

# TestCockpit: Business Intelligence for Test Management

Stefan Larndorfer and Rudolf Ramler<sup>1</sup>

*Software Competence Center Hagenberg GmbH, Softwarepark 21, A-4232 Hagenberg, Austria  
{stefan.larndorfer, rudolf.ramler}@scch.at*

## 1. Introduction

Up to 50 percent and more of the costs of software projects account for testing activities. Software development is often under pressure to meet high-quality standards and, at the same time, to deliver in a tight schedule and budget. Under these circumstances it makes sense to ask: How can software testing be made more effective and more efficient?

Besides making sure that the test process itself is carried out efficiently by using tools that actively support the testing activities (e.g., test case administration tools or bug tracking systems), it is as important to support the activities that determine what, how and to which extent parts of a software system are tested. These activities fall into the scope of test management [3] and involve hard decisions and tradeoffs. Thereby, in a typical software project, test managers are faced with the following situations:

- A typical software project uses a broad variety of different software tools. Each of the tools contains only a fraction of the information relevant to plan and control testing activities. Thus, the test manager is usually confronted with poorly integrated data from various sources at different level of detail.
- To extract the relevant information, the test manager has to handle huge amounts of data. For example, test results from periodical test execution in different environments have to be evaluated for upcoming testing cycles.
- When it comes to decisions in test management, personal experience and intuition play a dominant role. There is often not enough time to prepare all relevant information for structured decision making. As a result of too many influencing factors that can not be objectively overviewed, decisions are often made intuitively.

- In general, software testing is handled very differently in various environments. There is no common established test process or tool set that is used across different project types. The transfer of experience and knowledge between projects is typically restricted to a personal level.

Test managers and business decision makers have something in common: Both have to make impacting decisions in a dynamic, time-driven environment assessing information from different data sources across the organization. In terms of business decisions, business intelligence solutions have already become an important support tool. Cockpits (also known as dashboards) support three main sets of functionality [1]: Monitoring of critical operations using metrics, analyzing the root causes in case of problems, and managing people and processes to improve decisions and to optimize performance. Up to now, test management still is missing systematic support at this level.

The objective of this paper is to describe a concept for a cockpit aggregating and visualizing key performance indicators about software testing to support decision making in test management. This paper is structured as follows: After presenting the features of a test cockpit from the user's perspective in Section 2, the technical solution concept is proposed in Section 3. In the final section the status of the project and the next steps are described.

## 2. Features of a Test Cockpit

The features of a cockpit supporting test management can be derived from the related theory of business intelligence. Following key features are highly beneficial for test management:

- *A test project overview* visualizes product quality metrics from static and dynamic testing combined with process and project management via charts, diagrams or gauges and widgets such as traffic lights, speedometers, or level indicators.

---

<sup>1</sup> This work has partially been conducted within the competence network Softnet Austria ([www.soft-net.at](http://www.soft-net.at)) and funded by the Austrian Federal Ministry of Economics (bm:wa), the province of Styria, the Steirische Wirtschaftsförderungsgesellschaft mbH (SFG), and the city of Vienna in terms of the center for innovation and technology (ZIT).

- *Drill down and data filtering* allows zooming in on selected aspects along dimensions like priority, time span, system module, version, or configuration.
- *Trend analysis* shows the data series of selected aspects over time, e.g. as graph, and supports the prediction of future events.
- *By benchmarking* the data of different ongoing or historical projects is compared side by side.
- *User defined ad-hoc queries* are the entry point for advanced data analysis like data mining and what-if simulations.
- *A report generator* creates compelling, printable reports in conformance to documentation standards.
- In addition, *planning* is supported by a visual editor for expected values and views that compare planned and actual data.

Related solutions with similar functionality (e.g., Mercury Quality Center Dashboard) are usually additions to tools for issue tracking or continuous integration, and are therefore restricted to a single or a fixed set of data sources. They do not offer advanced data analysis methods and are hard to customize to the actual needs of a team in a specific project. In contrast, the concept of the proposed cockpit is to provide a platform that can be used to integrate user-defined test tools as data sources and harness them to implement custom software metrics and models [2], which are visualized in the cockpit. By virtue of this approach, the test cockpit can serve as a comprehensive decision support system for test management in various different project environments.

### 3. Technical Solution Concept

As shown in Figure 1, the system architecture of the proposed TestCockpit consists of three main tiers (from bottom to top):

(1) **Data adapters** periodically extract relevant data from different data sources actively used in the project, e.g., the issue database of Bugzilla, the change log of CVS, or the results from static code analyzers. The data is transformed to a standard data structure and stored in the central data warehouse.

(2) **The data warehouse** organizes the data as cubes amenable for on-line analytical data processing. The data schema supports managing multiple projects at the same time and defines a common set of comparable attributes for benchmarking across projects.

(3) **The user interface** of the TestCockpit visualizes aggregated information and offers the flexibility to customize views, metrics and models. The user interface is part of the TestCockpit platform based on the Eclipse RCP and, thus, extendable with additional

plug-ins. It can be used as stand-alone application or integrated in the Eclipse development environment.

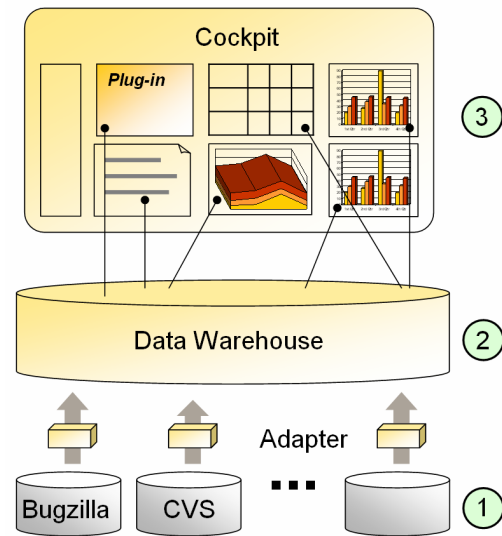


Figure 1: System architecture of the TestCockpit

### 4. Status and Future Work

At present, we are developing a prototype that implements a key scenario with a minimal feature set for the planning and controlling activities based on defect data. Therefore we currently support the visual representation of defect arrival curves [2] for different time spans, severity levels and system components. The data is extracted from the widely-used issue tracking system Bugzilla. Furthermore, future scenarios can be planned by defining expected defect arrival curves, which are then compared to the actual curves.

In a next step we evaluate the prototype in industrial projects. Future work contains the implementation of additional software metrics and models that integrate additional data sources, e.g., test case definitions and test results. The implementation of the prototype will be published as open source to encourage involvement of interested parties from industry and academia.

### 5. References

1. Eckerson W.W.: *Performance Dashboards: Measuring, Monitoring, and Managing Your Business*, Wiley, 2005
2. Kan S.H.: *Metrics and Models in Software Quality Engineering*, Addison-Wesley, 2003.
3. Spillner A., Linz T., Rossner T., Winter M.: *Software Testing Process: Test Management: A Study Guide for the Certified Tester Exam ISTQB Advanced Level*, Rocky Nook, 2007