularity of teaching children to program games, little is known about the benefits for learning. In this article, the authors propose that game mechanics can be used as a window into how the children are thinking and describe a strategy for using them to analyze students’ games. The study involved sixty 10-14 year old students in the US who spent 10 hours learning to use the Alice programming environment, and 10 hours designing and creating their games, alone or with a partner. Forty games were coded for five game mechanics that require the programmers to think in ways that are dynamic, time dependent, or complex. The results describe the mechanics that students were most and least likely to use, and how these varied depending on whether students worked with a partner or alone. The findings contribute to efforts to assess what novice programmers learn by creating games.

Keywords: Assessment, Children, Complex Problem Solving, Creating Games, Game Mechanics, Learning

INTRODUCTION

The field of games and learning has exploded in the last decade, but most of this work focuses on game play, rather than on the creation of games. A growing number of freely available and novice-friendly game authoring tools has led to increased interest in the educational benefits of computer game programming (CGP). Despite this interest, little is known about what novices learn from CGP, due in part to assessment-related challenges. In this article, we describe a strategy for assessing whether CGP can engage students in computational thinking, which involves formulating as well as solving complex problems.

The ability to solve complex problems is a characteristic found in most lists of essential 21st century skills (Computer Science Teachers Association, 2011). Complex problem solving (CPS) involves tasks that are dynamic (each action changes the environment), time dependent, and complex (require a collection of decisions that determine later ones) (Quesada,
Kintsch, & Gomez, 2005). Efforts to study these features have focused on how students attempt to solve problems, but we argue that the design of a complex problem is particularly relevant to computer-based CPS. Designing and programming a game is what Jonassen (2000) has described as a ‘design problem’; it is ill-structured, which means the student defines the goal, the solution path, and how to evaluate the solution. The task of programming a game offers the opportunity for students to engage in CPS tasks, but it also allows them to create complex problems (for the game player).

We examine CPS in the context of middle school students making computer games. The “problem” is the situation that the game creator has formulated for the player to deal with. For example, games can include situations that are dynamic, where the task environment is a system that is influenced by and influences the player’s actions. In addition, games can include features that are time dependent; for example, the challenge of a game is increased by a time limit in which the tasks must be completed. Finally, games can be complex in that a decision or series of decisions during game play determine the available game play decisions and outcomes.

**Computational Thinking**

To assess how students design complex, computer-based problems, we draw on recent research on computational thinking (CT). Wing (2006) stated that “CT involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (p. 33). But CT is not only about solving problems, it is about designing or formulating problems (Barr & Stephenson, 2011) including creating models and simulations, and programming computer games (Lee et al., 2011). In this article, we extend this work to look at aspects of CT that involve formulating complex problems, designing systems, and understanding human behavior.

The aspect of CT that involves engaging students in the formulation of complex problems is situated within the theoretical perspective of constructionism. In this view, learning is the process of knowledge construction in the context of social and cultural participation (Kafai 2006). Learning about complexity requires students to be actively engaged in design and modeling activities (Hmelo, Holton, & Kolodner, 2000); when making a game, students define the narrative, goals, rules and choices to be negotiated by the player. The programmer must not only know how to add code, they must also consider the design features—how the player will interact with the game (Peppler & Kafai, 2007), the outcomes of player action, what feedback will result, and the goal of the game (e.g., how the player will win or lose). This process of creating interactivity requires problem posing and testing (rather than just problem solving), a process that is linked to increased flexibility in thinking, problem solving skills, and conceptual understanding (Smith & Cypher, 1999; Silver, 1994).

The second aspect of CT in game programming involves designing and thinking in terms of systems, which is a key aspect of complex problem solving (Fisher, Greiff, & Funke, 2012). Systems thinking requires knowledge of how individual parts work, as well as an understanding of how they interact to form a whole system (Shute, Masduki, & Donmez, 2010). Students must first imagine the system, then put the pieces together in a way that embodies the rules or laws (a logic) for how the system will run, including opportunities for the player to interact since the user is an integral piece of the system they are building.

The third aspect of CT involves understanding human behavior. In the field of Game Studies, the term “game mechanics” is used to describe how the player interacts with the game rules, including the sets of rules that make games interactive and fun (or not) to play. Game mechanics are the actions, behaviors, and control mechanisms that are available to the player (Hunicke, LeBlanc, & Zubek, 2004); engaging with game mechanics moves the game play along (Sicart, 2008). The game designer must engage in complex problem solving to
How do Professionals’ Attitudes Differ between what Game-Based Learning could Ideally Achieve and what is Usually Achieved


[www.igi-global.com/article/professionals-attitudes-differ-between-game/62853?camid=4v1a](www.igi-global.com/article/professionals-attitudes-differ-between-game/62853?camid=4v1a)