Brady Field

Software Design: An Overview

**Important points**

“Software design plays an important role in developing a software system. During design, developers produce various models that form a kind of blueprint of the solution to be implemented. Developers can then analyze and evaluate these models to determine if they will fulfill the various requirements and to evaluate various alternative solutions and trade-offs.” [195]

“Software architecture, in fact, has been emerging as a discipline of its own, involved with the study, in a generic way, of software structures and architectures. This broader meaning of software architecture gave rise to a number of interesting ideas and concepts about software design at different levels of abstraction. Some of these concepts can be useful during architectural design (for example, architectural styles) whereas some pertain more specifically to detailed design (for example, design patterns).” [196]

“A software architecture description is a complex entity as it serves many purposes, a key one being its use for communication among the various stakeholders involved in the development of the software system.” [197]

“More precisely, the key idea behind patterns is that, over the years, software development practitioners have observed and identified a number of recurring problems and solutions. The key goal of patterns is then to describe – thus, to codify and document – those commonly recurring solutions to typical problems.” [198]

“An important goal of software design has always been to allow for the reuse of software elements. Recent approaches toward that goal are based on software product lines and software components.” [199]

“Although some qualities can be achieved through appropriate architectural choices – for example, modifiability and reusability, performance – some others cannot – for example, functionality and usability.” [200]

“Given the variety of notations available for design, a key question is how these various notations can be combined to obtain a coherent design document.” [201]

“Documenting a view involves, among other things, describing the interfaces of the elements from that view.” [202]

“Key concepts of structured design are those of coupling and cohesion, which characterize a design of good quality.” [203]

“The notion of object is intimately tied to the notions of data abstraction, encapsulation, and abstract data type (ADT).” [204]

“How objects from the various classes collaborate to provide the desired system behavior is described using interaction diagrams.” [205]

“Data-structure-oriented-design – also known as Jackson Structured Programming (JSP) is an approach in which the emphasis is on the data that a program manipulates rather than the functions it performs.” [206]

**Disagreements**

“According to ISO/IEC Standard 9126-1, software quality – defined as ‘the totality of features and characteristics of a software product or service that bear on its ability to satisfy stated or implied needs’ – can be characterized by the following six properties: functionality, reliability, usability, efficiency, maintainability, and portability.” [199]

“Although measures can be used to measure certain quality attributes – for instance, complexity metrics can be used to evaluate the testability of a software unit and to determine how much testing needs to be performed – many quality attributes are hard to quantify.” [200]

In the first paragraph, the authors cite the standard of software quality and then go on to outline various metrics. On the next page the authors do admit that there are some attributes that are hard to quantify. They also cite that complexity is not one of these. I would argue that complexity is not easy to quantify. I would further argue that indeed none of the attributes the authors selected are quantifiable. Even if they could be quantified, there would be no way to guarantee or even test the metrics.

**Questions**

I have no questions. I understood everything in the article.