# Evaluating Software Project Control Centers in Industrial Environments

M. Ciolkowski, J. Heidrich, J. Münch
*Fraunhofer IESE*
*Fraunhofer-Platz 1*
*67663 Kaiserslautern, Germany*
*marcus.ciolkowski@iese.fraunhofer.de,*
*jens.heidrich@iese. fraunhofer.de,*
*juergen.muench@iese.fraunhofer.de*

F. Simon
*SQS AG*
*Stollwerckstraße 11*
*51149 Köln, Germany*
*frank.simon@sqs.de*

M. Radicke
*BTU Cottbus*
*PO Box 10 13 44*
*03013 Cottbus, Germany*
*mathias.radicke@informatik.*
*tu-cottbus.de*

## Abstract

*Many software development organizations still lack support for detecting and reacting to critical project states in order to achieve planned goals. One means to institutionalize project control, systematic quality assurance, and management support on the basis of measurement and explicit models is the establishment of so-called Software Project Control Centers. However, there is only little experience reported in the literature with respect to setting up and applying such control centers in industrial environments. One possible reason is the lack of appropriate evaluation instruments (such as validated questionnaires and appropriate analysis procedures). Therefore, we developed an initial measurement instrument to systematically collect experience with respect to the deployment and use of control centers. Our main research goal was to develop and evaluate the measurement instrument. The instrument is based on the Technology Acceptance Model (TAM) and customized to project controlling. This article illustrates the application and evaluation of this measurement instrument in the context of industrial case studies and provides lessons learned for further improvement. In addition, related work and conclusions for future work are given.*

## 1. Introduction

The complexity of software development projects requires efficient ways to organize project control. Software is often developed in different distributed locations, which, in turn, makes it even harder to develop systems or services according to plan (i.e., matching time and budget constraints), because data has to be integrated into one view to get a complete picture of the current project state. Thus, support for project management and control is one of the key factors for project success [8]. Many organizations have problems with establishing efficient project control processes. One of the reasons is that projects typically have to fulfill a set of goals whose interrelationship is not clear. This complicates the tracking of goal fulfillment. In addition, there is typically no systematic process for determining key performance indicators for project success and deriving concrete measures from project goals, which makes it very hard to determine which measurement data to collect. Furthermore, it is often not clear how to interpret the collected data, which makes it more difficult to determine appropriate response actions in case of project plan deviations. One well-proven way to document the relationship between goals and data to be collected is the Goal Question Metric (GQM) paradigm [2], [15]. However, interpretation and visualization of the collected measurement data in a goal-oriented way (along the GQM plan) is also needed in order to relate them to statements about a previously defined goal.

We define a Software Project Control Center (SPCC) according to [13] as a means for process-accompanying interpretation and visualization of measurement data. An SPCC is comparable to a control room, which is a well known term in the mechanical production domain. It is used there as a central node for all incoming information of a production process. An SPCC has to be customized with respect to project-specific goals and collects, interprets, and visualizes measurement data in order to provide context-, purpose-, and role-oriented information for all parties involved (e.g., project managers) during the execution of a software development project. This includes, for instance, monitoring defect profiles, detecting abnormal effort deviations, code quality management, cost estimation, and cause analysis of plan deviations.

Reported experience in the literature with respect to

0-7695-2886-4/07 $20.00 © 2007 IEEE
DOI 10.1109/ESEM.2007.51

setting up and applying such control centers in an industrial environment is very limited. Therefore, an empirical instrument to systematically collect experience with respect to the deployment and use of control centers was developed, which aims at eliciting users' needs and identifying opportunities for improvement. The development of the evaluation instrument and its maturation is performed in the context of the so-called Soft-Pit research project funded by the German Federal Ministry of Education and Research (*http://www.soft-pit.de*)[1]. The project focuses on getting experience and methodological support for operationally introducing control centers into companies and projects. The Soft-Pit project includes the conduction of several industrial case studies with four different German companies, in which control centers and their deployment are evaluated. These case studies are clustered into so-called iterations (i.e., sets of case studies that are performed in parallel) so that learning about the technology and deployment processes can be done from iteration to iteration.

The goal of this paper is to illustrate the application of the evaluation instrument we developed for control centers, combining quantitative and qualitative approaches. The evaluation instrument was applied in six case study projects in four different domains that formed the first Soft-Pit iteration. All case studies ran for about two months, during which the control center under examination was applied. It is planned to improve the evaluation instrument based on lessons learned from this application and to use the improved version in the next set of case studies.

The article is organized as follows: Section 2 discusses related work in the field, Section 3 describes the industrial case studies, Section 4 derives lessons learned with respect to the evaluation instrument and the control center application, and, finally, Section 5 presents a brief conclusion and discusses future work.

## 2. Related Work

The following section presents related work. The first part addresses developing an evaluation instrument, in particular the Technology Acceptance Model (TAM). The second one discusses the evaluated object (Software Project Control Centers). The third one addresses experience already gathered with measurement-based project control approaches.

