National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de Technologie de l'information

ERB-1061

NRC·CNRC

Validating the ISO/IEC15504 Measures of Software Development Process Capability

Khaled El Emam and Andreas Birk

August1999

Canada

NRC 43605

National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de Technologie de l'information

Validating the ISO/IEC 15504 Measures of Software Development Process Capability

Khaled El Emam and Andreas Birk

August 1999

Validating the ISO/IEC 15504 Measures of Software Development Process Capability

Khaled El Emam
National Research Council, Canada
Institute for Information Technology
Building M-50, Montreal Road
Ottawa, Ontario
Canada K1A OR6
Khaled.El-Emam@iit.nrc.ca

Andreas Birk
Fraunhofer Institute for Experimental Software Engineering
Sauerwiesen 6
D-67661 Kaiserslautern
Germany
Andreas.Birk@iese.fhg.de

Abstract

ISO/IEC 15504 is an emerging international standard on software process assessment. It defines a number of software engineering processes, and a scale for measuring their capability. A basic premise of the measurement scale is that higher process capability is associated with better project performance (i.e., predictive validity). This paper describes an empirical study that evaluates the predictive validity of the capability measures of the ISO/IEC 15504 software development processes (i.e., develop software design, implement software design, and integrate and test). Assessments using ISO/IEC 15504 were conducted on projects world wide over a period of two years. Performance measures on each project were also collected using questionnaires, such as the ability to meet budget commitments and staff productivity. The results provide evidence of predictive validity for the development process capability measures used in ISO/IEC 15504 for large organizations (defined as having more than 50 IT staff). Furthermore, it was found that the "Develop Software Design" process was associated with most project performance measures. For small organizations evidence of predictive validity was rather weak. This can be interpreted in a number of different ways: that the measures of capability are not suitable for small organizations, or that software development process capability has less affect on project performance for small organizations.

1 Introduction

Improving software processes is by now recognized as an important endeavor for software organizations. A commonly used paradigm for improving software engineering practices is the benchmarking paradigm (Card, 1991). This involves identifying an 'excellent' organization or project and documenting its practices. It is then assumed that if a less-proficient organization or project adopts the practices of the excellent one, it will also become excellent. Such best practices are commonly codified in an assessment model, like the SW-CMM[1] (Software Engineering Institute, 1995) or the emerging ISO/IEC 15504 international standard (El Emam et al., 1998). These assessment models also order the practices in a recommended sequence of implementation, hence providing a predefined improvement path.[2]

[1] The Capability Maturity Model for Software.

[2] The logic of this sequencing is that this is the natural evolutionary order in which, historically, software organizations improve (Humphrey, 1988), and that practices early in the sequence are prerequisite foundations to ensure the stability and optimality of practices implemented later in the sequence (Software Engineering Institute, 1995).

Improvement following the benchmarking paradigm almost always involves a software process assessment (SPA).[3] A SPA provides a quantitative score reflecting the extent of an organization's or project's implementation of the best practices defined in the assessment model. The more of these best practices that are adopted, the higher this score is expected to be. The obtained score provides a baseline of current implementation of best practices, serves as a basis for making process improvement investment decisions, and also provides a means of tracking improvement efforts.

The emerging ISO/IEC 15504 international standard is an attempt to harmonize the existing assessment models that are in common use. It defines a scheme for measuring the capability of software processes. A basic premise of 15504 is that the quantitative score from the assessment is associated with the performance of the organization or project. In fact, this is a premise of all assessment models. Therefore, improving the software engineering practices according to the assessment model is expected to subsequently improve the performance. This is termed the *predictive validity* of the process capability score. Empirically validating the verisimilitude of such a premise is of practical importance since substantial process improvement investments are made by organizations guided by the assessment results.

While there have been some correlational studies that substantiate the above premise, none evaluated the predictive validity of the process capability measures defined in ISO/IEC 15504. The implication then is that it is not possible to substantiate claims that improvement by adopting the practices stipulated in 15504 really results in performance improvements.

In this paper we empirically investigate the relationship between the capability of the software development processes (namely design, coding, and integration and testing) as defined in the emerging ISO/IEC 15504 international standard[4] and the performance of software projects. The study was conducted in the context of the SPICE Trials, which is an international effort to empirically evaluate the emerging international standard world wide. To our knowledge, this is the first study to evaluate the predictive validity of software development process capability using the ISO/IEC 15504 measure of process capability.

Briefly, our results can be summarized in the form of Table 1. This indicates that for small organizations (less than or equal to 50 IT staff), we only found evidence that the "Develop Software Design" process is related with a project's ability to meet schedule commitments. For large organizations, the same process is related to five different project performance measures, including customer satisfaction and the ability to satisfy specified requirements. The "Implement Software Design" process is associated with an improved

[3] Here we use the term "SPA" in the general sense, not in the sense of the SEI specific assessment method (which was also called a SPA).

[4] In this paper we only refer to the PDTR version of the ISO/IEC 15504 document set since this was the one used during our empirical study. The PDTR version reflects one of the stages that a document has to go through on the path to international standardization. The PDTR version is described in detail in (El Emam et al., 1998).

ability to meet budget commitments, and the "Integrate and Test Software" process is associated with improved productivity.

Performance Measure	**Process(es)**
Small Organizations	
Ability to meet budget commitments	
Ability to meet schedule commitments	Develop Software Design
Ability to achieve customer satisfaction	
Ability to satisfy specified requirements	
Staff productivity	
Staff morale / job satisfaction	
Large Organizations	
Ability to meet budget commitments	Develop Software Design Implement Software Design
Ability to meet schedule commitments	Develop Software Design
Ability to achieve customer satisfaction	Develop Software Design
Ability to satisfy specified requirements	Develop Software Design
Staff productivity	Integrate and Test Software
Staff morale / job satisfaction	Develop Software Design

Table 1: Summary of the findings from our predictive validity study. In the first column are the performance measures that were collected for each project. In the second column are the development processes whose capability was evaluated. The results are presented separately for small (equal to or less than 50 IT staff) and large organizations (more than 50 IT staff). For each performance measure we show the software development processes that were found to be related to it. For example, for large organizations, we found that the "Develop Software Design" and "Implement Software Design" processes were associated with the "Ability to meet budget commitments". Also, for instance, we did not find any processes that were associated with "Staff productivity" in small organizations. A process was considered to be associated with a performance measure if it had a correlation coefficient that was greater than or equal to 0.3, and that was statistically significant at a one-tailed (Bonferonni adjusted) alpha level of 0.1.

In the next section we provide the background to our study. This is followed in Section 3 with an overview of the ISO/IEC 15504 architecture and rating scheme that was used during our study. Section 4 details our research method, and Section 5 contains the results. We conclude the paper in Section 6 with a discussion of our results and directions for future research.

2 Background

A recent survey of assessment sponsors found that baselining process capability and tracking process improvement progress are two important reasons for conducting a SPA (El Emam and Goldenson, 2000). Both of these reasons rely on the quantitative score obtained from an assessment, indicating that sponsors perceive assessments as a measurement procedure.

As with any measurement procedure, its validity must be demonstrated before one has confidence in its use. The validity of measurement is defined as the extent to which a measurement procedure is measuring what it is purporting to measure (Kerlinger, 1986). During the process of validating a measurement procedure one attempts to collect evidence to support the types of inferences that are to be drawn from measurement scores.

A basic premise of SPAs is that the resultant quantitative scores are predictors of the performance of the project and/or organization that is assessed. Testing this premise can be considered as an evaluation of the *predictive validity* of the assessment measurement procedure (El Emam and Goldenson, 1995).

In this section we review existing theoretical and empirical work on the measurement of development process capability and the predictive validity of such measures.

2.1 Theoretical Model Specification

A predictive validity study typically tests the hypothesized model shown in Figure 1. This shows that there is a relationship between process capability and performance, and that this relationship is dependent upon some context factors (i.e., the relationship functional form or direction may be different for different contexts, or may exist only for some contexts).

The hypothesized model can be tested for different units of analysis (Goldenson et al., 1999). The three units of analysis are the life cycle process (e.g., the design process), the project (which could be a composite of the capability of multiple life cycle processes of a single project, such as design and coding), or the organization (which could be a composite of the capability of the same or multiple processes across different projects). All of the three variables in the model can be measured at any one of these units of analysis.[5]

[5] Although, if process capability is measured on a different unit from the performance measure, then the results of a predictive validity study may be more difficult to interpret.

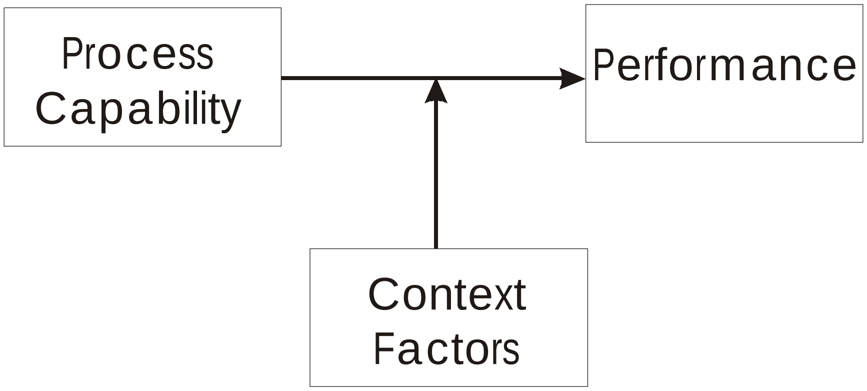

Figure 1: Theoretical model being tested in a predictive validity study of process capability.

The literature refers to measures at different units of analysis using different terminology. To remain consistent, we will use the term “process capability”, and preface it with the unit of analysis where applicable. For example, one can make a distinction between measuring process capability, as in ISO/IEC 15504, and measuring organizational maturity, as in the SW-CMM (Paulk and Konrad, 1994). Organizational maturity can be considered as a measure of organizational process capability.

2.2 Theoretical Basis for Validating Software Development Process Capability

Three existing models explicitly hypothesize benefits as software development process capability is improved. These are reviewed below.

The SW-CMM defines 18 KPAs that are believed to represent good software engineering practices (Software Engineering Institute, 1995). The main design, construction, integration, and testing processes are embodied in the Software Product Engineering KPA (Software Engineering Institute, 1998a). This is defined at Level 3.

As organizations increase their organizational process capability by implementing progressively more of these processes, it is hypothesized that three types of benefits will accrue (Paulk et al., 1993):

- the differences between targeted results and actual results will decrease across projects,
- the variability of actual results around targeted results decreases, and
- costs decrease, development time shortens, and productivity and quality increase.

However, these benefits are not posited only for the Software Product Engineering KPA, but rather as a consequence of implementing combinations of practices.

The emerging ISO/IEC 15504 international standard, on the other hand, defines a set of processes, and a scale that can be used to evaluate the capability of each process separately (El Emam et al., 1998) (details of the ISO/IEC 15504 architecture are provided in Section 3). The initial requirements for ISO/IEC 15504 state that an organization’s assessment results should reflect its ability to achieve productivity and/or development cycle time goals (El Emam et al., 1998). It is not clear however, whether this is hypothesized for each individual process, or for combinations of processes.

The Software Engineering Institute has published a so-called *Technology Reference Guide* (Software Engineering Institute, 1997), which is a collection and classification of software technologies. Its purpose is to foster technology dissemination and transfer. Each technology is classified according to processes in which it can be applied (*application taxonomy*) and according to qualities of software systems that can be expected as a result from applying the technology (*quality measures taxonomy*). The classifications have passed a comprehensive review by a large number of noted software engineering experts. This accumulated expert opinion can be used as another source of claims on the impact of design, implementation, integration and testing processes on overall project performance.

The technologies listed for the process categories related to design, implementation, integration and testing can be mapped to quality measures such as correctness, reliability, maintainability, understandability, and cost of ownership. Through such a mapping one can posit relationships between the practices and the quality attributes.

Therefore, the existing literature does strongly suggest that there is a relationship between software development process capability and performance. However, the models differ in the expected benefits that they contend will accrue from their implementation, and also the former two in their process capability measurement schemes.

2.3 Evidence of the Predictive Validity of Development Process Capability Measures

To our knowledge, no empirical evidence exists supporting the predictive validity of the software development process capability measures as defined in ISO/IEC 15504. The Technology Reference Guide is based largely on expert judgement. Nevertheless, there have been studies of predictive validity based on the SW-CMM and other models. In the following we review these studies.

Two classes of empirical studies have been conducted and reported thus far: case studies and correlational studies (Goldenson et al., 1999). Case studies describe the experiences of a single organization (or a small number of selected organizations) and the benefits it gained from increasing its process capability. Case studies are most useful for showing that there are organizations that have benefited from increased process capability. Examples of these are reported in (Humphrey et al., 1991; Herbsleb et al., 1994; Dion, 1992; Dion, 1993; Wohlwend and Rosenbaum, 1993; Benno and Frailey, 1995; Lipke and Butler, 1992; Butler 1995; Lebsanft, 1996; Krasner, 1999). However, in this context, case studies have a methodological disadvantage that makes it difficult to generalize the results from a single case study or even a small number of case studies. Case studies tend to suffer from a selection bias because:

- Organizations that have not shown any process improvement or have even regressed will be highly unlikely to publicize their results, so case studies tend to show mainly success stories (e.g., all the references to case studies above are success stories), and
- The majority of organizations do not collect objective process and product data (e.g., on defect levels, or even keep accurate effort records). Only organizations that have made improvements and reached a reasonable level of maturity will have the actual objective data to demonstrate improvements (in productivity, quality, or return on investment). Therefore failures and non-movers are less likely to be considered as viable case studies due to the lack of data.[6]

With correlational studies, one collects data from a larger number of organizations or projects and investigates relationships between process capability and performance statistically. Correlational studies are useful for showing whether a general association exists between increased capability and performance, and under what conditions.

