Chapter 2
Educational Software Design: Education, Engagement, and Productivity Concerns

Steve Ritter
Carnegie Learning, USA

R. Charles Murray
Carnegie Learning, USA

Robert G. M. Hausmann
Carnegie Learning, USA

ABSTRACT

Educational software is somewhat unique in that the goal of the software is not to facilitate use of the software itself, but to produce an impact on the user - learning - that will affect the user's behavior outside of the software. Although there are many areas where educational software designers can learn from practices in productivity and game design, there are reasons to be cautious in applying such principles to educational software. This chapter considers several design elements in educational software and discusses ways that software principles taken from other areas do or do not apply to educational software design.

INTRODUCTION

Educational software often takes its cues from productivity software and from software games. However, those categories of software often have different goals from educational software, requiring caution in applying principles derived from these other categories to educational software design. This chapter argues that educational software constitutes a category of software different from productivity software and from game software. The goals and conditions of educational software use differ enough that usability considerations sometimes differ from these other two categories. The goal of this chapter is to elucidate some ways in which educational software can benefit from lessons learned from game and

DOI: 10.4018/978-1-5225-2639-1.ch002

productivity software and some ways in which the goals of educational software are different enough that different principles need to be used.

In recent years, game designers have sought to distinguish usability principles for good game design from those espoused for normal business (often called "productivity") software (Isbister & Schaffer, 2008; Schell, 2014). The two types of software have different goals. Productivity software typically strives to allow the user to accomplish her goal in as efficient a manner as possible. Clarity and predictability are paramount. In contrast, the primary goal in game software is to entertain. Rather than trying to minimize the time the user spends in the software (or the number of clicks to accomplish a goal), the game designer's goal may be to encourage the user to stay with the software for a longer period of time and to do more "work." Clarity and predictability are not always the most important considerations. Some mystery and surprise may add to the delight of a game. The difference in goals between games and productivity software has led to differences in design styles and recommendations for the two types of software.

Educational software constitutes a third type of software, distinct from either productivity software or game software. While some principles of design overlap with principles from these domains, others remain distinct. These distinctions are driven by the different context of educational software. Like productivity software, educational software emphasizes clarity, at least with respect to educational content. However, like game software, educational software is not overly focused on reaching the goal with a minimum amount of effort. In fact, imposing effort is often part of the educational pedagogy (c.f. Bjork and Bjork, 2011).

These three types of software overlap and intermingle in interesting ways, mixing goals and methods to achieve their aims. For instance, "serious games" (aka educational games) combine the high-level goals of entertaining and educating the user. At a more detailed level, good educational software borrows techniques from productivity software to help the user efficiently get past topics that are not on the learning agenda in order to focus on topics that are. This chapter outlines ways in which educational software both differs from, and shares with, productivity and game software.

Based on over 50 years combined experience in designing adaptive educational software, the authors will propose principles of good educational software design, illustrated with examples from Carnegie Learning's Cognitive Tutor® and MATHia®. Topics discussed will include unique characteristics of educational software users, balancing helping the student with achieving educational goals, assessing the student's knowledge, transparency and "gaming" the system, the several voices of educational software, consistency vs. variety in the user interface, encouraging mistakes, and aligning the software's reward structure with educational objectives.

BALANCING USABILITY WITH EDUCATIONAL GOALS

Good productivity software tries to do everything it reasonably can for the user, even anticipating the user's needs. From automatically completing forms, to scheduling appointments based on email, to actually driving the user to the appointments, productivity software's attempts to help users complete tasks more easily and efficiently is ever-expanding.

Game software often does something similar, if more subtly. For instance, virtual worlds abstract away from messy details of reality so that the user can focus on the game action. If "eating" to have energy is not important to the game action, it is not important in the virtual world. Game software helps