APPLICATIONS ARCHITECTURE ANALYSIS BASED ON DESIGN PATTERNS AND IMAGE RECOGNITION

Kopp A.M., Orlovskyi D.L., Ersoyleyen D.
National Technical University
«Kharkiv Polytechnic Institute», Kharkiv

Introduction. In the recent decades a software engineering industry has changed sufficiently. Multiple approaches and practices of the software development appeared and outdated in literally several years. However, staring from the waterfall model and up to modern agile practices, applications architecture design is the essential phase of the software development lifecycle. Several researches, one of which is by IBM, have concluded that design errors made at the early stages of software development result into significant monetary losses (see Fig. 1). These losses are caused by efforts and resources required to fix detected errors on testing or maintenance stages of software product lifecycle, while fixing such errors on system design stage may cost 15x–100x times cheaper [1]. Thus, applications architecture analysis is considered in this study.

Relevance. According to [2], application architecture describes design patterns (reusable building blocks used to design and develop software products). Design blocks described by patterns are used by architects to define baseline of the system and main principles of its functioning. Architecture patterns are different from software design patterns, since they define whole application scope unlike the particular module or source code unit. However, architecture patterns have their own advantages and disadvantages, which define usage scopes of some patterns depending on project size, industry, and environment. Sometimes designed solutions seem well on blueprints and previously hidden drawbacks become noticeable too late. Therefore, analysis of applications architecture design becomes relevant problem that should be solved at early design stage in order to consider all possible limitations and make decisions on applications architecture re-design or improvement.

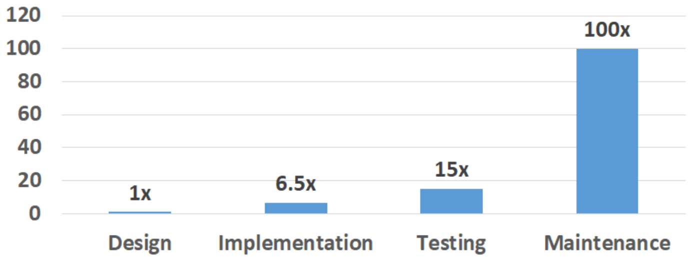

Fig. 1. – Defects fix costs growth by software product development stages

Proposal. There could be defined “images” for each architecture design pattern, e.g. using *n*-space vectors. Structural metrics of design patterns (e.g. size, complexity, coupling, and cohesion) may be used as vector’s elements. Then, architecture models are compared to images of patterns using recognition techniques and limitations are suggested to system architects with respect to detected design patterns [2].

References:

1. Defect Prevention: Reducing Costs and Enhancing Quality // URL: https://www.isixsigma.com/tools-templates/software/defect-prevention-reducing-costs-and-enhancing-quality/

2. 10 Common Software Architectural Patterns in a nutshell // URL: https://towardsdatascience.com/10-common-software-architectural-patterns-in-a-nutshell-a0b47a1e9013