### 2.1. Technology Acceptance Evaluation

The Technology Acceptance Model (TAM) is an information systems theory that models how users come to accept and use a technology. The model suggests that when users are presented with a new software package, a number of factors influence system usage (i.e., their decision about how and when they will use it), notably perceived usefulness and ease of use [7]. Here, *perceived usefulness* is defined as "the degree to which a person believes that using a particular system would enhance his or her job performance" [7]; that is, a tool is useful if it enhances the user's productivity. *Perceived ease of use* is defined as "the degree to which a person believes that using a particular system would be free of effort" [7]; that is, it is easy to use if the user can quickly learn to use it. Perceived usefulness and ease of use are each measured via a set of questions (called scale; six in Davis' original version). Each of the scale items is rated on a seven-point Likert scale ("extremely/quite/slightly unlikely, neither, slightly/quite/extremely likely"). The aggregate score for perceived ease of use and usefulness is computed by summing the scores of the associated questions.

The measurement instrument has been validated (in particular, in terms of its robustness and validity), for example by Davis [7], or Adams et al. [1]. The score for the aggregate metrics perceived usefulness and ease of use is defined as the average score of each answer. Different versions of TAM have been proposed, and have been consolidated by the Unified Theory of Acceptance and Use of Technology (UTAUT), published by Venkatesh et al. [22]. It combines different existent variants of TAM, and was found to outperform each of the individual models. For the purpose of our investigation, however, we decided to use Davis' original model as a starting point, since in terms of usefulness and ease of use, UTAUT largely relies on Davis' model [22].

### 2.2. Software Project Control Centers

An overview of the state of the art in Software Project Control Centers can be found in [13]. The scope was defined as generic approaches for on-line data interpretation and visualization on the basis of past experience. Project dashboards were not included in this overview. In practice, many companies develop their own dashboards (mainly based on Spreadsheet applications) or use dashboard solutions that provide a fixed set of predefined functions for project control (e.g., deal with product quality only or solely focus on

[1] Grant number: 01ISE07A.

project costs) and are very specific to the company for which they were developed.

Most of the existing, more generic approaches for control centers offer only partial solutions. Especially purpose- and role-oriented usages based on a flexible set of techniques and methods are not comprehensively supported. For instance, SME (Software Management Environment) [10], [12] offers a number of role-oriented views on analyzed data, but has a fixed built-in set of control indicators and corresponding visualizations. The SME successor WebME (Web Measurement Environment) [20] has a scripting language in order to customize the interpretation and visualization process, but does not provide a generic set of applicable controlling functions. Unlike Provence [11] and PAMPA [17], the approaches Amadeus [16] and Ginger2 [21] offer a set of purpose-oriented controlling functions with a certain flexibility, but lack a role-oriented approach to data interpretation and visualization. There also exist lightweight SPCC implementations (e.g., [6]) that concentrate on core metrics, which are adapted to the corresponding project environment. The G-SPCC approach [9] offers a generic framework with support for goal-oriented selection of reusable controlling components and was used as a basis for implementing a control center in the context of the Soft-Pit project.

### 2.3. Measurement-Based Project Control

Most experiences with measurement-based project control are specific to the underlying entities that are measured. In general, these entities can be classified as follows:

- Products: Here, the measurement-based project control collects measures focusing on the developed product itself. Typical attributes that are indirectly measured are reliability, performance, and maintainability. Specific project control centers exist that correspond to each of these entity-related attributes. For example, reliability can be continuously controlled by commercial test dashboards (e.g. [19]), performance is monitored by visualizing the history of runtime-data (e.g. [3]), and maintainability can be monitored in code control cockpits based on commercial analysis tools ([14]). However, only very limited experience exists in integrating these different dashboards into one central dashboard, even though all are focusing on the product.
- Processes: Here, the underlying process is measured, e.g., by analyzing derivations of planned milestones, cost overruns, or activity dependencies. Measurement-based project control mechanisms focusing on this entity are often based on MS-Project®, which offers analysis techniques like Gantt diagrams and Pert charts.
- Resources: Here, the involved resources are controlled. Typical project control focuses on technical resources (such as net monitors or CPU usage-monitors). New project control centers also focus on measuring the productivity of development teams or even single developers. This can be done by relating the developed functionality (often measured by backfired Function Points) to specific developers (cf. the productivity assessment features available at CAST, [5]).

There exists very limited reported experience in having one project control center integrate these different views. Most are very project-specific (cf. [18]) and do not have a universal underlying framework. Therefore, most control centers are very specific to different roles (like performance engineer or CFO).