There have been a few correlational studies in the past that evaluated the predictive validity of various process capability measures. For example, Goldenson and Herbsleb (Goldenson and Herbsleb, 1995) evaluated the relationship between SW-CMM capability scores and organizational performance measures. They surveyed individuals whose organizations have been assessed against the SW-CMM. The authors evaluated the benefits of higher process capability using subjective measures of performance. Organizations with higher capability tend to perform better on the following dimensions (respondents chose either the "excellent" or "good" response categories when asked to characterize their organization's performance on these dimensions): ability to meet schedule, product quality, staff productivity, customer satisfaction, and staff morale. The relationship with the ability to meet budget commitments was not found to be statistically significant.

A more recent study considered the relationship between the implementation of the SW-CMM KPA's and delivered defects (after correcting for size and personnel capability) (Krishnan and Kellner, 1998). They found evidence that increasing process capability is negatively associated with delivered defects.

Another correlational study investigated the benefits of moving up the maturity levels of the SW-CMM (Flowe and Thordahl, 1994; Lawlis et al., 1996)[7]. They obtained data from historic U.S. Air Force contracts. Two measures were considered: (a) cost performance index which evaluates deviations in actual vs. planned project cost, and (b) schedule performance index which evaluates the extent to which schedule has been over/under-run. Generally, the results show that higher maturity projects approach on-target cost, and on-target schedule.

[6] Exceptions would be where contractual requirements mandate the collection and reporting of performance data, such as schedule and cost performance. This, for instance, occurs with DoD contracts.

[7] This data set was reanalyzed by El Emam and Goldenson (El Emam and Goldenson, 2000) to address some methodological questions. Although, the conclusions of the reanalysis are the same as the original authors'.

McGarry et al. (McGarry et al., 1998) investigated the relationship between assessment scores using an adaptation of the SW-CMM process capability measures and project performance for fifteen projects within a single organization. They did not find strong evidence of predictive validity, although all relationships were in the expected direction.

Clark (Clark, 1997) investigated the relationship between satisfaction of SW-CMM goals and software project effort, after correcting for other factors such as size and personnel experience. His results indicate that the more KPAs are implemented, the less effort is consumed on projects.

Gopal et al. (Gopal et al., 1999) performed a study with two Indian software firms, collecting data on 34 application software projects. They investigated the impact of process capability as measured by the SW-CMM KPAs. Specifically, they identified two dimensions of capability: Technical Processes (consisting of Requirements Management, Software Product Engineering, Software Configuration Management, and Software Product Planning) and Quality Processes (consisting of Training Program, Peer Reviews, and Defect Prevention). The Technical Processes dimension embodies the software development processes that are of primary interest in our study. They found that the Quality Processes were related to a reduction in rework and increases in overall effort, and the Technical Processes were associated with a reduction in effort and increases in elapsed time.

Harter et al. (Harter et al., 1999) report on a comprehensive study to evaluate the impact of process maturity as measured by the SW-CMM levels. They found that higher maturity is associated with higher product quality. No direct effect of maturity on cycle time was found, but the net effect of process maturity on cycle time was negative due to the improvements in product quality (i.e., higher maturity leads to better quality which in turn leads to reduced cycle time due to less rework). Also, the direct effect of maturity on development effort was found to be positive. But the net effect of higher maturity on effort was negative due to improvements in quality (i.e., higher maturity leads to higher quality which in turn leads to reduced effort due to less rework). In this particular study product quality was measured as the reciprocal of defect density for defects found during system and acceptance testing.

Jones presents the results of an analysis on the benefits of moving up the 7-level maturity scale of Software Productivity Research (SPR) Inc.'s proprietary model (Jones, 1999; Jones, 1996). This data were collected from SPR's clients. His results indicate that as organizations move from Level 0 to Level 6 on the model they witness (compound totals): 350% increase in productivity, 90% reduction in defects, 70% reduction in schedules.

Deephouse et al. evaluated the relationship between individual processes and project performance (Deephouse et al., 1995). As would be expected, they found that evidence of predictive validity depends on the particular performance measure that is considered. One study by El Emam and Madhavji (El Emam and Madhavji, 1995) evaluated the relationship between four dimensions of organizational process capability and the success of the requirements engineering process. Evidence of predictive validity was found for only one dimension. However, neither of these studies used the ISO/IEC 15504 measure of process capability.

As can be seen from the above review that despite there being studies with other capability measurement schemes, no evidence exists that demonstrates the relationship between the capability of software development processes as defined in ISO/IEC 15504 and the performance of software projects. This means that we cannot substantiate claims that improving the capability of the ISO/IEC 15504 software development processes will lead to any improvement in project performance, and we cannot be specific about which performance measures will be affected. Hence, the rationale for the current study.

2.4 Moderating Effects

A recent review of the empirical literature on software process assessments noted that existing evidence suggests that the extent to which a project's or organization's performance improves due to the implementation of good software engineering practices (i.e., increasing process capability) is dependent on the context (El Emam and Briand, 1999). This highlights the need to consider the project and/or organizational context in predictive validity studies. However, it has also been noted that the overall evidence remains equivocal as to which context factors should be considered in predictive validity studies (El Emam and Briand, 1999).

In our current study we consider the size of the organization as a context factor. This is not claimed to be the only context factor that ought to be considered, but is only one of the important ones that has been mentioned repeatedly in the literature.

Previous studies provide inconsistent results about the effect of organizational size. For example, there have been some concerns that the implementation of some of the practices in the CMM, such as a separate Quality Assurance function and formal documentation of policies and procedures, would be too costly for small organizations (Brodman and Johnson, 1994). Therefore, the implementation of certain processes or process management practices may not be as cost-effective for small organizations as for large ones. However, a moderated analysis of the relationship between organizational capability and requirements engineering process success (using the data set originally used in (El Emam and Madhavji, 1995)) found that organizational size does not affect predictive validity (El Emam and Briand, 1999). This result is consistent with that found in (Goldenson and Herbsleb, 1995) for organization size and (Deephouse et al., 1995) for project size, but is at odds with the findings from (Brodman and Johnson, 1994).

To further confuse the issue, an earlier investigation (Lee and Kim, 1992) studied the relationship between the extent to which software development processes are standardized and MIS success.[8] It was found that standardization of life cycle processes was associated with MIS success in smaller organizations but not in large ones. This is in contrast to some of the findings cited above. In summary, it

[8] Process standardization is a recurring theme in process capability measures.

is not clear if and how organization size moderates the benefits of process and the implementation of process management practices.

We therefore explicitly consider organizational size as a factor in our study to identify if the predictive validity results are different for different sized organizations.

3 Overview of the ISO/IEC PDTR 15504 Rating Scheme

3.1 The Architecture

The architecture of ISO/IEC 15504 is two-dimensional as shown in Figure 2. One dimension consists of the processes that are actually assessed (the Process dimension) that are grouped into five categories. The second dimension consists of the capability scale that is used to evaluate the process capability (the Capability dimension). The same capability scale is used across all processes. Software development processes are defined in the Engineering process category in the Process dimension.

During an assessment it is not necessary to assess all the process in the process dimension. Indeed, an organization can scope an assessment to cover only the subset of processes that are relevant for its business objectives. Therefore, not all organizations that conduct an assessment based on ISO/IEC 15504 will necessarily cover all of the development processes within their scope.

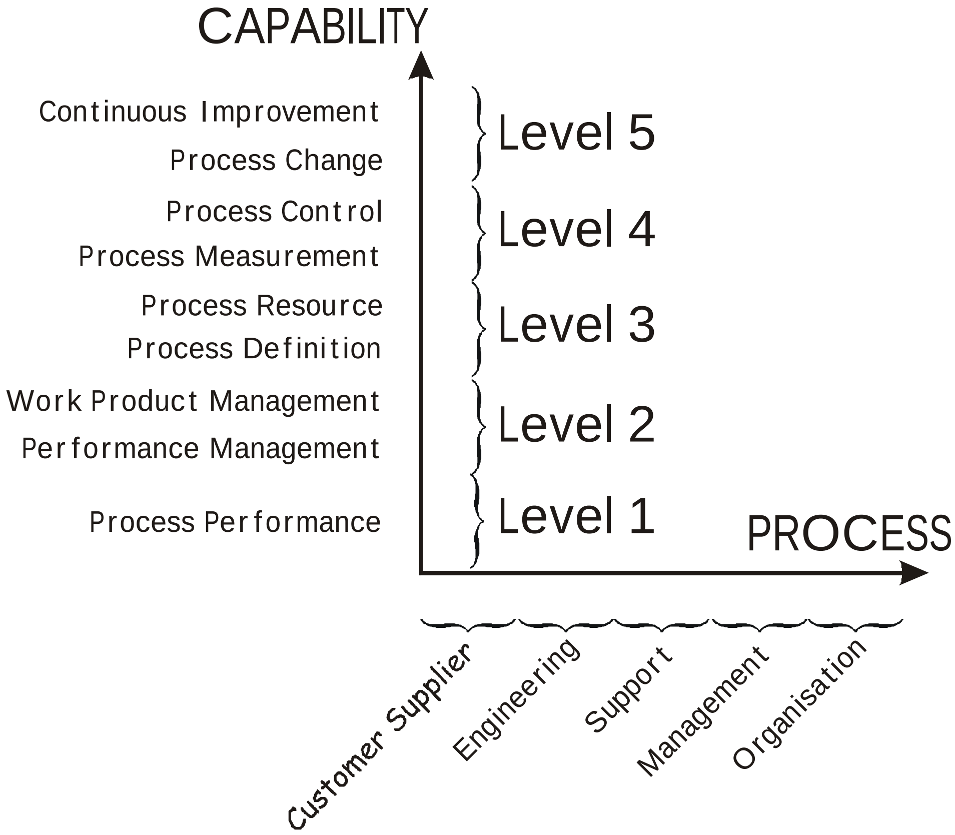

Figure 2: An overview of the ISO/IEC 15504 two dimensional architecture.

In ISO/IEC 15504, there are 5 levels of capability that can be rated, from Level 1 to Level 5. A Level 0 is also defined, but this is not rated directly. These 6 levels are shown in Table 2. In Level 1, one attribute is directly rated. There are 2 attributes in each of the remaining 4 levels. The attributes are also shown in Table 2, and explained in more detail in (El Emam et al., 1998).

The rating scheme consists of a 4-point *achievement* scale for each attribute. The four points are designated as F, L, P, N for *Fully Achieved*, *Largely Achieved*, *Partially Achieved*, and *Not Achieved*. A summary of the definition for each of these response categories is given in Table 3.

It is not required that all the attributes in all five levels be rated during an assessment. For example, it is permissible that an assessment only rate attributes up to say level 3, and not rate at levels 4 and 5.

The unit of rating in an ISO/IEC PDTR 15504 process assessment is the process instance. A process instance is defined as a singular instantiation of a process that is uniquely identifiable and about which information can be gathered in a repeatable manner (El Emam et al., 1998).

ID	Title
Level 0	**Incomplete Process** There is general failure to attain the purpose of the process. There are no easily identifiable work products or outputs of the process.
Level 1	**Performed Process** The purpose of the process is generally achieved. The achievement may not be rigorously planned and tracked. Individuals within the organization recognize that an action should be performed, and there is general agreement that this action is performed as and when required. There are identifiable work products for the process, and these testify to the achievement of the purpose.
1.1	**Process performance attribute**
Level 2	**Managed Process** The process delivers work products of acceptable quality within defined timescales. Performance according to specified procedures is planned and tracked. Work products conform to specified standards and requirements. The primary distinction from the Performed Level is that the performance of the process is planned and managed and progressing towards a defined process.
2.1	**Performance management attribute**
2.2	**Work product management attribute**
Level 3	**Established Process** The process is performed and managed using a defined process based upon good software engineering principles. Individual implementations of the process use approved, tailored versions of standard, documented processes. The resources necessary to establish the process definition are also in place. The primary distinction from the Managed Level is that the process of the Established Level is planned and managed using a standard process.
3.1	**Process definition attribute**
3.2	**Process resource attribute**
Level 4	**Predictable Process** The defined process is performed consistently in practice within defined control limits, to achieve its goals. Detailed measures of performance are collected and analyzed. This leads to a quantitative understanding of process capability and an improved ability to predict performance. Performance is objectively managed. The quality of work products is quantitatively known. The primary distinction from the Established Level is that the defined process is quantitatively understood and controlled.
4.1	**Process measurement attribute**
4.2	**Process control attribute**
Level 5	**Optimizing Process** Performance of the process is optimized to meet current and future business needs, and the process achieves repeatability in meeting its defined business goals. Quantitative process effectiveness and efficiency goals (targets) for performance are established, based on the business goals of the organization. Continuous process monitoring against these goals is enabled by obtaining quantitative feedback and improvement is achieved by analysis of the results. Optimizing a process involves piloting innovative ideas and technologies and changing non-effective processes to meet defined goals or objectives. The primary distinction from the Predictable Level is that the defined process and the standard process undergo continuous refinement and improvement, based on a quantitative understanding of the impact of changes to these processes.
5.1	**Process change attribute**
5.2	**Continuous improvement attribute**

Table 2: Overview of the capability levels and attributes.

The scope of an assessments is an Organizational Unit (OU) (El Emam et al., 1998). An OU deploys one or more processes that have a coherent process context and operates within a coherent set of business goals. The characteristics that determine the coherent scope of activity - the process context - include the application domain, the size, the criticality, the complexity, and the quality characteristics of its products or services. An OU is typically part of a larger organization, although in a small organization the OU may be the whole organization. An OU may be, for example, a specific project or set of (related) projects, a unit within an organization focused on a specific life cycle phase (or phases), or a part of an organization responsible for all aspects of a particular product or product set.

Rating & Designation	Description
Not Achieved - N	There is no evidence of achievement of the defined attribute.
Partially Achieved - P	There is some achievement of the defined attribute.
Largely Achieved - L	There is significant achievement of the defined attribute.
Fully Achieved - F	There is full achievement of the defined attribute.

