Summary of Design Principles Survey

Reinhold Ploesch and Johannes Braeuer
Department of Business Informatics – Software Engineering
Johannes Kepler University Linz, Austria
reinhold.ploesch@jku.at, johannes.braeuer@jku.at

This summary describes the major results of our design principles survey. It first shows an overview of the survey design including some demographical aspects of the participants. Afterwards, a ranking of the 15 design principles is depicted. This ranking assembles all rankings made by the participants. The summary continues with a discussion of three design principles that are considered as missing by survey participants. Finally, a conclusion and an outlook for future work are given.

I. Survey Design and Overview

The survey was available online from March 16th to April 20th. During this time 104 participants completed the survey while 37 questionnaires were started but not finished successfully. This set of unfinished surveys is ignored in the analysis.

By analysing the demographical aspects of the participants, the result shows that more than 50 per cent of the participants are employed in a company with more than 1000 employees. In more detail: 2 from company < 10 employees, 13 from company < 50 employees, 12 from company < 250 employees, 12 from company < 1000 employees, 54 from company > 1000 employees, and 11 from an academic organization.

Next to the affiliation, the question regarding the current job role indicates that many software architects and developers were participating. To be more exact, 12 project managers, 4 quality managers, 37 software architects, 62 software developers, 7 software testers, 2 software support engineer, 22 consultants, and 12 scientists completed the questionnaire (multiple job roles were allowed).

Last but not least, the analysis of the engineering domains highlights a suitable distribution except of mobile systems with only one participant. All in all, 23 participants are working on web/service oriented systems, 14 on embedded systems, 17 on development tools, 25 on business information systems, 1 on mobile systems, 8 on expert and knowledge based systems and 16 on a system from another (unspecified) domain.

II. Final Ranking

Table I shows the final result that composes all rankings. The weighted rank next to each design principle is calculated based on the rank given by a participant. In other words, when a design principle was ranked on the first place, ten points were added to its weighted rank; nine points were added for rank two, eight points for rank three and so on.

Based on that final ranking, we tried to figure out whether different opinions exist within different job positions, domains or programming languages. Therefore, the data was clustered according to these viewpoints and the clusters were compared with the composed ranking. As a conclusion, no ranking showed a significant difference.

TABLE I. Ranking of Design Principles

Design Principle	*weighted Ranks*
Single Responsibility Principle	695
Separation of Concern Principle	647
Information Hiding Principle	611
Don't Repeat Yourself Principle	535
Open Closed Principle	459
Acyclic Dependency Principle	384
Interface Segregation Principle	378
Liskov Substitution Principle	365
Self-Documentation Principle	332
Favour Composition over Inheritance Principle	326
Interface Separability	295
Stable Dependencies Principle	195
Law of Demeter	188
Command Query Separation	174
Common Closure Principle	136

III. Missing Design Principles

The survey was also asking about missing design principles by means of an open question. The following three principles were selected as the most important ones that were missing.

A. Dependency Inversion Principle (DIP)

13 answers referred to the DIP as one of the five SOLID principles. Many opinions of the participants stress the power of DIP in breaking cyclic dependencies and foster a loose coupling of software modules. Consequently, it supports software development in re-using software components and optimizing modularity.

B. KISS (Keep It Simple & Stupid) Principle

In eight surveys the KISS principles was mentioned as missing. KISS concentrates on reducing complexity and building software that is as simple as possible, but still meets the requirements of stakeholders. By reducing complex constructs, it is possible to create a common code ownership that supports the development of comprehensive solutions.

C. YAGNI (You Ain't Gonna Need It) Principle

The third principle is YAGNI that is referenced in six answers. YAGNI has a similar goal like KISS that focuses on building solutions, which are not overloaded with unnecessary functionality.

IV. Conclusion and Future Work

This survey provides a good understanding of the relevance of design principles in practice. Based on that result, we will continue to focus on the most important design principles and establish approaches to operationalize them. For validating these approaches, qualitative and quantitative assessments will be conducted on industrial projects.