## 3. Industrial Case Studies

All case studies have been conducted at four different German companies that applied the control center developed as part of the Soft-Pit project on different projects. The Soft-Pit project follows an iterative approach and will be conducted in three iterations that will conceptually and technically add new features to the control center implemented. For the first iteration, a J2EE-based control center tool, named Specula, developed at Fraunhofer IESE, was used in combination with SISSy (*http://sissy.fzi.de*), a tool for measuring code quality, and a corresponding front end, which was developed at BTU Cottbus. The case studies were conducted according to the following steps (based on the Quality Improvement Paradigm [2]):

Step 1 - Characterize: Managers from each company determined projects for which the control center should be applied. Table 1 presents the application domain and the number of developers *D* for a case study project *P* of company *C*.

**Table 1. Project overview.**

| *P* | *C* | *Application Domain* | *D* |
|---|---|---|---|
| P1 | C1 | web application development | 4-5 |
| P2 | C1 | web application development | 4-5 |
| P3 | C1 | web application development | 4-5 |
| P4 | C2 | data base development | 10 |
| P5 | C3 | information system dev. | 2-3 |
| P6 | C4 | web application development | 1-2 |

Step 2 - Set Goals: For the first Soft-Pit iteration, three control goals were predefined: (1) Effort Consumption (control effort for all activities of the project), (2) Schedule Completion (control the completion state of all activities compared to their schedule), and (3) Code Quality (control the quality of source code). These goals will be extended for the next Soft-Pit iterations.

Step 3 - Choose Process: Based on those control goals, concrete measurement goals (that make control goals operational through measurement) have been derived and broken down into concrete metrics using the Goal Question Metric paradigm [2]. Visualizations were defined in order to explicitly specify how to interpret and visualize the gathered measurement data according to the goal definition [9]. After that, the control mechanisms were customized to the different case study projects. This included: (a) selection of a subset of control goals, (b) definition of a concrete measurement plan to define which person has to collect and use the measurement data at which point in time or before/after which activity of the project, and (c) adaptation of mechanisms to the development process and the project plan. Table 2 gives an overview of which project focused on which control goal.

**Table 2: Projects and control goals.**

| *Control Goal* | *P1* | *P2* | *P3* | *P4* | *P5* | *P6* |
|---|---|---|---|---|---|---|
| Effort Consumption | X | X | X | | X | X |
| Schedule Completion | X | X | X | | X | X |
| Code Quality | X | X | | X | X | |

For each company, a coach was assigned who was familiar with the control goals and measurement plan and supported the people involved in doing the local setup and application of all control mechanisms. This included, but was not limited to, testing system installation, training of project participants in using the control center, checking data validity, and collecting feedback during and after the enactment of the case studies.

Step 4 - Execute: For a previously defined evaluation period (about two months per project), the control center was used for each project of the case studies. For each control goal, measurement data was collected, interpreted, and visualized. The visualization of data for a specific goal and stakeholder is called a control view. The coaches checked the collected data as well as whether the control views were used regularly.

Step 5 - Analyze: After the end of the evaluation period, the evaluation instrument was used to gain feedback from all participants. The results were analyzed and presented to the companies' managers (who had selected the projects in the beginning) and were forwarded to the project participants afterwards.

Step 6 - Package: The gathered experience was packaged in order to be used for the next iteration.

## 3.1. Case Study Design

The main goals for the case studies were to evaluate the *ease of use* and *usefulness* of a control center from the viewpoint of different stakeholders (project and quality managers, and developers), and the *improvement potential* of the control center in the context of the Soft-Pit project. The control center offered different views, corresponding to control goals: Effort consumption, schedule consumption, and code quality view. In addition, an administration view allowed configuring and setting up the control center. As described previously, a control center was used in the case studies to monitor and control project data. Control center users were project or quality managers, as well as some developers (depending on the control goals chosen).

The research questions were mainly motivated by the need to find out how to improve a control center to increase acceptance and use in subsequent case study iterations. We defined the following constructs:

- **Ease of use** is defined as the degree to which a person believes that using a particular system would be free of effort, in accordance with Davis' TAM [7].
- **Usefulness** is defined by Davis [7] as the degree to which a person believes that using a particular system would enhance his or her job performance. In our case, we decided to define "job performance" (and, thus, usefulness) more precisely as the degree to which a person believes that a control view provides support for effective project control.
- **Improvement potential** is defined as the information that would be required in addition to the information already provided by the control center for effective project control from the viewpoint of different stakeholders.

We developed an evaluation instrument consisting of a questionnaire comprising several scales (i.e., sets of questions) to address each of the constructs. We evaluated the reliability of the questionnaire scales through Cronbach's alpha. This value indicates the extent to which a set of scale items (i.e., questions) can be treated as measuring a single construct (or latent variable, such as usefulness). Cronbach's alpha can be thought of as describing the overall consistency of the scale by addressing how much each scale item is correlated with every other item; that is, the extent to which

high responses go with highs and low responses go with lows across all items of a certain construct. For this reason, a reliability coefficient computed in this manner is also referred to as the internal-consistency reliability. As a rule of thumb, a reliability of .70 or higher is considered as sufficient; some prefer .80 for widely used scales [4]. Thus, in our case, we set .70 as the threshold for sufficient reliability of the scales.