Table 3: The four-point attribute rating scale.

3.2 Measuring Software Development Process Capability

In ISO/IEC 15504, the software development process is embodied in three processes: *Develop Software Design*, *Implement Software Design*, *Integrate and Test Software*.

One of the ISO/IEC 15504 documents contains an exemplar assessment model (known as Part 5). This provides further details of how to rate the development processes. Almost all of the assessments that were part of our study used Part 5 directly, and those that did not used models that are based on Part 5. Therefore a discussion of the guidance for rating the software development processes in Part 5 is relevant here.

For each process there are a number of *base practices* that can be used as indicators of performance (attribute 1.1 in Table 2). A base practice is a software engineering or project management activity that addresses the purpose of a particular process. Consistently performing the base practices associated with a process will help in consistently achieving its purpose. The base practices are described at an abstract level, identifying "what" should be done without specifying "how". The base practices characterize performance of a process. Implementing only the base practices of a process may be of minimal value and represents only the first step in building process capability, but the base practices represent the unique, functional activities of the process, even if that performance is not systematic. The base practices are summarized below for each of our three processes.

3.2.1 Develop Software Design

The purpose of the *Develop software design* process is to define a design for the software that accommodates the requirements and can be tested against them. As a result of successful implementation of the process:

- an architectural design will be developed that describes major software components which accommodate the software requirements;
- internal and external interfaces of each software component will be defined;
- a detailed design will be developed that describes software units that can be built and tested;
- traceability will be established between software requirements and software designs.

Base practices that should exist to indicate that the purpose of the Develop Software Design process has been achieved are:

Develop software architectural design. Transform the software requirements into a software architecture that describes the top-level structure and identifies its major components.

Design interfaces. Develop and document a design for the external and internal interfaces.

Develop detailed design. Transform the top level design into a detailed design for each software component. The software components are refined into lower levels containing software units. The result of this base practice is a documented software design which describes the position of each software unit in the software architecture.[9]

Establish Traceability. Establish traceability between the software requirements and the software designs.

3.2.2 Implement Software Design

The purpose of the *Implement software design* process is to produce executable software units and to verify that they properly reflect the software design. As a result of successful implementation of the process:

- verification criteria will be defined for all software units against software requirements;
- all software units defined by the design will be produced;
- verification of the software units against the design is accomplished.

Base practices that should exist to indicate that the purpose of the Implement Software Design process has been achieved are:

[9] The detailed design includes the specification of interfaces between the software units.

Develop software units. Develop and document each software unit.[10]

Develop unit verification procedures. Develop and document procedures for verifying that each software unit satisfies its design requirements.[11]

Verify the software units. Verify that each software unit satisfies its design requirements and document the results.

3.2.3 Integrate and Test Software

The purpose of the *Integrate and test software* process is to integrate the software units with each other producing software that will satisfy the software requirements. This process is accomplished step by step by individuals or teams. As a result of successful implementation of the process:

- an integration strategy will be developed for software units consistent with the release strategy;
- acceptance criteria for aggregates will be developed that verify compliance with the software requirements allocated to the units;
- software aggregates will be verified using the defined acceptance criteria;
- integrated software will be verified using the defined acceptance criteria;
- test results will be recorded;
- a regression strategy will be developed for retesting aggregates or the integrated software should a change in components be made.

Base practices that should exist to indicate that the purpose of the Integrate and Test Software process has been achieved are:

Determine regression test strategy. Determine the strategy for retesting aggregates should a change in a given software unit be made.

Build aggregates of software units. Identify aggregates of software units and a sequence or partial ordering for testing them.[12]

Develop tests for aggregates. Describe the tests to be run against each software aggregate, indicating software requirements being checked, input data and acceptance criteria.

Test software aggregates. Test each software aggregate against the acceptance criteria, and document the results.

[10] This base practice involves creating and documenting the final representations of each software unit.

[11] The normal verification procedure will be through unit testing, and the verification procedure will include unit test cases and unit test data.

[12] Typically, the software architecture and the release strategy will have some influence on the selection of aggregates.

Integrate software aggregates. Integrate the aggregated software components to form a complete system.

Develop tests for software. Describe the tests to be run against the integrated software, indicating software requirements being checked, input data, and acceptance criteria. The set of tests should demonstrate compliance with the software requirements and provide coverage of the internal structure of the software.[13]

Test integrated software. Test the integrated software against the acceptance criteria, and document the results.

3.2.4 Rating Level 2 and 3 Attributes

For higher capability levels, a number of *Management Practices* have to be evaluated to determine the attribute rating. For each of the attributes in levels 2 and 3, the management practices are summarized below. We do not consider levels above 3 because we do not include higher level ratings within our study.

3.2.4.1 Performance management attribute

This is defined as the extent to which the execution of the process is managed to produce work products within stated time and resource requirements. In order to achieve this capability, a process needs to have time and resources requirements stated and produce work products within the stated requirements. The related Management Practices are:

Management practices
Identify resource requirements to enable planning and tracking of the process.
Plan the performance of the process by identifying the activities of the process and the allocated resources according to the requirements.
Implement the defined activities to achieve the purpose of the process.
Manage the execution of the activities to produce the work products within stated time and resource requirements.

3.2.4.2 Work product management attribute

This is defined as the extent to which the execution of the process is managed to produce work products that are documented and controlled and that meet their functional and non-functional requirements, in line with the work product quality goals of the process. In order to achieve this capability, a process needs to have stated functional and non-functional requirements, including integrity, for work products and to produce work products that fulfil the stated requirements. The related Management Practices are:

[13] Tests can be developed during processes *Develop software design* and *Implement software design*. Commencement of test development should generally not wait until software integration.

Management practices
Identify requirements for the integrity and quality of the work products.
Identify the activities needed to achieve the integrity and quality requirements for work products.
Manage the configuration of work products to ensure their integrity.
Manage the quality of work products to ensure that the work products meet their functional and non-functional requirements.

3.2.4.3 Process definition attribute

This is defined as the extent to which the execution of the process uses a process definition based upon a standard process, that enables the process to contribute to the defined business goals of the organization. In order to achieve this capability, a process needs to be executed according to a standard process definition that has been suitably tailored to the needs of the process instance. The standard process needs to be capable of supporting the stated business goals of the organization. The related Management Practices are:

Management practices
Identify the standard process definition from those available in the organization that is appropriate to the process purpose and the business goals of the organization.
Tailor the standard process to obtain a defined process appropriated to the process context.
Implement the defined process to achieve the process purpose consistently, and repeatably, and support the defined business goal of the organization.
Provide feedback into the standard process from experience of using the defined process.

3.2.4.4 Process resource attribute

This is defined as the extent to which the execution of the process uses suitable skilled human resources and process infrastructure effectively to contribute to the defined business goals of the organization. In order to achieve this capability, a process needs to have adequate human resources and process infrastructure available that fulfil stated needs to execute the defined process. The related Management Practices are:

Management practices
Define the human resource competencies required to support the implementation of the defined process.
Define process infrastructure requirements to support the implementation of the defined process.
Provide adequate skilled human resources meeting the defined competencies.
Provide adequate process infrastructure according to the defined needs of the process.

3.3 Summary

In this section we have presented a summary of the two-dimensional ISO/IEC 15504 architecture. An important element of that architecture for our purposes is the process capability measurement scheme.

This consists of 9 attributes that are each rated on a 4-point "achievement" scale. We are only interested in the first 5 of these (i.e., the first three levels) since these are the ones we use in our study. We also presented some details about the type of information that an assessor would typically look for when making a rating on each of these five attributes.

4 Research Method

4.1 Approaches to Evaluating Predictive Validity in Correlational Studies

		Measuring the Criterion		
		Questionnaire	**Measurement Program**	
Measuring Capability	**Questionnaire**	Q1	Q2	(low cost)
	Assessment	Q3	Q4	(high cost)
		(across organizations)	(within one organization)	

Table 4: Different correlational approaches for evaluating predictive validity.

Correlational approaches to evaluating the predictive validity of a process capability measure can be classified by the manner in which the variables are measured. Table 4 shows a classification of approaches. The columns indicate the manner in which the criterion (i.e., performance) is measured. The rows indicate the manner in which the process capability is measured. The criterion can be measured using a questionnaire whereby data on the perceptions of experts are collected. It can also be measured through a measurement program. For example, if our criterion is defect density of delivered software products, then this could be measured through an established measurement program that collects data on defects found in the field. Process capability can also be measured through a questionnaire whereby data on the perceptions of experts on the capability of their processes are collected. Alternatively, actual assessments can be performed, which are a more rigorous form of measurement[14].

A difficulty with studies that attempt to use criterion data that are collected through a measurement program is that the majority of organizations do not collect objective process and product data (e.g., on defect levels, or even keep accurate effort records). Primarily organizations that have made improvements and reached a reasonable level of process capability will have the actual objective data to demonstrate improvements (in productivity, quality, or return on investment).[15] This assertion is supported by the results in (Brodman and Johnson, 1995) where, in general, it was found that organizations at lower SW-CMM maturity levels are less likely to collect quality data (such as the number of development defects). Also, the same authors found that organizations tend to collect more data as their CMM maturity

[14] "More rigorous" is intended to mean with greater reliability and construct validity.

[15] This is the same disadvantage of case studies as described in Section 2.3.

levels rise. It was also reported in another survey (Rubin, 1993) that for 300 measurement programs started since 1980, less than 75 were considered successful in 1990, indicating a high mortality rate for measurement programs. This high mortality rate indicates that it may be difficult right now to find many organizations that have implemented measurement programs. This means that organizations or projects with low process capability would have to be excluded from a correlational study. Such an exclusion would reduce the variation in the performance measure, and thus reduce (artificially) the validity coefficients. Therefore, correlational studies that utilize objective performance measures are inherently in greater danger of not finding significant results.

Furthermore, when criterion data are collected through a measurement program, it is necessary to have the criterion measured in the same way across all observations. This usually dictates that the study is done within a single organization where such measurement consistency can be enforced, hence reducing the generalizability of the results.

Conducting a study where capability is measured through an assessment as opposed to a questionnaire implies greater costs. This usually translates into smaller sample sizes and hence reduced statistical power.

Therefore, the selection of a quadrant in Table 4 is a tradeoff amongst cost, statistical power, measurement rigor, and generalizability.

There are a number of previous studies that evaluated the relationship between process capability (or organizational maturity) and the performance of projects that can be placed in quadrant Q1. For example, (Goldenson and Herbsleb, 1995; Deephouse et al., 1995; Clark, 1997; Gopal et al., 1999). These studies have the advantage that they can be conducted across multiple projects and across multiple organizations, and hence can produce more generalizable conclusions.

A more recent study evaluated the relationship between questionnaire responses on implementation of the SW-CMM KPA's and defect density (Krishnan and Kellner, 1998), and this would be placed in quadrant Q2. However, this study was conducted across multiple projects within a single organization, reducing its generalizability compared with studies conducted across multiple organizations.

Our current study can be placed in quadrant Q3 since we use process capability measures from actual assessments, and questionnaires for evaluating project performance. This retains the advantage of studies in quadrant Q1 since it is conducted across multiple projects in multiple organizations, but utilizes a more rigorous measure of process capability. Similarly, the study of Jones can be considered to be in this quadrant (Jones, 1996; Jones, 1999).[16]

[16] Since it is difficult to find low maturity organizations with objective data on effort and defect levels, and since there are few high maturity organizations, Jones' data relies on the reconstruction of, at least, effort data from memory, as noted in (Jones, 1994): "The SPR approach is to ask the project team to reconstruct the missing elements from memory." The rationale for that is stated as "the alternative is to have null data for many important topics, and that would be far worse." The general approach is to show staff a set of standard activities, and then ask them questions such as which ones they used and whether they put in any unpaid overtime during the performance of these activities. For defect levels, the general approach is to do a matching between companies that do

Studies in quadrant Q4 are likely to have the same limitations as studies in quadrant Q2: being conducted across multiple projects within the same organization. For instance, the study of McGarry et al was conducted within a single company (McGarry et al., 1998), the AFIT study was conducted with contractors of the Air Force (Flowe and Thordahl, 1994; Lawlis et al., 1996), and the study by Harter et al. (Harter et al., 1999) was performed on 30 software products created by the systems integration division within one organization.

Therefore, the different types of studies that can be conducted in practice have different advantages and disadvantages, and predictive validity studies have been conducted in the past that populate all four quadrants. It is reasonable then to encourage studies in all four quadrants. Consistency in the results across correlational studies that use the four approaches would increase the weight of evidence supporting the predictive validity hypothesis.

4.2 Source of Data

The data that was used for this study was obtained from the SPICE Trials. The SPICE Trials are an international effort to empirically evaluate the emerging ISO/IEC 15504 international standard world wide. The SPICE Trials have been divided into three broad phases to coincide with the stages that the ISO/IEC 15504 document was expected to go through on its path to international standardization. The analyses presented in this paper come from phase 2 of the SPICE Trials. Phase 2 lasted from September 1996 to June 1998.

During the trials, organizations contribute their assessment ratings data to an international trials database located in Australia, and also fill up a series of questionnaires after each assessment. The questionnaires collect information about the organization and about the assessment. There is a network of 26 SPICE Trials co-ordinators around the world who interact directly with the assessors and the organizations conducting the assessments. This interaction involves ensuring that assessors are qualified, making questionnaires available, answering queries about the questionnaires, and following up to ensure the timely collection of data.

not measure their defects with similar companies that do measure, and then extrapolate for those that don't measure. It should be noted that SPR does have a large data base of project and organizational data, which makes this kind of matching defensible. However, since at least some of the criterion measures are not collected from measurement programs, we place this study in the same category as those that utilize questionnaires.

