Chapter I

The Object-Oriented Design Knowledge

Javier Garzás, Oficina de Cooperación Universitaria (OCU) S.A., Spain

Mario Piattini, University of Castilla - La Mancha, Spain

Abstract

In order to establish itself as a branch of engineering, a profession must understand its accumulated knowledge. In this regard, software engineering has advanced greatly in recent years, but it still suffers from the lack of a structured classification of its knowledge. In this sense, in the field of object-oriented micro-architectural design designers have accumulated a large body of knowledge and it is still have not organized or unified. Therefore, items such as design patterns are the most popular example of accumulated knowledge, but other elements of knowledge exist such as principles, heuristics, best practices, bad smells, refactorings, and so on, which are not clearly differentiated; indeed, many are synonymous and others are just vague concepts.
Introduction

“Chaos is order waiting to be deciphered”
~ José Saramago

Twenty years ago, Redwine (1984) commented that “an expert in a field must know about 50,000 chunks of information, where a chunk is any cluster of knowledge sufficiently familiar that it can be remembered rather than derived,” adding that in mature areas it usually takes about 10 years to acquire this knowledge. Since then, many authors (Shaw, 1990) have commented on the need for defined chunks of knowledge in the software engineering field. In this regard, the software engineering community has advanced greatly in recent years, and we currently have numerous and defined chunks of knowledge, including standards, methodologies, methods, metrics, techniques, languages, patterns, knowledge related to processes, concepts, and so on.

Nevertheless, the field of software engineering is still beset by a lack of structured and classified chunks of knowledge (McConnell, 2003) and not all knowledge is transmitted, accessible or studied in the same way. For example, what and where is the enormous amount of practical knowledge regarding object-oriented micro-architectural design? We mean knowledge that has been accumulated from the experience of working with the inherent properties of software, knowledge which normally comes under what is generally accepted or “practices which are applicable to most projects, about which there is a widespread consensus regarding value and usefulness” (Bourque & Dupuis, 2004, p. A-10). Such knowledge may take the form of a source code, components, frameworks, and so on, but these are no mechanisms for obtaining designs throughout the software life cycle.

At this point, many will have already identified one of the essential items of knowledge based on experience with object-oriented micro-architectural design: design patterns. These are just the tip of the iceberg. Let us simplify matters and suppose that we want to specialize as software engineers in object-oriented design. By means of projects like SWEBOK, we can now ascertain what “design” is, how it is subdivided, find the main bibliographical references, and so on, and quite easily acquire a sound theoretical knowledge. If indeed we concentrate part of our professional activity on design, we find that we need to study the practical experience of other experts in the area, and at that moment, the concept of pattern occurs to us. Yet, after examining the main pattern references in object-oriented design, we still feel that something is missing. Missing elements for the formulation of a good micro-architectural design include principles, heuristics, best practices, bad smells, refactorings, and so on. Table 1 gives an example of each of these.

Table 1: Examples of Missing Elements for a Good Micro-Architectural Design

<table>
<thead>
<tr>
<th>Element</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principles</td>
<td>Formulation and definition of a micro-architectural design</td>
</tr>
<tr>
<td>Heuristics</td>
<td>Guidelines for the design process</td>
</tr>
<tr>
<td>Best Practices</td>
<td>Best practices for software engineering</td>
</tr>
<tr>
<td>Bad Smells</td>
<td>Common pitfalls in software design</td>
</tr>
<tr>
<td>Refactorings</td>
<td>Techniques for improving software design</td>
</tr>
</tbody>
</table>

Copyright © 2007, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.