Scale reliability was basically computed on two levels: for pooled answers (to increase the valid number of responses for each scale), and separately for each control view. For the analysis of pooled answers, we combined the answers for all views except the administration view, as this view used slightly different scales. Since the "look and feel" for each of the different controlling views is similar, this should not be a problem for ease of use. However, since usefulness may be perceived differently for each of the different views, we expect that the pooled reliability for usefulness may be low.

Usefulness and ease of use were compared for each view, based on mean scores for each scale. In addition, we conducted a qualitative analysis of answers in order to triangulate the evaluation of usefulness and ease of use, and to identify additional need for data and visualizations.

### 3.2. Evaluation Instruments

We adapted and modified the Davis' Technology Acceptance Model to address usefulness and ease of use, and we defined a set of open questions in addition in order to identify improvement potential. We developed an evaluation instrument (i.e., a questionnaire) to systematically capture the constructs for different stakeholders and the different control views of the control center. Therefore, we developed a scale with a set of questions for each construct. Usefulness, ease of use, and improvement potential were gathered for every control view applied in the project (addressing the previously mentioned control goals: effort consumption, schedule completion, and code quality). In addition, one similar set of questions was defined for the control center as a whole, and for the administration viewpoint (i.e., for setting up and configuring it). We asked participants to provide the following information:

- **Information on participant**: We asked about the role in the organization or project, and for each view, about the frequency of control view usage.
- **Ease of use**: Ease of use is defined as the degree to which a person believes that using a particular system would be free of effort. We adapted the corresponding TAM questions, reducing them to four questions, as the tasks supported by the control center were not complex enough for some questions to make sense[2] (cf. Table 3). Questions were stated as hypotheses that needed to be confirmed or rejected by the participants.
- **Usefulness**: TAM's definition of usefulness is focused on productivity. In our case, usefulness rather means that the control center provides useful information for project control. In a full-blown case study, this could be measured in terms of the project risks or deviations that could be identified using the control center. However, since the case studies described here did not cover a complete development cycle, usefulness had to be subjectively rated in terms of perceived benefits for project control. That is, usefulness in this context can be defined as the degree to which a person believes that a control view provides support for effective project control. Therefore, we defined five questions to cover potential benefits of control centers, each of which were ranked on a five-point Likert scale (Table 3).

**Table 3. Questions used.**

| *ID* | *Question* |
|---|---|
| | *Ease of use* |
| C1 | Using the view is easy to learn. |
| C2 | It was easy to become skillful using the view. |
| C3 | Interaction with the view is clear and understandable. |
| C4 | The view is easy and intuitive to use. |
| | *Usefulness* |
| B1 | The view allows early detection of plan deviations. |
| B2 | The view allows complete detection of plan deviations. |
| B3 | Using the view facilitates identification of causes for plan deviation. |
| B4 | The view enables me to gain a holistic view on project control. |
| B5 | The view is a useful tool for project control. |
| | *Improvement potential* |
| X1 | Perceived benefits of control views. |
| X2 | Problems in using the control center. |
| X3 | Missing information for project control. |

- **Improvement Potential**: In addition, respondents could describe several potential benefits of the control center that they perceived. We defined a set of open questions aimed at identifying weaknesses in the control center implementation (i.e., its usability)

[2] Note that the original scale items were phrased in German.

as well as about useful information for project control that was missing (cf. Table 3).

### 3.3. Results of the Case Studies

A questionnaire was sent out to all participants of the case study after the end of the evaluation period. Table 4 gives an overview of the number of responses for each project and the main roles of the participants who sent back a questionnaire. In total, 11 questionnaires were received. The job experience of the participants was about 7 years on average (with a standard deviation of 4 years and a standard error of .5 years). Control center usage ranged from three times a week to once a month.

**Table 4. Overview of questionnaire responses.**

| *P* | # | *Roles* |
|---|---|---|
| P1 | 2 | Project Manager, Developer |
| P2 | 1 | Project Manager |
| P3 | 1 | Project Manager |
| P4 | 3 | Project Manager, Developer, Administrator |
| P5 | 3 | Project Manager, Developer, Management |
| P6 | 1 | Project Manager |

With respect to the control center's usability and ease of use, we conducted a box plot analysis for each of the control center's views: effort consumption, schedule completion, code quality, general control center application, and control center administration. For computing the aggregate score for usefulness and ease of use, we decided to compute the mean value instead of the sum of all values. We did this because the number of questions was different for the two scales. This is in general possible because summative Likert-scales are usually interpreted as interval or ratio data measuring a latent variable. Figure 1 shows the mean, 25-75% quartiles, and total range for usefulness of the different views. Here, the mean value lies between 3 and 4; that is, between a neutral rating of the usefulness and a slightly positive rating. The same holds for the most probable area of answers (the interval between 25% and 75% quartiles).