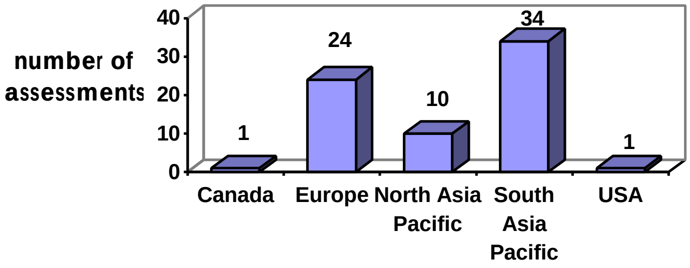

Figure 3: Distribution of assessments by region.

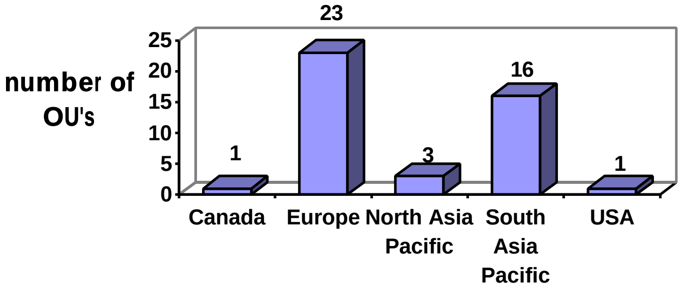

Figure 4: Distribution of assessed OUs by region.

During Phase 2 of the SPICE Trials a total of 70 assessments had been conducted and contributed their data to the international database. The distribution of assessments by region is given in Figure 3.[17] In total 691 process instances were assessed. Since more than one assessment may have occurred in a particular OU (e.g., multiple assessments each one looking at a different set of processes), a total of 44 OUs were assessed. Their distribution by region is given in Figure 4.

Given that an assessor can participate in more than one assessment, the number of assessors is smaller than the total number of assessments. In total, 40 different lead assessors took part.

The employment status of the assessors is summarized in Figure 5. As can be seen, most assessors consider themselves in management or senior technical positions in their organizations, with a sizeable number of the rest being consultants.

[17] Within the SPICE Trials, assessments are coordinated within each of the five regions shown in the figures above.

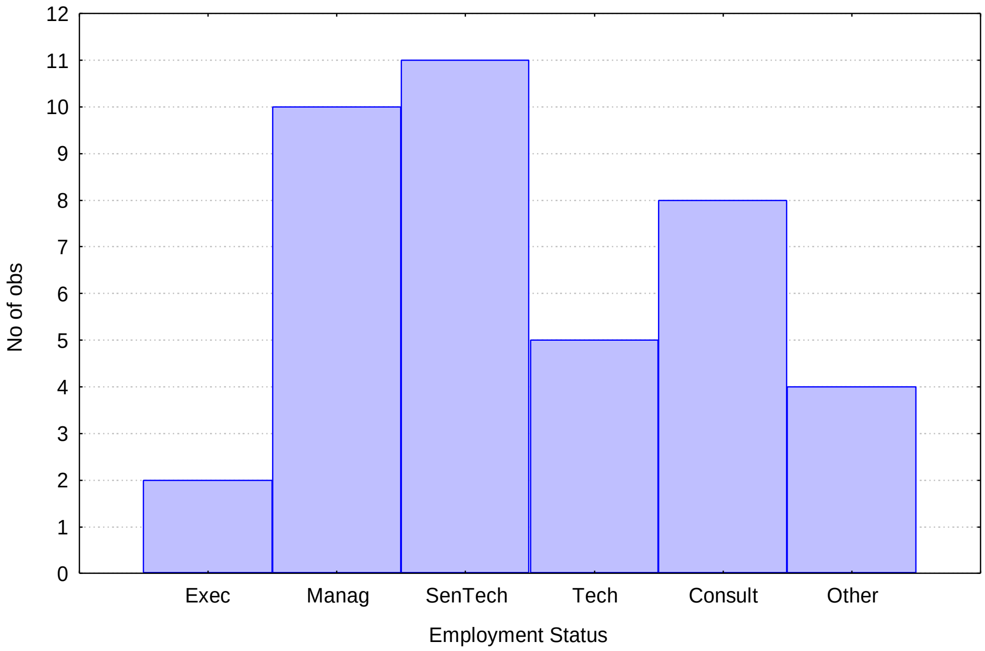

Figure 5: Employment status of (lead) assessors. The possible options were as follows: "Exec" stands for executive, "Manag" stands for management, "SenTech" stands for senior technical person, "Tech" stands for a technical position, and "Consult" stands for consultant.

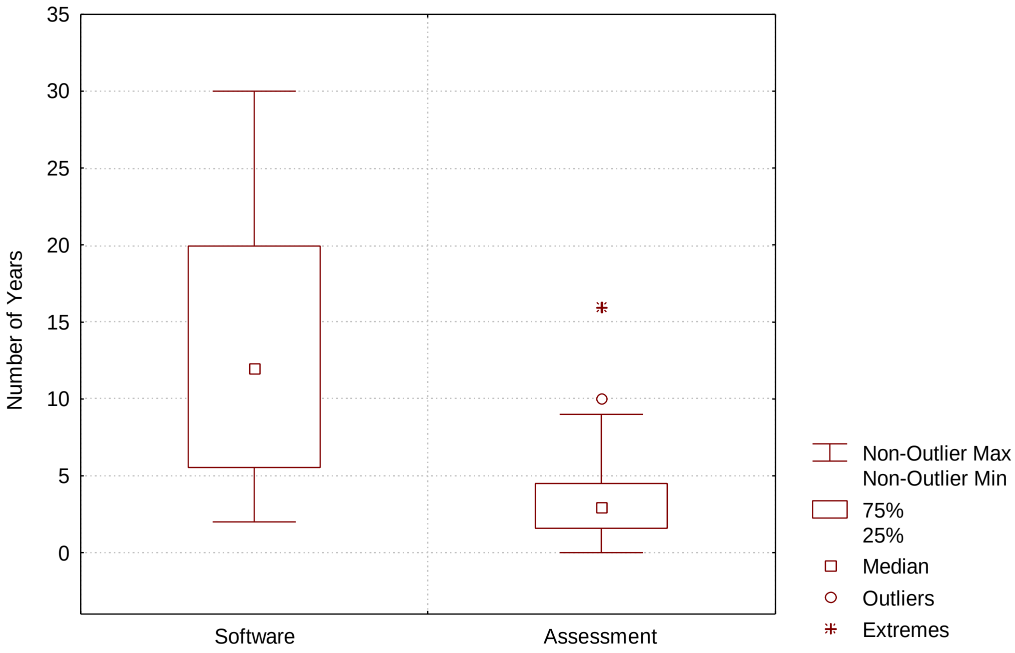

Figure 6: Software engineering (left panel) and assessment experience (right panel) of the (lead) assessors who participated in the SPICE Trials. The figure is a box and whisker plot. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

The variation in the number of years of software engineering experience and assessment experience of the assessors is shown in Figure 6. The median experience in software engineering is 12 years, with a maximum of 30 years experience. The median experience in assessments is 3 years, indicating a non-trivial background in assessments.

The median number of assessments performed in the past by the assessors is 6, and the median number of 15504-based assessments is 2. This indicates that, in general, assessors had a good amount of experience with software process assessments.

4.3 Unit of Analysis

The unit of analysis for this study is the software project. This means that process capability ratings are obtained for the relevant process in each project, and project performance measures are collected for the same project.

4.4 Measurement

4.4.1 Measuring Process Capability

A previous study had identified that the capability scale of ISO/IEC 15504 is two dimensional (El Emam, 1998).[18] The first dimension, which was termed "Process Implementation", consists of the first three levels. The second dimension, which was termed "Quantitative Process Management", consists of levels 4 and 5. It was also found that these two dimensions are congruent with the manner in which assessments are conducted in practice: either only the "Process Implementation" dimension is rated or both dimensions are rated (recall that it is not required to rate at all five levels in an ISO/IEC 15504 assessment).

In our data set, 33% of the Develop Software Design processes, 31% of the Implement Software Design processes, and 44% of the Integrate and Test Software processes were not rated on the "Quantitative Process Management" dimension. If we exclude all processes with this rating missing then we lose a substantial proportion of our observations. Therefore, we limit ourselves in the current study to the first dimension only.

To construct a single measure of "Process Implementation" we code an 'F' rating as 4, down to a 1 for an 'N' rating. Subsequently, we construct an unweighted sum of the attributes at the first three levels of the capability scale. This is a common approach for the construction of summated rating scales (McIver and Carmines, 1981). The range of this summated scale is 4 to 20.

[18] A similar study was performed by Curtis (Curtis, 1996) to identify the underlying dimensions of the SW-CMM practices, and a number of different dimensions were identified. El Emam and Goldenson (El Emam and Goldenson, 2000) reanalyzed the data in (Clark, 1997) also to identify the underlying dimensions in the SW-CMM. Gopal et al. also identified two dimensions of a subset of the SW-CMM KPAs (Gopal et al., 1999). Therefore, current literature does clearly signify that process capability is indeed a multidimensional construct.

4.4.2 Measuring Project Performance

The performance measures were collected through a questionnaire. The respondent to the questionnaire was the sponsor of the assessment, who should be knowledgeable about the projects that were assessed. In cases where the sponsor was not able to respond, s/he delegated the task to a project manager or senior technical person who completed the questionnaire.

To maintain comparability with previous studies, we define project performance in a similar manner. In the Goldenson and Herbsleb study (Goldenson and Herbsleb, 1995) performance was defined in terms of six variables: customer satisfaction, ability to meet budget commitments, ability to meet schedule commitments, product quality, staff productivity, and staff morale / job satisfaction. We use these six variables, except that product quality is generalized to "ability to satisfy specified requirements". We therefore define project performance in terms of the six variables summarized in Table 5. Deephouse et al. (Deephouse et al., 1995) consider software quality (defined as match between system capabilities and user requirements, ease of use, and extent of rework), and meeting targets (defined as within budget and on schedule). One can argue that if "ease of use" is not in the requirements then it ought not be a performance criterion, therefore we can consider it as being a component of satisfying specified requirements. Extent of rework can also be considered as a component of productivity since one would expect productivity to decrease with an increase in rework. Therefore, these performance measures are congruent with our performance measures, and it is clear that they represent important performance criteria for software projects.

Definition	**Variable Name**
Ability to meet budget commitments	BUDGET
Ability to meet schedule commitments	SCHEDULE
Ability to achieve customer satisfaction	CUSTOMER
Ability to satisfy specified requirements	REQUIREMENTS
Staff productivity	PRODUCTIVITY
Staff morale / job satisfaction	MORALE

Table 5: The criterion variables that were studied. These were evaluated for every project. The question was worded as follows: "How would you judge the process performance on the following characteristics …". The response categories were: "Excellent", "Good", "Fair", "Poor", and "Don't Know".

The responses were coded such that the "Excellent" response category is 4, down to the "Poor" response category which was coded 1. The "Don't Know" responses were treated as missing values.[19] The implication of this coding scheme is that all investigated relationships are hypothesized to be positive.

[19] It is not uncommon to treat "Don't Know" (DK) responses as missing values when there is no intrinsic interest in the fact that a DK response has been provided (Rubin et al., 1995).

4.5 Data Analysis

4.5.1 Evaluating the Relationships

A common coefficient for the evaluation of predictive validity in general is the correlation coefficient (Nunnally and Bernstein, 1994). It has also been used in the context of evaluating the predictive validity of project and organizational process capability measures (McGarry et al, 1998; El Emam and Madhavji, 1995). We therefore use this coefficient in our study to indicate the magnitude of a relationship.

We follow a two staged analysis procedure. During the first stage we determine whether the association between "Process Implementation" of the development processes and each of the performance measures is "clinically significant" (using the Pearson correlation coefficient). This means that it has a magnitude that is sufficiently large. If it does, then we test the statistical significance of the association. The logic of this is explained below.

It is known that with a sufficiently large sample size even very small associations can be statistically significant. Therefore, it is also of import to consider the magnitude of a relationship to determine whether it is meaningfully large. Cohen has provided some general guidelines for interpreting the magnitude of the correlation coefficient (Cohen, 1988). We consider "medium" sized (i.e., r =0.3) correlations as the minimal magnitude that is worthy of consideration. The logic behind this choice is that of elimination. If we take "small" association (i.e., r = 0.1) as the minimal worthy of consideration we may be being too liberal and giving credit to weak associations that are not congruent with the broad claims made for the predictive validity of assessment scores. Using a "large" association (i.e., r = 0.5) as the minimal value worthy of consideration may place a too high expectation on the predictive validity of assessment scores; recall that many other factors are expected to influence the success of a software project apart from the capability of the development processes.

In the social sciences predictive validity studies with one predictor rarely demonstrate correlation coefficients exceeding 0.3 to 0.4 (Nunnally and Bernstein, 1994). The rationale is that "people are far too complex to permit a highly accurate estimate of their proficiency in most performance-related situations from any practicable collection of test materials" (Nunnally and Bernstein, 1994). Therefore, we would expect that such guidelines would be at least equally applicable to studies of projects and organizations.

For statistical significance testing, we perform an ordinary least squares regression:

$$PERF = a_0 + a_1 CAP$$

where $PERF$ is the performance measure according to Table 5 and CAP is the "Process Implementation" dimension of process capability. We test whether the a_1 regression coefficient is different from zero. If there is sufficient evidence that it is, then we claim that CAP is associated with $PERF$. The above model is constructed separately for each of the performance measures. All tests performed were one-tailed since our hypotheses are directional.

4.5.2 Scale Type Assumption

According to some authors, one of the assumptions of the OLS regression model is that all the variables should be measured at least on an interval scale (Bohrnstedt and Carter, 1971). This assumption is based on the mapping originally developed by Stevens (Stevens, 1951) between scale types and "permissible" statistical procedures. In our context, this raises two questions. First, what are the levels of our measurement scales? Second, to what extent can the violation of this assumption have an impact on our results?

