INTRODUCTION

Patterns are widely used in several domains and are rapidly growing in numbers as an effective way of communicating knowledge between designers (Erickson, 2000). Currently, there are hundreds of HCI (human-computer interaction) patterns that are published in books and on the Internet (HCI Patterns, 2003). The sheer number of HCI patterns and the lack of a delivery system can confuse and overwhelm a novice pattern user, even when they are meant to help novice users in the first place.

BACKGROUND

The first use of patterns is often attributed to the work of Christopher Alexander (Alexander, Ishikawa, & Silverstein, 1977). It is often claimed that in order to get the essential spirit of patterns, we should take a look at his work. The next milestone is attributed to the renowned book Design Patterns (Gamma, Helm, Johnson, & Vlissides, 1995), widely acknowledged in the software design community. Plenty of other patterns emerged since then, all of which are generally targeting novice designers. They aimed to provide solutions to common design problems in different contexts. This plethora of patterns started to overwhelm the users as to how to find and use this ever-growing number of patterns. So far, most efforts have focused on “generating” more patterns, but they stop short of addressing consequent phases like delivering patterns to their appropriate destinations and helping in how to effectively integrate them in a new design artifact. These essential activities are left to the user, who would consequently follow ad hoc techniques to lookup as many patterns as they can and then try to figure out how to remove redundancies, inconsistencies, and conflicts between them before attempting to put them together in a new design. This is a typical challenge of information management that has never been addressed in the pattern community before, despite the fact that elegant, well-established concepts already exist in the database domain. Besides the efforts of adapting these concepts to the specific domain of patterns, the gap between conceptual and practical aspects needs to be bridged in the form of a software system that implements these concepts by following a structured pattern dissemination process.

THE SYSTEM DESIGN

Our system is based on an integrative approach of pattern dissemination that complements the scattered efforts of writing patterns. We built a delivery system in the form of a digital library based on a database of patterns, a transformation logic for any desired algorithms, pattern processing and transformation tools, as well as an interface to offer these functionalities to the user. We selected XML as the language to represent patterns for several reasons. XML is becoming the norm for data interchange and is used extensively in IT systems. Besides the XML language, several XML-based technologies are emerging to enhance XML-compatible information systems.

The system architecture is shown in Figure 1. The dataflow starts at the top right corner in the form of the existing pattern collections. Looking at the data pathways depicted in the figure, we see that the input information is now bypassed from the direct path (from pattern collections directly to the user preparations block, marked as A) into the system pre-storage phase, the pattern corpus, and to the rest of the system (marked as B).

The user preparations block, depicted as pathway A, represents the cognitive activity by the user to search for patterns and read through the text, analyze its contents, and figure out how to apply which patterns in the design. This is a fundamental observation in our study. We can evaluate the difference between the two processes, referred to as pathways A and B. If we considered them as representing the dissemination processes between pattern authors and pattern users, we can use the capability maturity model (CMM) of the Software Engineering Institute to briefly evaluate them.
**An XML Multi-Tier Pattern Dissemination System**

**Process A:** This process does not follow a particular approach of dissemination except for relying on users’ preparations (looking up patterns, understanding them, and applying them in an ad hoc approach). We evaluated this to be a CMM level 1.

**Process B:** As suggested by the envisioned system, there is a process in place to help users interact with patterns in a structured way. Moreover, this process relies heavily on feedback and is constantly changing, as shown in implementing and validating the 7C’s process within the system (Gaffar, Sinnig, Javahery, & Seffah, 2003, Metzker, Seffah, & Gaffar, 2003). We estimate a CMM level 4.

The main modules of the system, as shown in Figure 1, are:

- The Pattern Corpus, a raw collection of patterns before being reformatted and saved in the XMLDB.
- The Data Models, used in the semantics of the XML Rewrites.
- The Expert Preparations, used as help in rewriting pattern information to ensure the integrity of the contents and avoid redundancies and inconsistencies.
- The System Process (the 7C’s process), a structured method applied across the system.
- The XML Subsystem (XML Rewrites, XMLDB, and XML Semantics/Scheme), the core constituent of the system, and the backbone of the three-tiered design.
- The Processing and the Presentation layers are the middle and front tiers, respectively.

One of the main aspects of our system is to be able to automatically process the contents of patterns to alleviate the user from this cognitive load. The main purpose is to have a scalable capability to effectively process a large number of patterns. Representing patterns in natural language defeats this purpose. XML is much more suitable in this regard due to its machine-readability. It is based on the fundamental concept of automatic processing. Goldfarb (Goldfarb & Prescod 2004), the inventor of SGML (the parent and superset of XML), explains that the vast majority of XML documents will be created by computer programs and processed by other computer programs, then destroyed. Humans will never see them. The first step is to rewrite patterns in an XML format. A simple XML syntax rewrite (like PLML, Pattern Language Markup Language; Fincher & Finlay, 2003) can be a small step in this direction, but—by itself—it will not do much good as will be discussed later. To achieve global interoperability, we needed to design the semantics and the behavior modeling behind the XML syntax, according to concrete data models.

The database is a core constituent of the system. An obvious choice for data store is an XMLDB. We implemented part of the system using it. The steps are as follows:

- **Phase 1, No Database System:** By temporarily including the data inside the system (hard-coding the data). We have two major prototypes: In the UPADE project, we developed a systematic approach to glue patterns together, support the integration of patterns at the high design level, and automate pattern composition. UPADE (Human-Centered Software Engineering Group, 2000) generated program elements from an extensible collection of “template” patterns.

In the second prototype (Gaffar et al., 2003; MOUDIL, 2001), we experimented with prototypes of the interface aspects, and we also hard-coded patterns inside the prototypes. At this stage we were able to test and refine the design and functionality of the system.

- **Phase 2, Flat-File Database:** To start building an independent database module, external to the system, and to connect it to the rest of the system. We used XMLESpy to prototype the XMLDB concept. The XMLESpy allowed us to build an actual, partially functional XMLDB based on flat files as the internal storage medium. The major advantage of this approach is the extreme simplicity of the system, which allowed us to further refine the fundamental concepts of the system without being carried away by the implementation details. The main drawback was its limited scalability and unacceptable performance, as will be explained in Phase 4.
Related Content

Benchmarking and Data Generation in Moving Objects Databases
www.igi-global.com/chapter/benchmarking-data-generation-moving-objects/11117?camid=4v1a

Knowledge Based System and Database Management System: An Integrative Framework
www.igi-global.com/article/knowledge-based-system-database-management/51095?camid=4v1a

Effects of Graphical Versus Textual Representation of Database Structure on Query Performance
www.igi-global.com/article/effects-graphical-versus-textual-representation/51114?camid=4v1a

A Comprehensive Framework Towards Information Sharing Between Government Agencies
www.igi-global.com/chapter/comprehensive-framework-towards-information-sharing/8001?camid=4v1a