The analysis for the ease of use of the control center is shown in Figure 2. In this case, more heterogeneous results were achieved than for usefulness. The effort consumption view was rated slightly positive, schedule completion was neutral, and code quality was seen as very easy to use. The general control center application was slightly positive, whereas the ease of use of the control center administration was perceived as slightly negative.

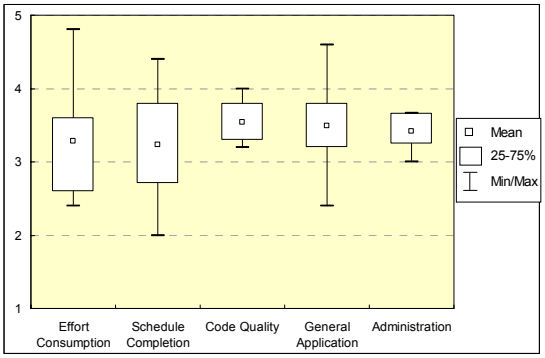


**Figure 1. Box plots for usefulness.**

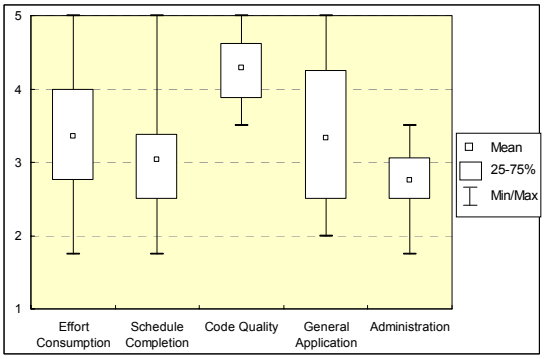


**Figure 2. Box plots for ease of use.**

Table 5 presents a summary of all quantitative results with respect to usefulness and ease of use. The mean value is presented, as are the standard deviation (in brackets) as well as the numbers of questionnaires *N* that had valid answers for the respective control view.

**Table 5. Summary of usefulness and ease of use evaluation.**

| | *Usefulness* | *Ease of Use* | *N* |
|---|---|---|---|
| *Effort Consumption* | 3.29 (.86) | 3.36 (1.07) | 7 |
| *Schedule Completion* | 3.23 (.85) | 3.04 (1.09) | 7 |
| *Code Quality* | 3.54 (.32) | 4.29 (.55) | 7 |
| *General Application* | 3.49 (.67) | 3.33 (1.09) | 9 |
| *Administration* | 3.42 (.30) | 2.75 (.53) | 8 |

The questionnaire also contained parts retrieving more qualitative feedback. Table 6 and Table 7 summarize those results with respect to the general control center application and administration aspects, respectively. Benefits and problems as well as currently missing aspects mentioned in the questionnaires are listed.

**Table 6. Summary of qualitative results for general control center application.**

| | |
|---|---|
| *Benefits* | More control over the project.<br>Central project data processing point.<br>Simple presentation of project state.<br>Early detection of plan deviations / problems.<br>Helps to stay within project plan.<br>No manual controlling needed. |
| *Problems* | Tool interface inhomogeneous.<br>Interpretation of diagrams difficult.<br>Takes a lot of time to interpret data.<br>Manual import of data requires time.<br>Scalability of diagrams.<br>Traceability to deviation causes.<br>Importance of control indicators hard to know. |
| *Missing Information* | Potential deviation causes.<br>Easy project overview (e.g., traffic lights).<br>Historical progression of data series.<br>Milestone adherence.<br>Resource usage.<br>Trends of time series.<br>More controlling aspects (holistic control center). |

**Table 7. Summary of qualitative results for control center administration.**

| | |
|---|---|
| *Benefits* | Organization-wide accessibility.<br>Customizability.<br>Extendibility.<br>Simple integration of existing data.<br>Fast information access.<br>Standardized controlling components. |
| *Problems* | Prototype installation and maintenance.<br>Consistency of imported data. |
| *Missing Information* | Data consistency evaluation.<br>Integration of existing systems. |

Finally, we want to have a look at the evaluation of the empirical instruments used. When pooling answers from all ease of use and usefulness questions, we receive 30 data points for each. Cronbach's alpha of .89 for ease of use with an average inter-item correlation of .71, and .59 for usefulness, with an average inter-item correlation of .26, indicates good reliability for ease of use, but some reliability problems for usefulness. Table 8 displays Cronbach's alpha and average inter-item correlation, as well as the total numbers of valid questionnaires *N*. For code quality, an exceptionally low reliability can be observed for usefulness, although the ease of use items are quite reliable. For control center administration, Cronbach's alpha is low for usefulness as well as ease of use. For all other views except administration, usefulness and ease of use are measured quite reliably.

## 3.4. Discussion of Results

As seen in Figure 1, people perceived the usefulness of the initial Soft-Pit project control center as slightly positive and made several suggestions on how to further improve implementation in future iterations (see Table 6 and Table 7). This result was expected for the first Soft-Pit iteration, because the focus was more on getting an initial version running within each specific organizational environment and reducing the number of control goals monitored by the control center.