The scaling model that is used in the measurement of the process capability construct is the summative model (McIver and Carmines, 1981). This consists of a number of subjective measures each on a 4-point scale that are summed up to produce an overall measure of the construct. Some authors state that summative scaling produces interval level measurement scales (McIver and Carmines, 1981), while others argue that this leads to ordinal level scales (Galletta and Lederer, 1989). In general, however, our process capability is expected to occupy the grey region between ordinal and interval level measurement.

Our criterion measures utilized a single item each. In practice, single item measures are treated as if they are interval in many instances. For example, in the construction and empirical evaluation of the User Information Satisfaction instrument, inter-item correlations and principal components analysis are commonly performed (Ives et al., 1983).

It is also useful to note a study by Spector (Spector, 1980) that indicated that whether scales used have equal or unequal intervals does not actually make a practical difference. In particular, the mean of responses from using scales of the two types do not exhibit significant differences, and that the test-retest reliabilities (i.e., consistency of questionnaire responses when administered twice over a period of time) of both types of scales are both high and very similar. He contends, however, that scales with unequal intervals are more difficult to use, but that respondents conceptually adjust for this.

Given the proscriptive nature of Stevens' mapping, the permissible statistics for scales that do not reach an interval level are distribution-free (or nonparametric) methods (as opposed to parametric methods, of which OLS regression is one) (Siegel and Castellan, 1988). Such a broad proscription is viewed by Nunnally as being "narrow" and would exclude much useful research (Nunnally and Bernstein, 1994). Furthermore, studies that investigated the effect of data transformations on the conclusions drawn from parametric methods (e.g., F ratios and t tests) found little evidence supporting the proscriptive viewpoint (Labovitz, 1970; Labovitz, 1967; Baker et al., 1966). Suffice it to say that the issue of the validity of the above proscription is, at best, debatable. As noted by many authors, including Stevens himself, the basic point is that of pragmatism: useful research can still be conducted even if, strictly speaking, the proscriptions are violated (Stevens, 1951; Bohrnstedt and Carter, 1971; Gardner, 1975; Velleman and Wilkinson, 1993). A detailed discussion of this point and the literature that supports our argument is given in (Briand et al., 1996).

4.5.3 Multiple Hypothesis Testing

Since we are performing multiple hypotheses testing (i.e., a regression model for each of the six performance measures), it is plausible that many a_1 regression coefficients will be found to be statistically significant since the more null hypothesis tests that one performs, the greater the probability of finding statistically significant results by chance. We therefore use a Bonferonni adjusted alpha level when performing significance testing (Rice, 1995). We set our overall alpha level to be 0.1.

4.5.4 Organization Size Context

It was noted earlier that the relationships may be of different magnitudes for small vs. large organizations. We therefore perform the analysis separately for small and large organizations. Our definition of size is the number of IT staff within the OU. We dichotomize this IT staff size into SMALL and LARGE organizations, whereby small is equal to or less than 50 IT staff. This is the same definition of small organizations that has been used in a European Commission project that is providing process improvement guidance for small organizations (The SPIRE Project, 1998).

4.5.5 Reliability of Measures

It is known that lack of reliability in measurement can attenuate bivariate relationships (Nunnally and Bernstein, 1994). It is therefore important to evaluate the reliability of our subjective measures, and if applicable, make corrections to the correlation coefficient that take into account reliability.

In another related scientific discipline, namely Management Information Systems (MIS), researchers tend to report the Cronbach alpha reliability coefficient (Cronbach, 1951) most frequently (Subramanian and Nilakanta, 1994). Also, it is considered by some researchers to be the most important reliability estimation approach (Sethi and King, 1991). This coefficient evaluates a certain type of reliability called internal consistency, and has been used in the past to evaluate the reliability of the ISO/IEC 15504 capability scale (El Emam, 1998; Fusaro et al., 1997). We also calculate the Cronbach alpha coefficient for the development process capability measures.

The Cronbach alpha coefficient varies between 0 and 1, where 1 is perfect reliability. Nunnally and Bernstein (Nunnally and Bernstein, 1994) recommend that a coefficient value of 0.8 is a minimal threshold for applied research settings, and a minimal threshold of 0.7 for basic research settings.

In our study we do not incorporate corrections for attenuation due to less than perfect reliability on the process capability measure, however. As suggested in (Nunnally, 1978), it is preferable to use the unattenuated correlation coefficient since this reflects the predictive validity of the process capability measure that will be used in actual practice (i.e., in practice it will have less than perfect reliability).

4.5.6 Multiple Imputation

In the performance measures that we used (see Table 5) there were some missing values. Missing values are due to respondents not providing an answer on all or some of the performance questions, or they selected the “Don’t Know” response category. Ignoring the missing values and only analyzing the completed data subset can provide misleading results (Little and Rubin, 1987). We therefore employ the method of multiple imputation to fill in the missing values repeatedly (Rubin, 1987). Multiple imputation is a preferred approach to handling missing data problems in that it provides for proper estimates of parameters and their standard errors.

The basic idea of multiple imputation is that one generates a vector of size M for each value that is missing. Therefore an $n_{mis} \times M$ matrix is constructed, where n_{mis} is the number of missing values. Each column of this matrix is used to construct a complete data set, hence one ends up with M complete data sets. Each of these data sets can be analyzed using complete-data analysis methods. The M analyses are then combined into one final result. Typically a value for M of 3 is used, and this provides for valid inference (Rubin and Schenker, 1991). Although, to err on the conservative side, some studies have utilized an M of 5 (Treiman et al., 1988), which is the value that we use.

For our analysis the two parameters of interest are the correlation coefficient, r, and the a_1 parameter of the regression model (we shall refer to this estimated parameter as $\hat{Q}$). Furthermore, we are interested in the standard error of $\hat{Q}$, which we shall denote as $\sqrt{U}$, in order to test the null hypothesis that it is equal to zero. After calculating these values for each of the 5 data sets, they can be combined to give an overall r value, $\bar{r}$, an overall value for $\hat{Q}$, $\overline{Q}$, and its standard error $\sqrt{T}$. Procedures for performing this computation are detailed in (Rubin, 1987), and summarized in (Rubin and Schenker, 1991). In Section 7 we describe the multiple imputation approach in general, its rationale, and how we operationalized it for our specific study.

4.6 Summary

In this section we presented the details of how data was collected, how measures were defined, and how the data was analyzed. In brief, the data was collected from phase 2 of the SPICE Trials. During the trials the process capability measures were obtained from 70 actual assessments. Six performance measures were defined, and data on the performance of each project were collected from the sponsor (or their designate) using a questionnaire during the assessment. We analyze the data separately for small and large organizations. Our analysis method consists of constructing an ordinary least squares regression model for each performance variable. We first determine whether the correlation is meaningfully large, and if it is, we test the statistical significance of the slope of the relationship between capability and performance.

5 Results

5.1 Description of Projects and Assessments

In this section we present some descriptive statistics on the projects that were assessed, and on the assessments themselves. In the SPICE Phase 2 trials, a total of 44 organizations participated. Their primary business sector distribution is summarized in Figure 7. As can be seen, the most frequently occurring categories are Defence, IT Products and Services, and Software Development organizations.

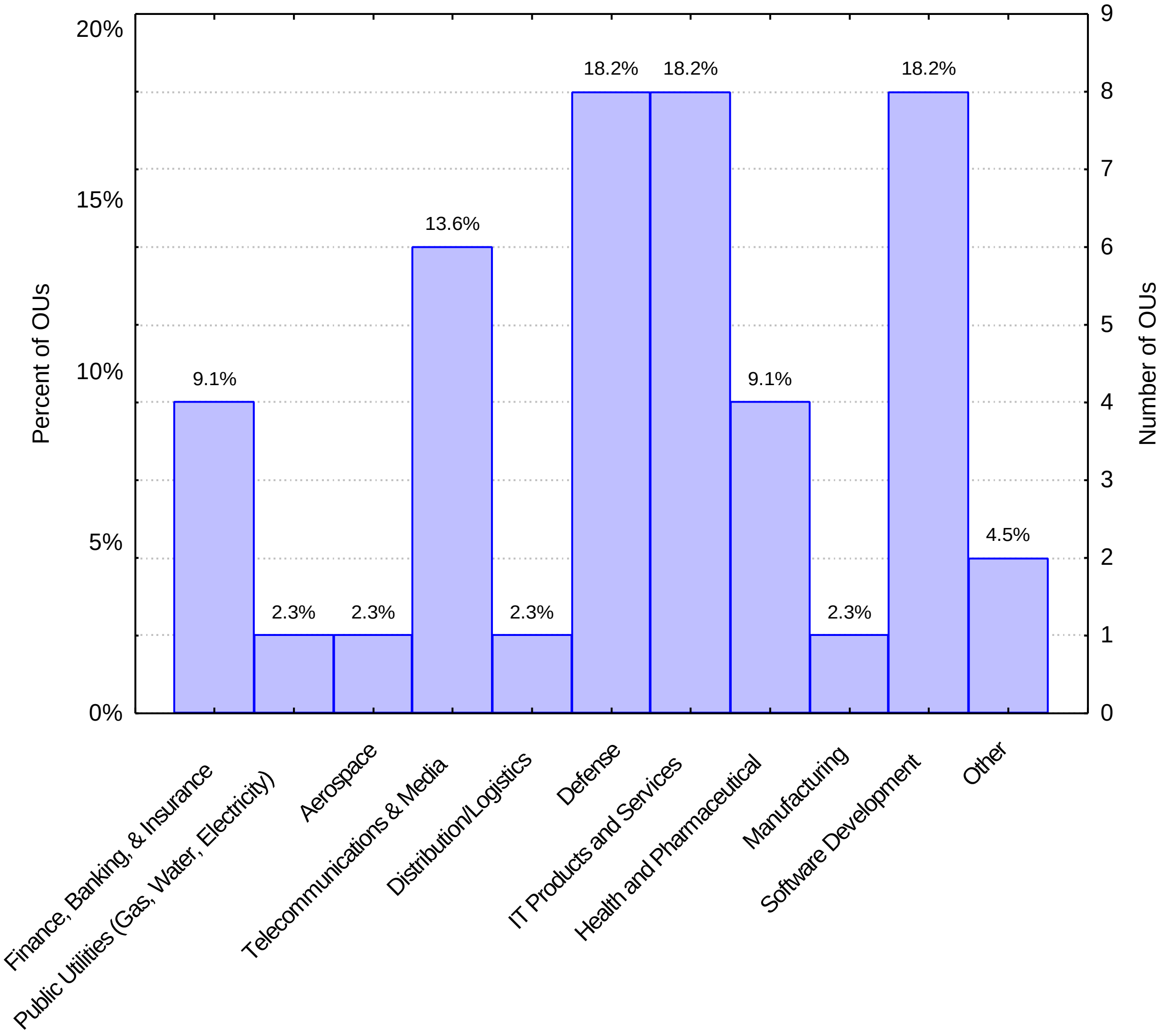

Figure 7: Business sector of all organizations that took part in SPICE Trials Phase 2 assessments (n=44).

Since it is not necessary that an assessment include all development processes within its scope, it is possible that the number of assessments that cover a particular development process is less than the total number of assessments, and it is also possible that a different number of OUs assess each of the development processes. Furthermore, since it is possible that an assessment's scope covers more than one project, the number of projects is not necessarily equal to the number of OUs assessed. Table 6 shows the number of OUs that assessed each of the development processes, the number of projects that were actually assessed, and the number of projects in small versus large OUs.

	Number of OUs	Number of Projects	Number of Projects in Small OUs	Number of Projects in Large OUs
Develop Software Design	25	45	18	27
Implement Software Design	18	32	18	14
Integrate and Test Software	25	36	18	18

Table 6: Number of OUs and projects that assessed each of the three software development processes, and their breakdown into small vs. large organizations.

5.1.1 Develop Software Design

Figure 8 shows the primary business sector distributions for those 25 organizations that assessed the Develop Software Design process. The most frequently occurring categories are similar to those for all organizations that participated in the trials. However, three categories disappeared altogether as well: Public Utilities, Health and Pharmaceutical, and Manufacturing.

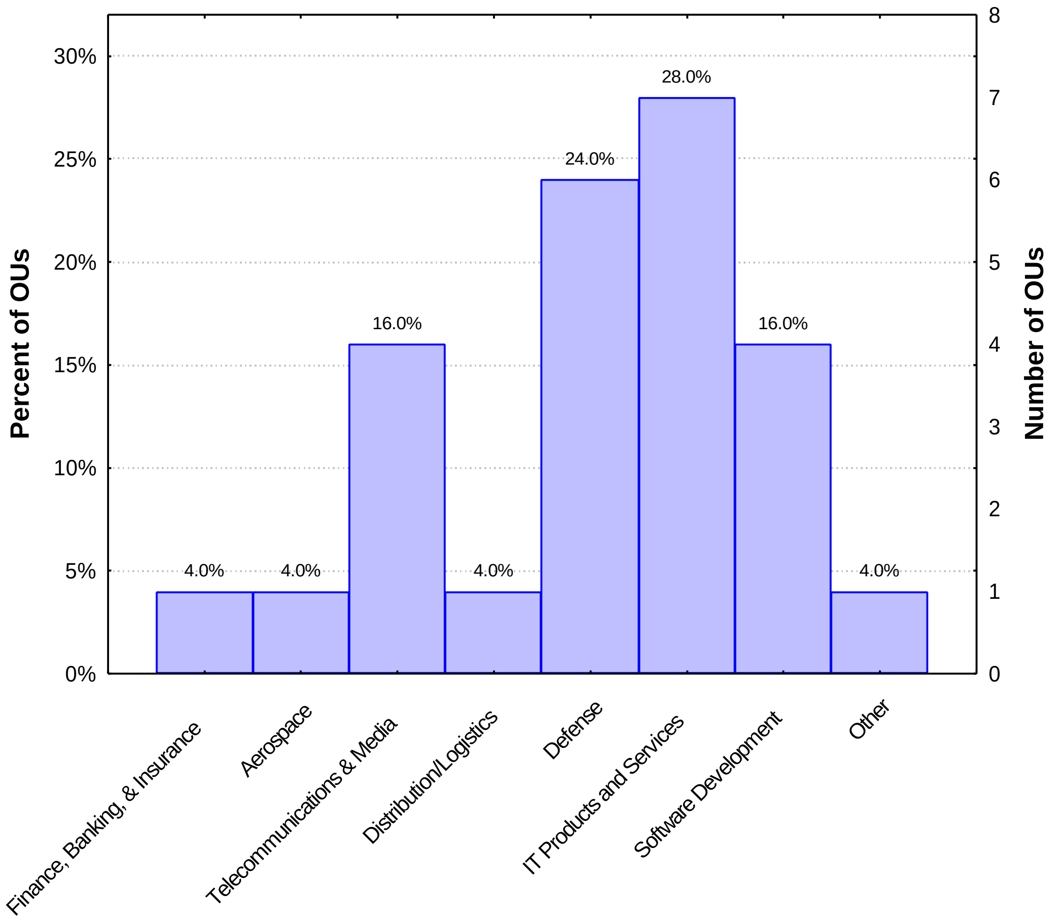

Figure 8: Business sector of all OUs that assessed the Develop Software Design process.

