Chapter XIII

Design Principles for Social Software in E-Learning

Evolution is to allegory as statues are to birds. It is a convenient platform upon which to deposit badly digested ideas. (Jones, 1999)

Introduction

E-learning environments, in which structure arises from dialogue, exhibit emergent behaviour. If this behaviour is to benefit learners, then it is important for system designers to understand what kinds of features such systems require and, more importantly, how they develop different behaviours, forms, and structures. This chapter presents eight basic principles that may be employed to ensure that the environment may develop through the actions of its users to become sustainable and self-organised.
Principle 1: Design for Change

Low Road and High Road Designs

As Stewart Brand has observed for buildings, low road freedom or high road flexibility gives a much more powerful and efficient way to survive than no road design (Brand, 1997, p. 52). Low road, architecturally dull houses which people live in, may easily grow and adapt over time, gaining extensions, new facades, being gutted then rebuilt from within, thus surviving changes in use and adapting to new uses in an almost organic way. Larger, more monumental high road buildings adapt differently, but still undergo change over many years. On the other hand, elegantly designed buildings, which perfectly fit their purpose, are in a much more precarious position once that purpose has gone. These are what Brand calls “magazine architecture.” No road designs may be beautiful, but are immutable and ultimately disposable.

It is significant that the common approach to software construction that follows a waterfall process of analysis, design, development, implementation, and evaluation is, more often than not, concerned with the creation of no road designs. Software that is built for a single purpose is the computer equivalent of magazine architecture, elegant, fit for purpose, but when that purpose changes, ultimately disposable. A social system must develop and grow with its community.

Frameworks, Mashups, and APIs

To build software that allows flexible and diverse uses, it must allow some form of adaptation and, ideally, self-organisation to occur after the system has initially been constructed. Good, modular design, ideally using object-oriented principles, is a basic requirement, but is not enough to provide great flexibility. One approach that has already been observed in the ELF framework (JISC, 2004) and the OKI architecture (Eduworks Corporation, 2002), is to build systems out of interchangeable and extensible components.

Content, too, should conform to standards. Minimally, simple but dumb standards such as HTML or XML should be used, and the hiding of content in uninteroperable databases should be avoided. If possible, more meaningful standards such as SCORM, the IEEE-LOM, and other popular learning technology standards that may be found at www.imsproject.org should be used.

It should be possible to exchange information about users, including log-in information, perhaps using standards such as Shibboleth (shibboleth.internet2.edu).

Systems should be as open as possible, allowing others to make use of their services, data, users and facilities.
A Contextualised Multi-Platform Framework to Support Blended Learning Scenarios in Learning Networks
www.igi-global.com/chapter/contextualised-multi-platform-framework-support/36069?camid=4v1a