**Table 8. Summary of Cronbach's alpha for usefulness and ease of use.**

| | *Usefulness* | *Ease of Use* | *N* |
|---|---|---|---|
| *Pooled evaluation (all views combined)* | .59 (.26) | .89 (.71) | 30 |
| *Effort Consumption* | .76 (.40) | .84 (.65) | 7 |
| *Schedule Completion* | .78 (.49) | .81 (.92) | 7 |
| *Code Quality* | -.16 (.01) | .83 (.62) | 7 |
| *General Application* | .81 (.51) | .95 (.88) | 9 |
| *Administration* | -1.4 (-.26) | .55 (.31) | 8 |

As seen in Figure 2, people also generally perceived the ease of use of the control views as positive. However, they had a negative perception of the ease of use of administering the control center. The latter is not astonishing when considering that the used tool is a scientific prototype. The respondents perceived the ease of use for the code quality view as particularly positive. The code quality view consists of a table listing indicator values over time and a simple color code that visualizes improvement over a previous version. In addition, a bar chart presents the indicator values over time. This diagram was found to be the easiest one to interpret.

The reliability of the scales for usefulness and ease of use is, in general, above the .70 threshold; thus, we can consider reliability as being sufficient. However, there are some exceptions. One is the administration view; both usefulness and ease of use have very low reliability. However, since this view is in a prototype state, and the concept for usefulness is quite different for administration than for using the control center, we did not expect consistent results. However, the low reliability of usefulness for the code quality view needs further examination.

For code quality, the reliability of the usefulness scale was exceptionally low. As we used the same questions as for the other views, it seems likely that the perception of usefulness is quite different for code

quality than for other control views. Table 9 presents the correlations of the scale items for usefulness for code quality; fields with negative correlation are marked in grey. In particular, B4 (view is holistic) is negatively correlated with B2 (view allows complete detection of problems), and B5 (view is a useful tool) with B1 (view allows early detection of problems) as well as with B2. One potential explanation is that the view does not provide sufficient information for the complete detection of product quality problems, and that the view is not holistic. This hypothesis is also confirmed by the box plot analysis of all questions addressing usefulness within the code quality view, as shown in Figure 3.

**Table 9. Correlations for usefulness of code quality.**

| | *B1* | *B2* | *B3* | *B4* | *B5* |
|---|---|---|---|---|---|
| *B1* | 1.000 | .271 | .091 | .050 | -.372 |
| *B2* | .271 | 1.000 | .198 | -.841 | -.101 |
| *B3* | .091 | .198 | 1.000 | .228 | .679 |
| *B4* | .050 | -.841 | .228 | 1.000 | .155 |
| *B5* | -.372 | -.101 | .679 | .155 | 1.000 |

The answers for B2 and B4 are, on average, much lower than the rest of the questions. That is, the respondents perceived the code quality view as useful, allowing early detection of problems and deviation causes, although it did not allow them to get a complete and holistic view; that is, they would need more information for a more complete detection of problems.

## 4. Lessons Learned

The following section discusses lessons learned with respect to the evaluation framework (LL1 to LL3) and the case study results (LL4 to LL8).

(LL1) Pretest of evaluation instrument: For the case studies conducted, the evaluation instrument was not tested before its application. On the one hand, this was justified for pragmatic reasons, as the questionnaire was based on a well proven model (TAM); on the other hand, this posed a high risk of having an unreliable instrument. However, time restrictions and the lack of appropriate subjects prohibited further pretesting (apart from plausibility checks). As the evaluation showed, the reliability of the scale items (questions) used for ease of use was quite high (see Table 8), whereas the reliability for usefulness was not that good (caused by very low reliability of the code quality and administration view). For the administration view, the questions seem to have some general problems for ease of use as well as for usefulness.

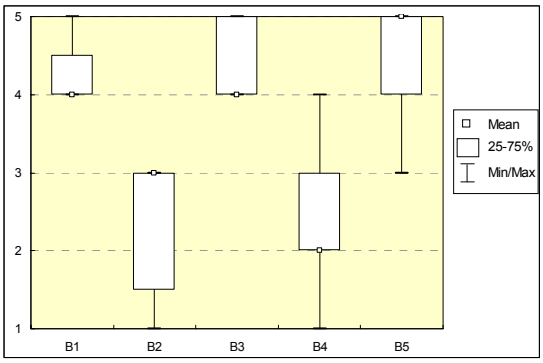


**Figure 3. Box plots for usefulness of code quality view.**

(LL2) Coach concept: The coach concept was positively accepted by all participants and has thus been proven. The main reason for establishing such a concept was the opportunity to give every evaluating company exactly one contact person (the coach). The coach was responsible for the correct execution of the evaluation by a company. From the company's point of view, this was seen as a big advantage. They had only one and always the same person to contact if problems regarding setup and usage of the control center, input data, or interpreting of control views occurred. Either the coach was able to solve those problems immediately or he requested a solution from Soft-Pit experts and communicated it to the one who needed help. To get profound knowledge about the control center, the mechanisms, and the responsible persons for every part of the system, the coaches were trained before the evaluation period in a coach tutorial. From the evaluator's point of view, the advantages were the uniform execution of the case studies supervised by the coaches. They checked that the measurement plan was fulfilled with respect to what, how, and when a metric was measured. They also checked regularly that the collected data was valid and the control views as well as the interpretation of the views were correct. Furthermore, the coaches collected qualitative feedback during and after the evaluation period. Another advantage for the evaluator was the distribution of responsibilities to the coaches. Thus, the effort to execute the evaluation was minimized for him.