Of the 25 OUs, they were distributed by country as follows: Australia (12), Canada (1), Italy (2), France (2), Turkey (1), South Africa (4), Germany (1), and Japan (2). Of these 9 were not ISO 9001 registered, and 16 were ISO 9001 registered.

Figure 9 shows the variation in the number of projects that were assessed in each OU. The median value is one project per OU, although in one case an OU assessed the Develop Software Design process in six different projects.

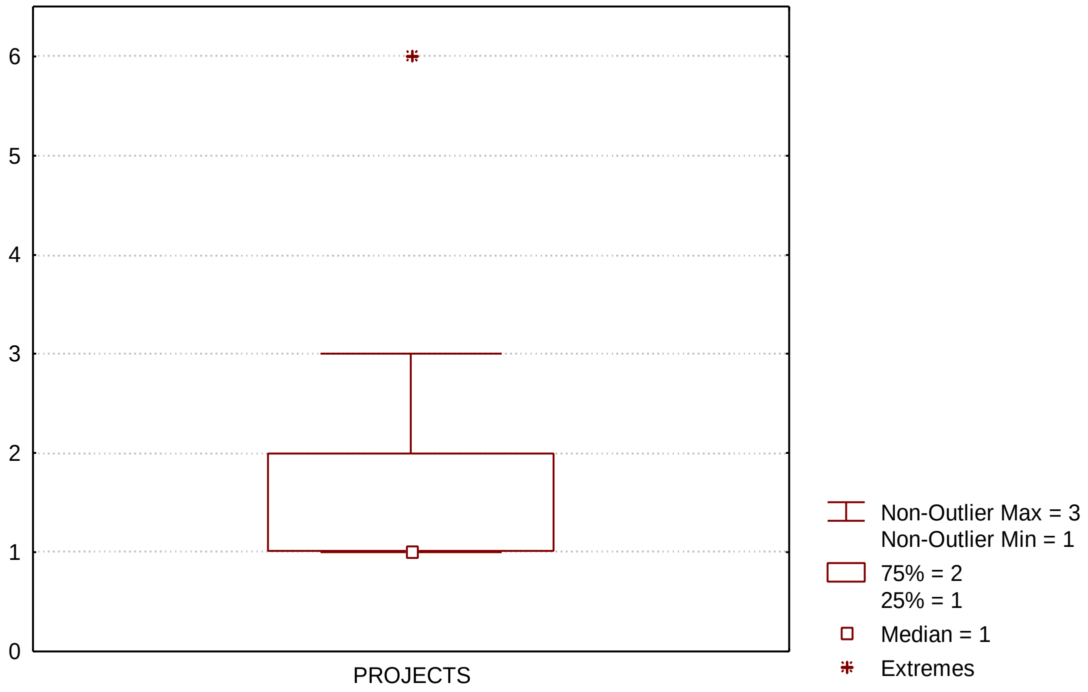

Figure 9: Box and whisker plot showing the variation in the number of projects that assessed their Develop Software Design process in each OU. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

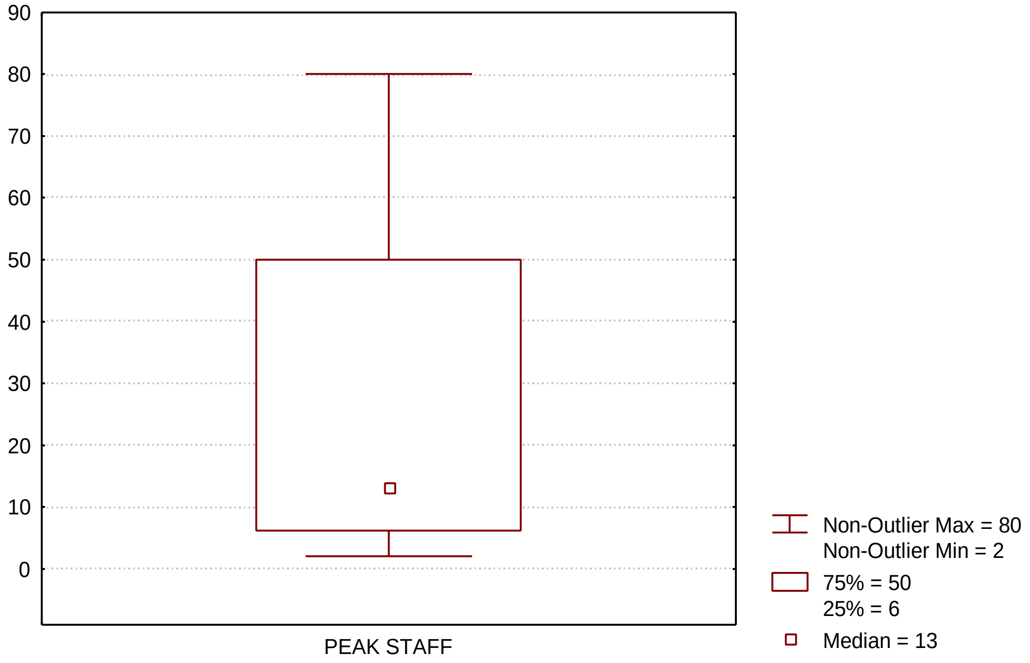

Figure 10: Box and whisker plot showing the variation in the peak staff load for projects that assessed the Develop Software Design process. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

The distribution of peak staff load for the projects that assessed the Develop Software Design process is shown in Figure 10. The median value is 13, with a minimum of two and a maximum of 80 staff. Therefore, there is nonnegligible variation in terms of project staffing.

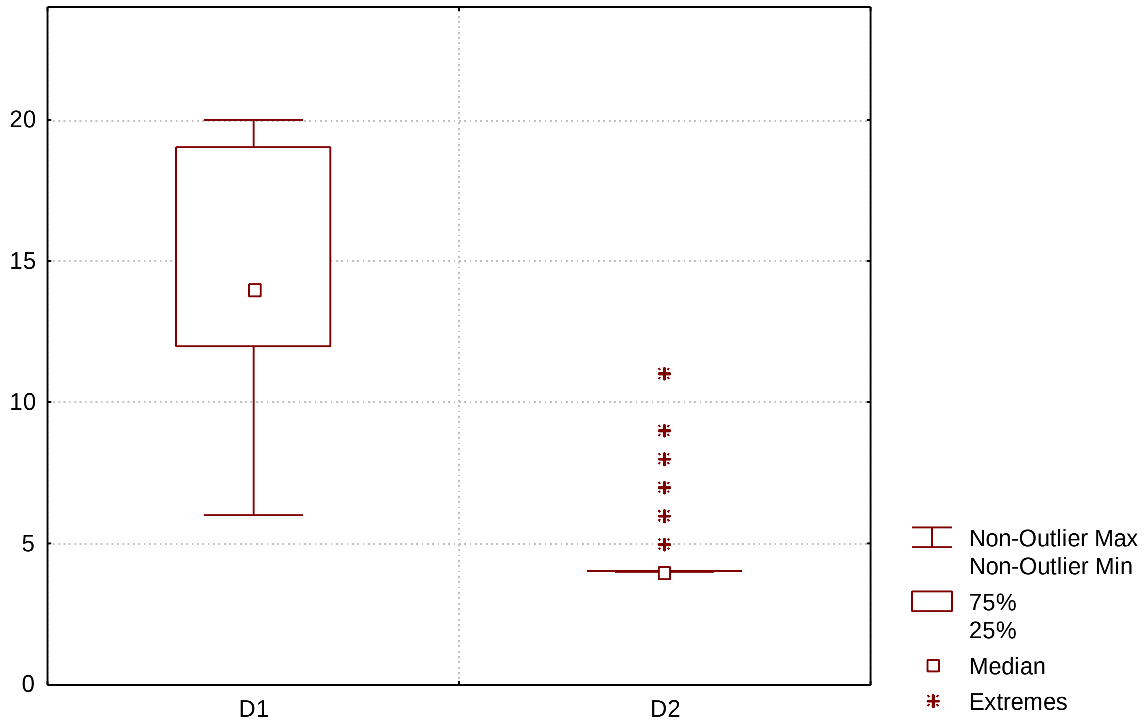

Figure 11: Box and whisker plot showing the variation in the measures of the two dimensions of Develop Software Design process capability. The first dimension, denoted D1, is “Process Implementation” and consists of levels 1 to 3. The second dimension, denoted D2, is “Quantitative Process Management” and consists of levels 4 and 5. The D2 measure used the same coding scheme as for the D1 measure, except that it consists of only 4 attributes. For the second dimension it is assumed that processes that were not rated would receive a rating of N (Not Achieved) if they would have been rated. This was done to ensure that the sample size for both dimensions was the same. Note that it is common practice not to rate the higher levels if there is strong *a priori* belief that the ratings will be N. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

In Figure 11 we can see the variation in the two measures of process capability. For the second dimension, D2 (Quantitative Process Management), there is little variation. The median of D2 is 4, which is the minimal value, indicating that at least 50% of projects do not implement any quantitative process management practices for their Develop Software Design process. On the first dimension, D1 (Process Implementation), there is substantially more variation with a median value of 14.

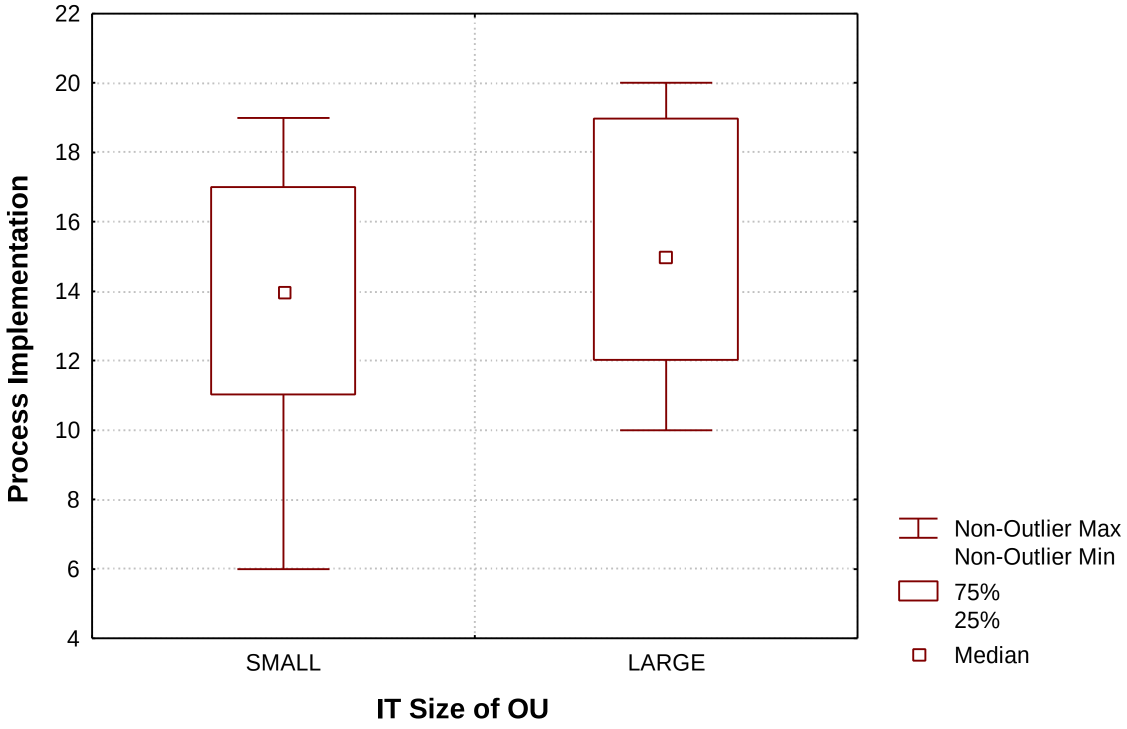

Figure 12: Box and whisker plot showing the variation in the "Process Implementation" capability dimension of the Develop Software Design process for the different OU sizes that were considered in our study (in terms of IT staff). A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

Figure 12 shows the variation along the D1 dimension for the two OU sizes that were considered during our study. Larger OUs tend to have greater implementation of Develop Software Design processes, although the difference is not marked.

5.1.2 Implement Software Design

Figure 13 shows the primary business sector distribution for those 18 organizations that assessed the Implement Software Design process. In this particular subset, the "Software Development" category dropped in occurrence compared with the overall trials participants. Moreover, three categories disappeared altogether: Finance, Banking & Insurance, Public Utilities, and Health and Pharmaceutical.

