Exercise 2 – Specification by Example Reflections

In Specification by Example (2011), Gojko Adzic covers a collaborative method for specifying requirements and tests. Key process patterns ensure the team shares the same view of the developed product by specifying it through clear examples. He presents seven key process patterns.

Seven Key Process Patterns

1. Deriving Scope from Goals

The business users communicate the solution's intent and value, while developers suggest an easier solution to develop and maintain. By involving the developers in defining the scope the project benefits from a diverse skill set from the beginning.

2. Specifying Collaboratively

As developers and testers contribute to the requirement gathering, the risk of misunderstanding is minimized. This creates a sense of collective ownership, engaging the team in the delivery process.

3. Illustrating Using Examples

Natural language has an ambiguous nature and can be interpreted differently by different people. Adzic proposes that teams should specify by using examples to be successful. Key examples are identified together with the business users. When the system works correctly for all the key examples, it meets the specification.

4. Refining the Specification

Often too many details in the specification constrain the development process. Thus, the team should focus on what the functionality of a system is, rather than how it works. The refined examples can be used as acceptance criteria.

5. Automating Validation Without Changing Specifications

Development requires quick feedback, which requires automation of the acceptance criteria. Requirements should exist only in one format; the terms test and specification are used interchangeably by Adzic. The specification is thus executable.

6. Validating frequently

As the specification that has been developed at this point is executable, they are easy to validate against the system to make sure the development is on track to build the right product.

7. Evolving a Documentation System

Specifications should be well-organized and updated according to business and market changes. Maintaining and updating the specifications keeps the documentation relevant.

Reflections on Behavior-driven Development

Behavior-driven development (BDD) uses the behavior of the software system to formulate the specifications by using a Given-When-Then structure. The set of examples can be used as both a high-level specification and a suite of acceptance tests.

In the article Maintaining Behavior Driven Development Specifications: Challenges and Opportunities (2018) Binamungu et al. explore how software development organizations use BDD, the main benefits and challenges.

Some of the benefits are similar to the methods Adzic proposes, such as involving members of different teams in the specification process. BDD requires collaboration, which is made easier by having specifications that are expressed in use cases. Some respondents did report that it was still challenging to engage different teams in the specification work.

There is a risk that BDD is seen as merely test automation, which results in omitting most of the benefits of the BDD process and failing projects. These benefits include living documentation that is easy to grasp for non-technical people as well and enables the QA team to write code without access to the implementation. The documentation is also in code format.

Many reported issues in maintaining the BDD specifications. In addition to facing similar issues to test automation, some developers experienced duplication with unit tests. BDD specifications can have the presence of duplication within themselves too. It would be interesting to gain more insight into found best practices in maintaining the specifications, and if Cucumber is used, the feature files and the step definition files.

BDD seems to offer many benefits, while also posing some new challenges. There are similarities in the proposed methods of both Adzic and Binamungu et al., in particular in the collaboration efforts, use-case thinking, and striving for living documentation.