(LL3) Questionnaire participants: Most of the conceptual setup of the control center (e.g., which initial goals to achieve, which data to collect for those goals, how to visualize the collected data) was not done by the project participants, but by Soft-Pit experts. How-

ever, those experts did not directly participate in the case study projects and did not fill out the questionnaire. The same holds for coaches assigned to the companies. However, both groups significantly contributed to setting up the control center conceptually (and technically) and should participate in the questionnaire in future iterations in order to provide a more complete picture.

(LL4) Web-based control center: The main part of the control center was implemented as a web application. This includes graphical user interface, visualization engine, visualization configuration, user management, administration, and data collection for project plans, schedules, and effort data. Data collection for the code quality view was implemented as a command line application. The benefits of the decision to make a web application were the relatively easy installation on only one computer (the server) and the absence of installing something else on the users' computers (the clients) as long as a web browser was installed. Another benefit is the opportunity to use the control center from anywhere in the world without any obstacles (which is especially suited for distributed software development). For every company, an administrative account and user accounts were preconfigured, so that the corresponding functionality was easy to reach. This was positively accepted and used by the participants.

(LL5) Scalability: Some problems mentioned when using the control center had to do with the scalability of visualizations. Some charts were hard to interpret and had layout problems if too much data was included (e.g., hundreds of activities instead of dozens).

(LL6) Holistic approach: In the first Soft-Pit iteration, the control center was focused on a very limited number of control goals. The qualitative user feedback revealed that more aspects, control views, and interactions were requested in order to get a true holistic control center that addresses all relevant aspects of an engineering-style software development approach relevant for a specific project.

(LL7) Additional effort: The installation and maintenance of the control center introduced some additional effort that we tried to compensate through extended support by the coaches. However, project participants had to be trained to use the control center and had to spend additional time to use it during project execution. For some companies, the concept of a generic control center was basically new, for others, sophisticated tools for project control were already in place. Because the control center focused on a very limited set of control goals, the additional value was limited depending on the previously used mechanisms for project control.

(LL8) Automated data collection: The automatic import of data was crucial for user acceptance. In order to use the control center regularly as a standard instrument for keeping the project on track, the barrier of being able to use the control center (having reliable and up-to-date data) has to be as low as possible. Due to some technical problems (unavailable interface definitions of commercial software), it was not possible for all case study projects to automatically retrieve data from already existing data repositories. Thus, measurement data had to be exported from those repositories and manually imported into the control center, which caused some effort before reliable and up-to-date data was presented.

## 5. Conclusion and Future Work

This article presented an evaluation framework for Software Project Control Centers and its practical evaluation in industrial case studies as part of the Soft-Pit project. The case studies and evaluation instruments were described and the results of the studies were discussed. Lessons learned were derived with respect to the evaluation framework and the evaluated Soft-Pit control center. In general, the evaluation framework could be applied successfully and should be transferable to other control center evaluations. It will be adapted for the next Soft-Pit iterations, so that the progression of the conceptual and technical control center implementation can be demonstrated. The control center used in the first iteration indicated quite a good start, but also (as expected) a lot of issues to be worked on until the project goal of developing a holistic control center approach is achieved.

The evaluation instrument will be re-applied in future iterations of the Soft-Pit control center. Therefore, we plan to adapt existing and introduce additional scale items for usefulness in order to increase the reliability (with respect to Cronbach's alpha). The scale items for the control center administration view will be completely reworked to get an acceptable reliability, as the evaluation results indicate that user perception of usefulness and ease of use are quite different from the other views. Moreover, the participants of the questionnaire will most probably be extended to at least include the coaches who were mainly responsible for customizing the control center for the specific organization and projects. If possible, we will conduct a pretest in order to check the general quality of the evaluation instrument.

With respect to the conceptual development of the Soft-Pit project control center approach, future work will focus on broadening the scope of project control in order to address a wider spectrum of goals and de-

velop an overall holistic approach. Moreover, interactions between different control goals will become an issue in order to detect and resolve goal conflicts. Furthermore, we plan to address abstraction and drill-down mechanism as well as messaging mechanisms in order to systematically support the control of distributed projects. Finally, intuitive, easy-to-use visualizations are a key factor for ease of use and, therewith, for the acceptance of a project control center in a productive environment. In this area, the focus will be on visualization metaphors that facilitate the interpretation of measurement data.