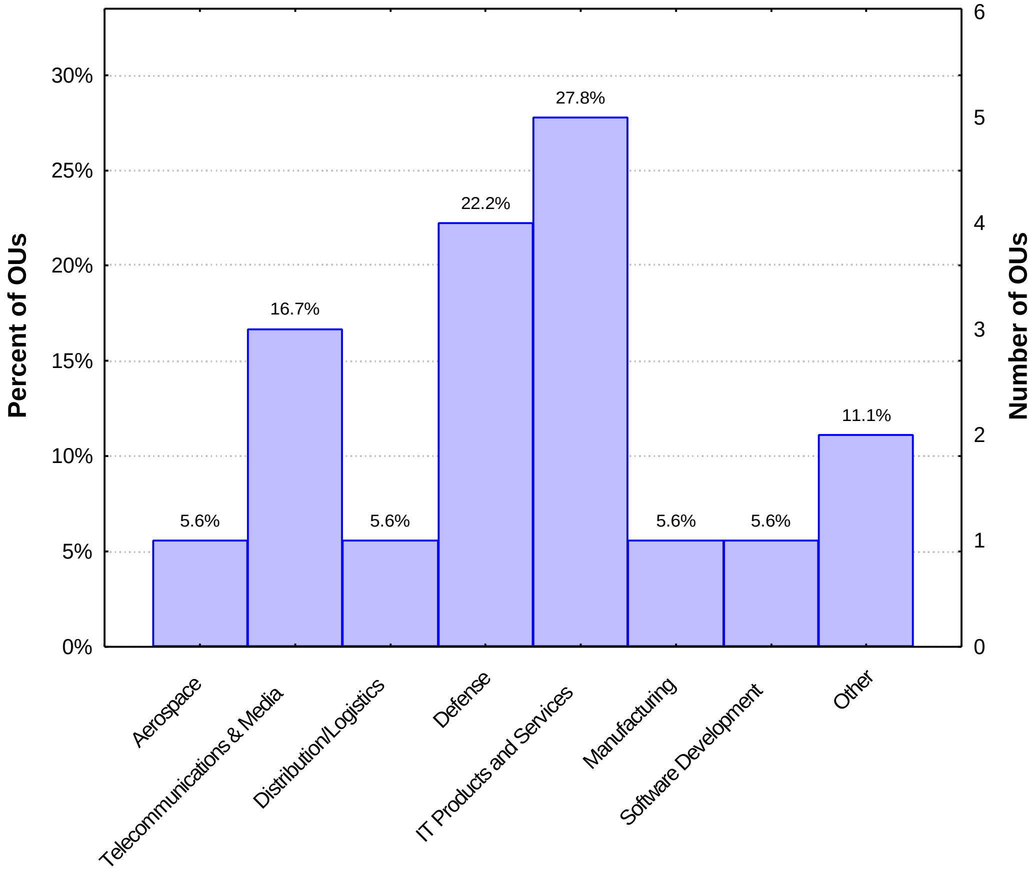

Figure 13: Business sector of all OUs that assessed the Implement Software Design process.

Of the 18 OUs, they were distributed by country as follows: Australia (5), Canada (1), Italy (2), Spain (1), Luxemburg (1), South Africa (4), France (1), Germany (1), Japan (2). Of these 8 were not ISO 9001 registered, and 10 were ISO 9001 registered.

Figure 14 shows the variation in the number of projects that were assessed in each OU. The median value is one project per OU, although in one case an OU assessed the Implement Software Design process in six different projects.

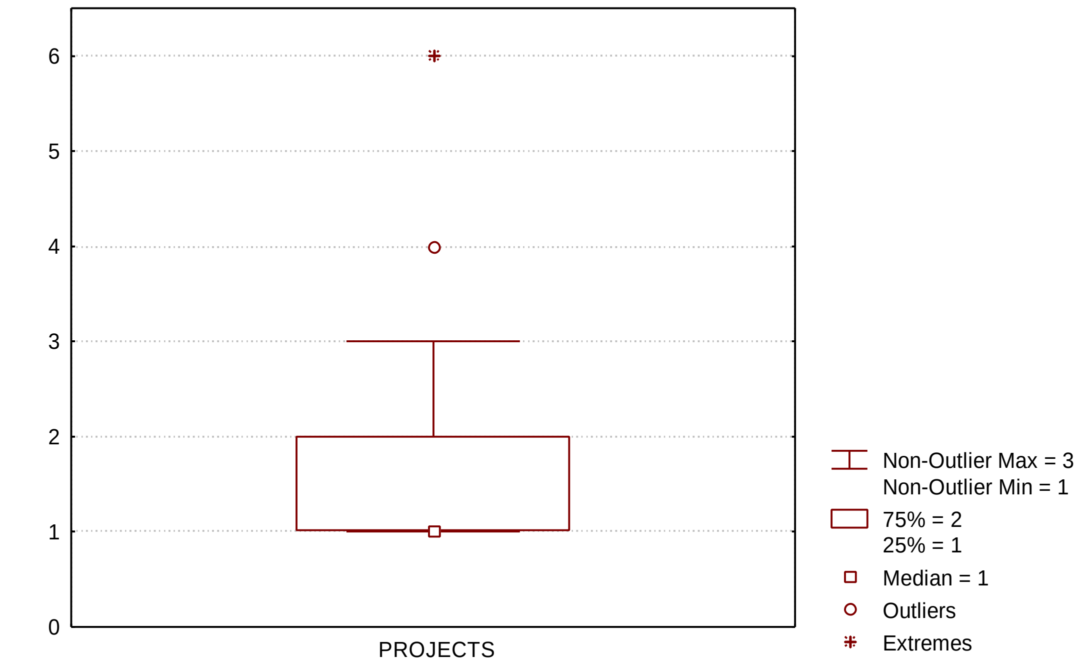

Figure 14: Box and whisker plot showing the variation in the number of projects that assessed their Implement Software Design process in each OU. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

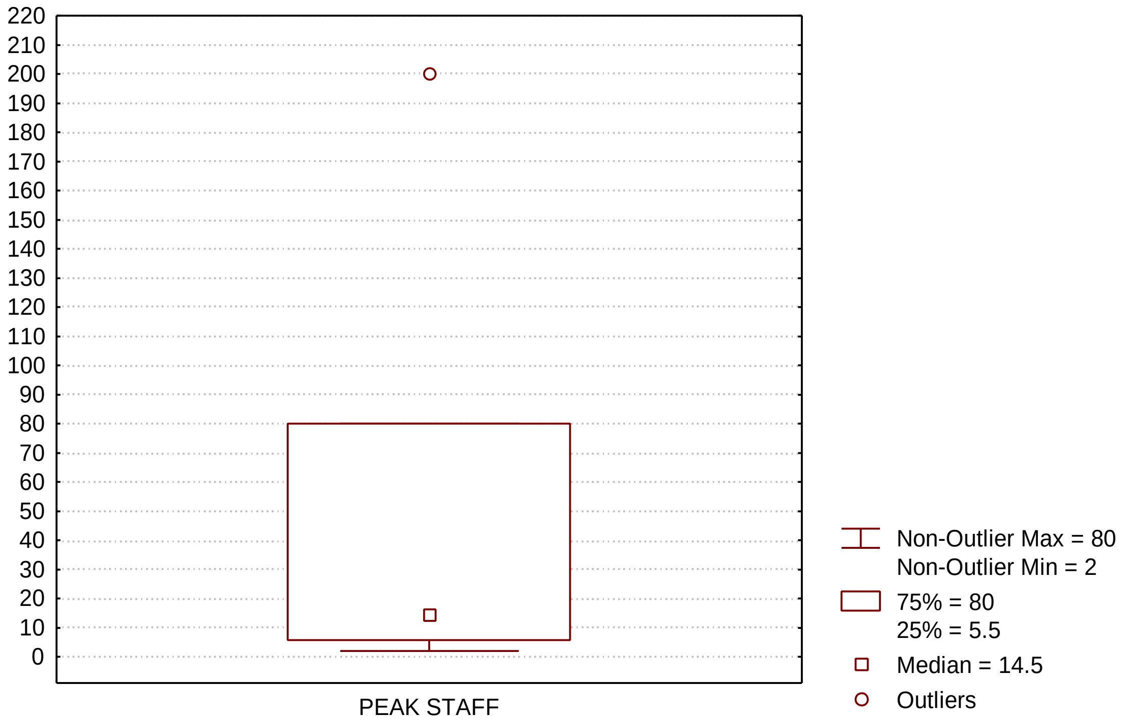

Figure 15: Box and whisker plot showing the variation in the peak staff load for projects that assessed the Implement Software Design process. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

The distribution of peak staff load for the projects that assessed the Implement Software Design process is shown in Figure 15. The median value is 14.5, with a minimum of two and a non-outlier maximum of 80 staff. Although, one project had a peak load of 200 staff. Therefore, there is nonnegligible variation in terms of project staffing.

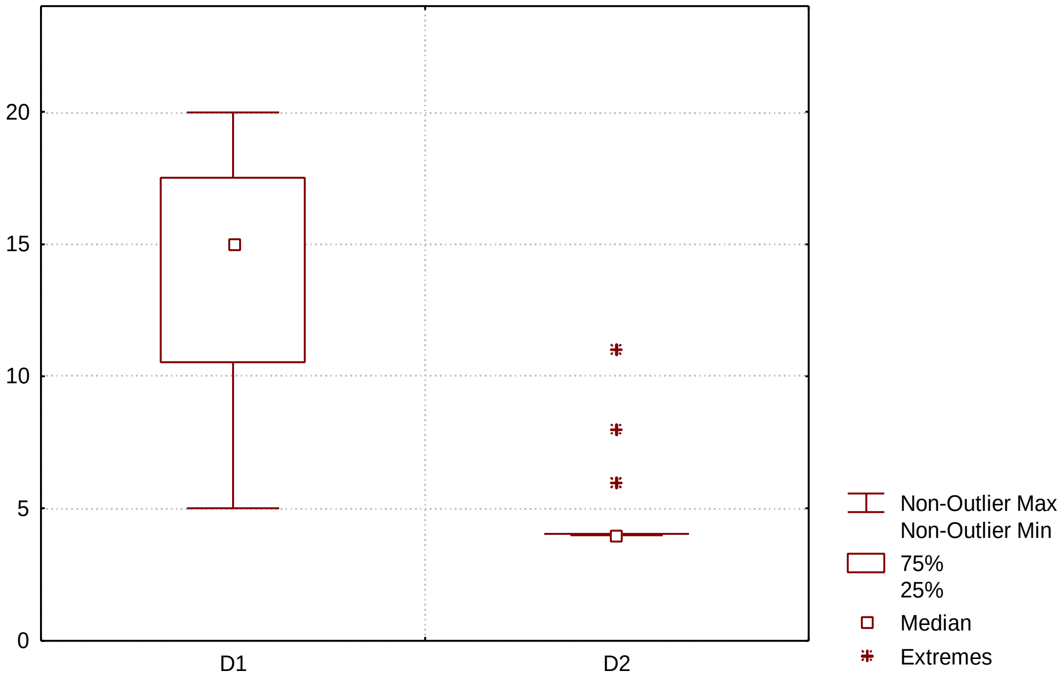

Figure 16: Box and whisker plot showing the variation in the measures of the two dimensions of Implement Software Design process capability. The first dimension, denoted D1, is "Process Implementation" and consists of levels 1 to 3. The second dimension, denoted D2, is "Quantitative Process Management" and consists of levels 4 and 5. The D2 measure used the same coding scheme as for the D1 measure, except that it consists of only 4 attributes. For the second dimension it is assumed that processes that were not rated would receive a rating of N (Not Achieved) if they would have been rated. This was done to ensure that the sample size for both dimensions was the same. Note that it is common practice not to rate the higher levels if there is strong *a priori* belief that the ratings will be N. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

In Figure 16 we can see the variation in the two measures of process capability. For D2 (Quantitative Process Management) there is little variation. The median of D2 is 4, which is the minimal value, indicating that at least 50% of projects do not implement any quantitative process management practices for their Implement Software Design process. On D1 there is substantially more variation with a median value of 15.

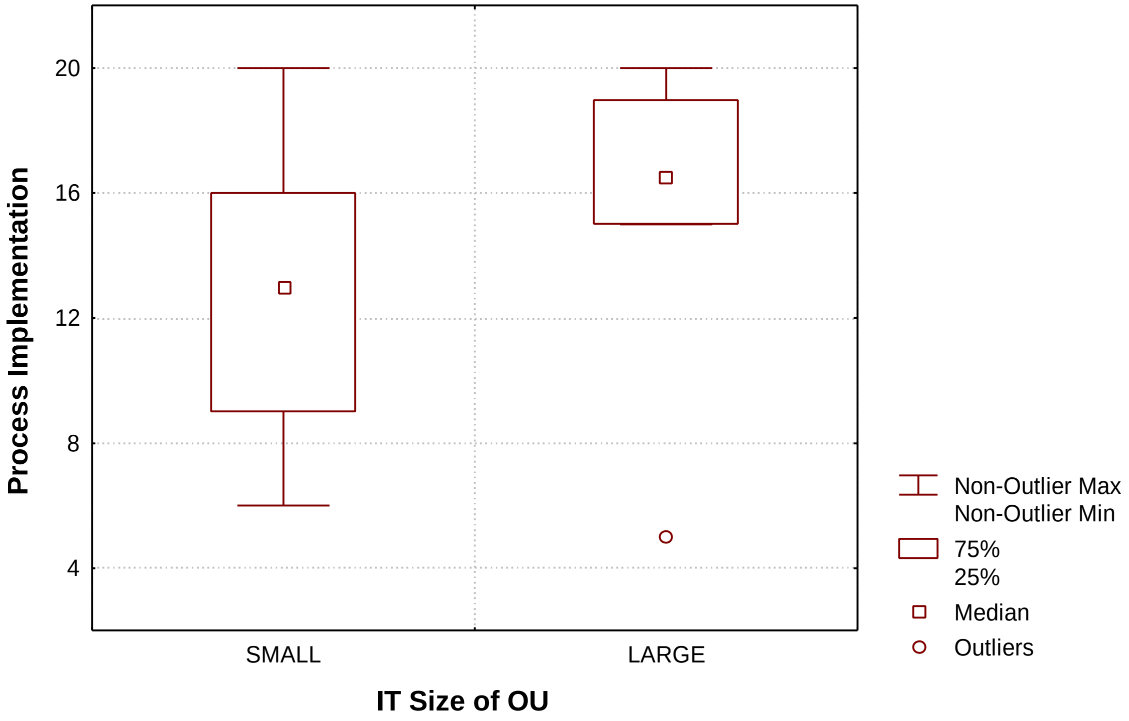

Figure 17: Box and whisker plot showing the variation in the "Process Implementation" capability dimension of the Implement Software Design process for the different OU sizes that were considered in our study (in terms of IT staff). A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

Figure 17 shows the variation along the D1 dimension for the two OU sizes that were considered during our study. Larger OUs tend to have greater implementation of Implement Software Design processes, and in this case the differences would seem to be marked.