## 6. Acknowledgements

We would like to thank Sonnhild Namingha from Fraunhofer IESE for reviewing a first version of the article. This work was partly funded by the German Federal Ministry of Education and Research (BMBF) in the context of the "Soft-Pit" project, a "Software Engineering 2006" research initiative.

## 7. References

[1] D.A. Adams, R.R. Nelson, and P.A. Todd, "Perceived usefulness, ease of use, and usage of information technology: A replication", *MIS Quarterly*, 16, 1992, pp. 227-247.

[2] V.R. Basili, G. Caldiera, D. Rombach, "The Experience Factory", *Encyclopaedia of Software Engineering*, 1, 1994, pp. 469-476.

[3] Borland, Managing J2EE™ Performance Using Borland® Optimizeit™ ServerTrace, WhitePaper, http://-www.borland.com/resources/en/pdf/white_papers/managing_J2EE_performance.pdf, last checked Jan 11, 2006.

[4] E.G. Carmines and R.A. Zeller. Reliability and Validity Assessment. Sage Publication, 1979.

[5] CAST, *CAST Application Intelligence Platform*, http://www.castsoftware.com, last checked Jan 11, 2006.

[6] B. Daubner, A. Henrich, and B. Westfechtel, "A Lightweight Tool Support for Integrated Software Measurement", *Proceedings of the International Workshop on Software Metrics and DASMA Software Metrik Kongress IWSM/MetriKon 2006*, Potsdam (Germany), 2006, pp. 67-80.

[7] F.D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology", *MIS Quarterly*, 13(3), 1990, pp. 319-340.

[8] W.W. Gibbs, "Software's Chronic Crisis", *Scientific American*, 1994, pp. 86-95.

[9] J. Heidrich, J. Münch, A. Wickenkamp, "Usage Scenarios for Measurement-based Project Control", *Proceedings of the 3rd Software Measurement European Forum (Smef 2006)*, Rome, Italy, May 10-12, 2006, pp. 47-60.

[10] R. Hendrick, D. Kistler, and J. Valett, *Software Management Environment (SME)—Concepts and Architecture (Revision 1)*, NASA Goddard Space Flight Center Code 551, Software Engineering Laboratory Series, Report SEL-89-103, Greenbelt, MD, USA, 1992.

[11] B. Krishnamurthy and N.S. Barghouti, "Provence: A Process Visualization and Enactment Environment", *Proceedings of the 4th European Software Engineering Conference*, Lecture Notes in Computer Science 717; Springer: Heidelberg, Germany, 1993, pp. 451-465.

[12] F. McGarry, R. Pajerski, G. Page, S. Waligora, V.R. Basili, and M.V. Zelkowitz, *An Overview of the Software Engineering Laboratory*, Software Engineering Laboratory Series Report SEL-94-005, Greenbelt, MD, USA, 1994.

[13] J. Münch, J. Heidrich, "Software Project Control Centers: Concepts and Approaches", *Journal of Systems and Software*, 70 (1), 2004, pp. 3-19.

[14] U. Richter and F. Simon, "Successful implementation of a Code-Control-Cockpit to improve internal code quality", *Proceedings of ICSTest2004*, Düsseldorf, 2004.

[15] H.D. Rombach, "Practical benefits of goal-oriented measurement", *Software Reliability and Metrics*, 1991, pp. 217-235.

[16] R.W. Selby, A.A. Porter, D.C. Schmidt, and J. Berney, "Metric-Driven Analysis and Feedback Systems for Enabling Empirically Guided Software Development", *Proceedings of the 13th International Conference on Software Engineering*, 1991, pp. 288-298.

[17] D.B. Simmons, N.C. Ellis, H. Fujihara, and W. Kuo, *Software Measurement – A Visualization Toolkit for Project Control and Process Improvement*, Prentice Hall Inc: New Jersey, USA, 1998.

[18] F. Simon "Metriken, weniger ist mehr", Computer Zeitung, May 22, 2006.

[19] SQS AG, *SQS-Test-Professional, Component Process Performance Management to Monitor Test Quality and Error Rates*, http://www.sqs.de/portfolio/tools/tools_ppm.htm, last checked Jan 11, 2006.

[20] R. Tesoriero and M.V. Zelkowitz, "The Web Measurement Environment (WebME): A Tool for Combining and Modeling Distributed Data", *Proceedings of the 22nd Annual Software Engineering Workshop (SEW)*, 1997.

[21] K. Torii, K. Matsumoto, K. Nakakoji, Y. Takada, S. Takada, and K. Shima, "Ginger2: An Environment for Computer-Aided Empirical Software Engineering", *IEEE Transactions on Software Engineering*, 25(4), 1999, pp. 474-492.

[22] V. Venkatesh, M.G. Morris, G.B. Davis, and F.D. Davis, "User acceptance of information technology: Toward a unified view", *MIS Quarterly*, 27(3), 2003, pp. 425-478.