5.1.3 Integrate and Test Software

Figure 18 shows the primary business sector distribution for those 25 organizations that assessed the Integrate and Test Software process. The most frequently occurring categories are the same as those for the whole of the trials. However, two categories disappeared altogether: Aerospace and Health and Pharmaceutical.

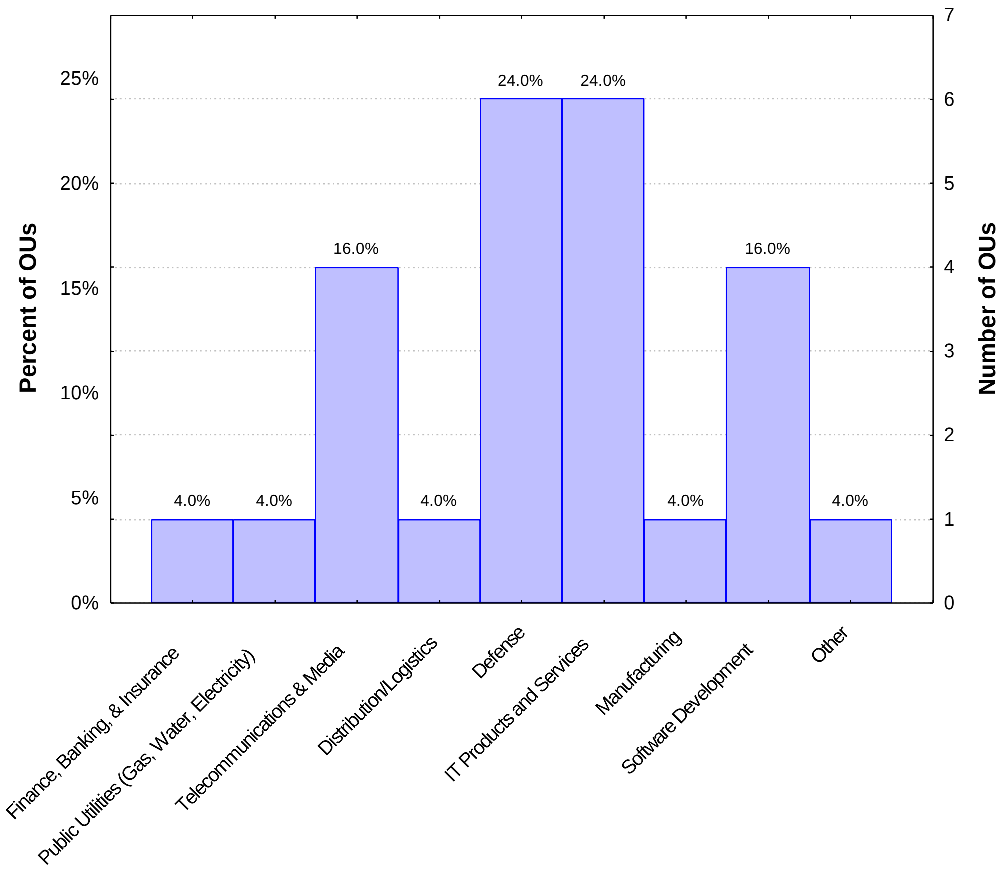

Figure 18: Business sector of all OUs that assessed the Integrate and Test Software process.

Of the 25 OUs, they were distributed by country as follows: Australia (10), Canada (1), Italy (1), France (2), Spain (3), Turkey (1), South Africa (4), Germany (1), Japan (2). Of these 12 were not ISO 9001 registered, and 13 were ISO 9001 registered.

Figure 19 shows the variation in the number of projects that were assessed in each OU. The median value is one project per OU, although in one case an OU assessed the Integrate and Test Software process in four different projects.

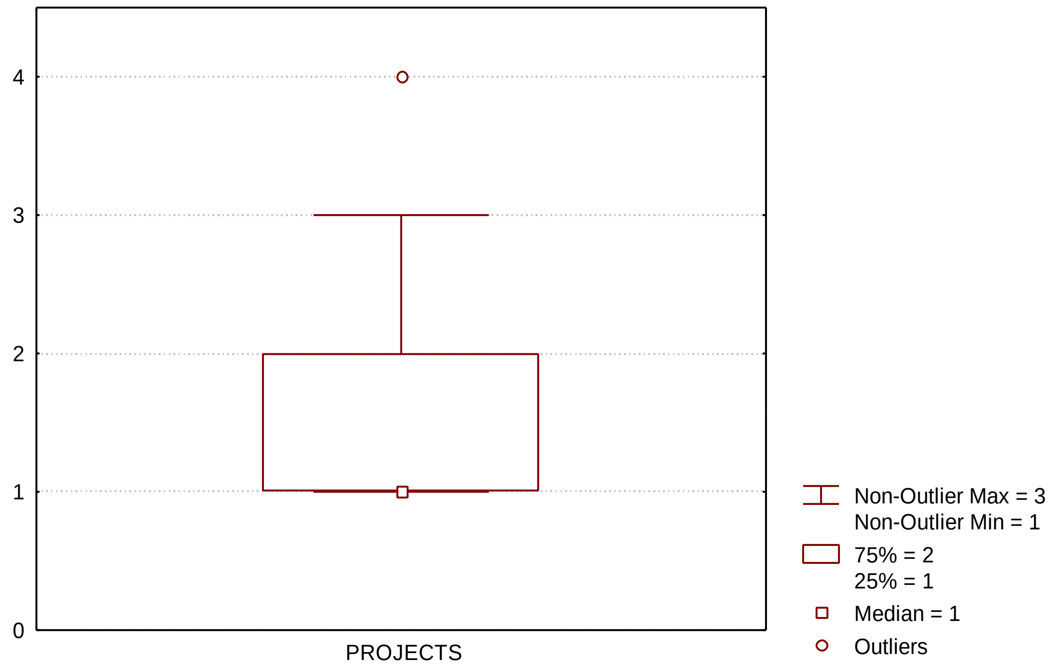

Figure 19: Box and whisker plot showing the variation in the number of projects that assessed their Integrate and Test Software process in each OU. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

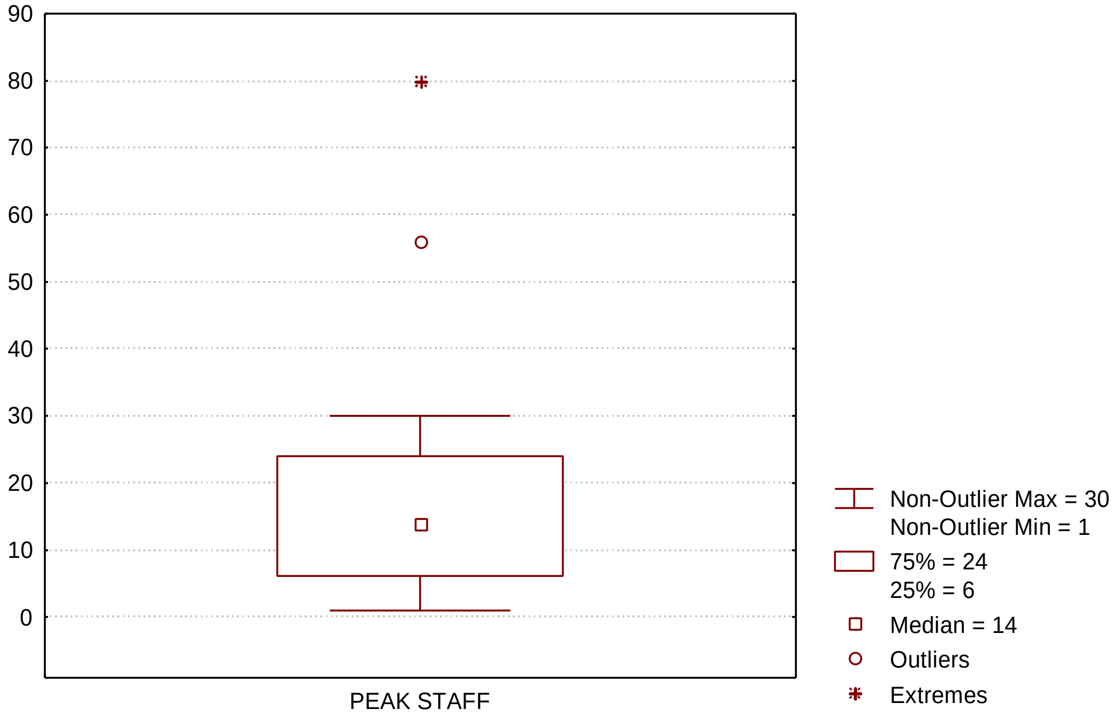

Figure 20: Box and whisker plot showing the variation in the peak staff load for projects that assessed the Integrate and Test Software process. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

The distribution of peak staff load for the projects that assessed the Integrate and Test Software process is shown in Figure 20. The median value is 14, with the smallest being a one person project and a non-outlier maximum of 30 staff. Although, one project had a peak load of 80 staff. It will be noted that the projects that assessed this process tend to be slightly smaller than the projects that assessed the other two development processes.

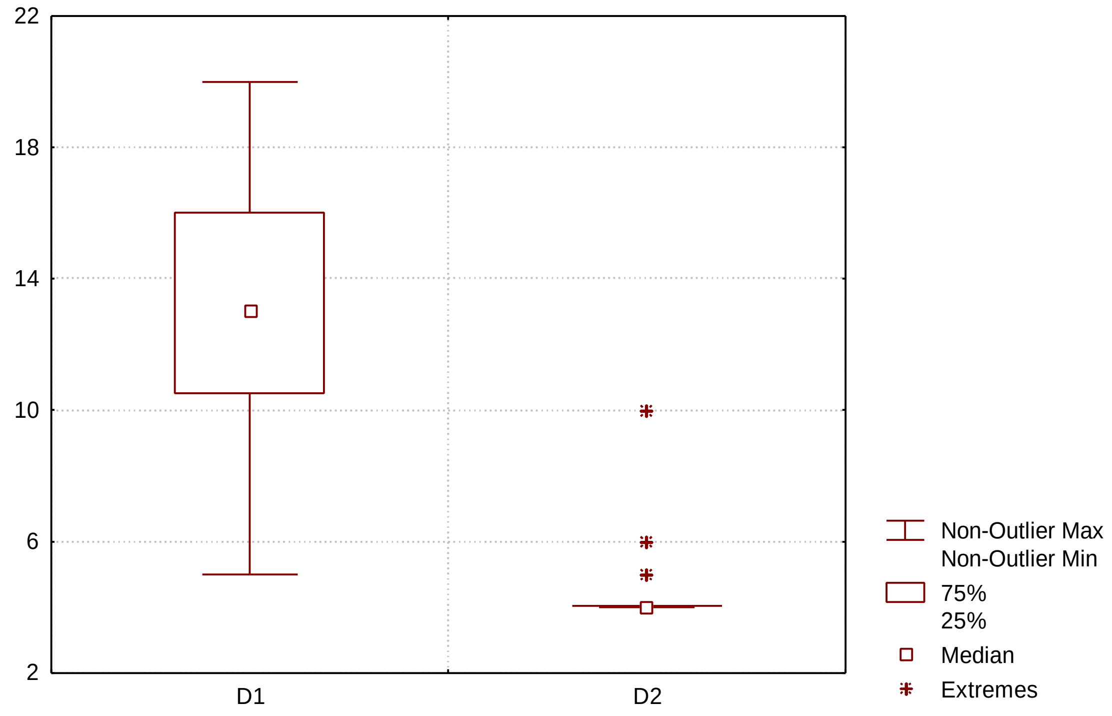

Figure 21: Box and whisker plot showing the variation in the measures of the two dimensions of Integrate and Test Software process capability. The first dimension, denoted D1, is “Process Implementation” and consists of levels 1 to 3. The second dimension, denoted D2, is “Quantitative Process Management” and consists of levels 4 and 5. The D2 measure used the same coding scheme as for the D1 measure, except that it consists of only 4 attributes. For the second dimension it is assumed that processes that were not rated would receive a rating of N (Not Achieved) if they would have been rated. This was done to ensure that the sample size for both dimensions was the same. Note that it is common practice not to rate the higher levels if there is strong *a priori* belief that the ratings will be N. A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

In Figure 21 we can see the variation in the two measures of process capability. For D2 (Quantitative Process Management) there is little variation. The median of D2 is 4, which is the minimal value, indicating that at least 50% of projects do not implement any quantitative process management practices for their Integrate and Test Software process. On D1 there is substantially more variation with a median value of 13.

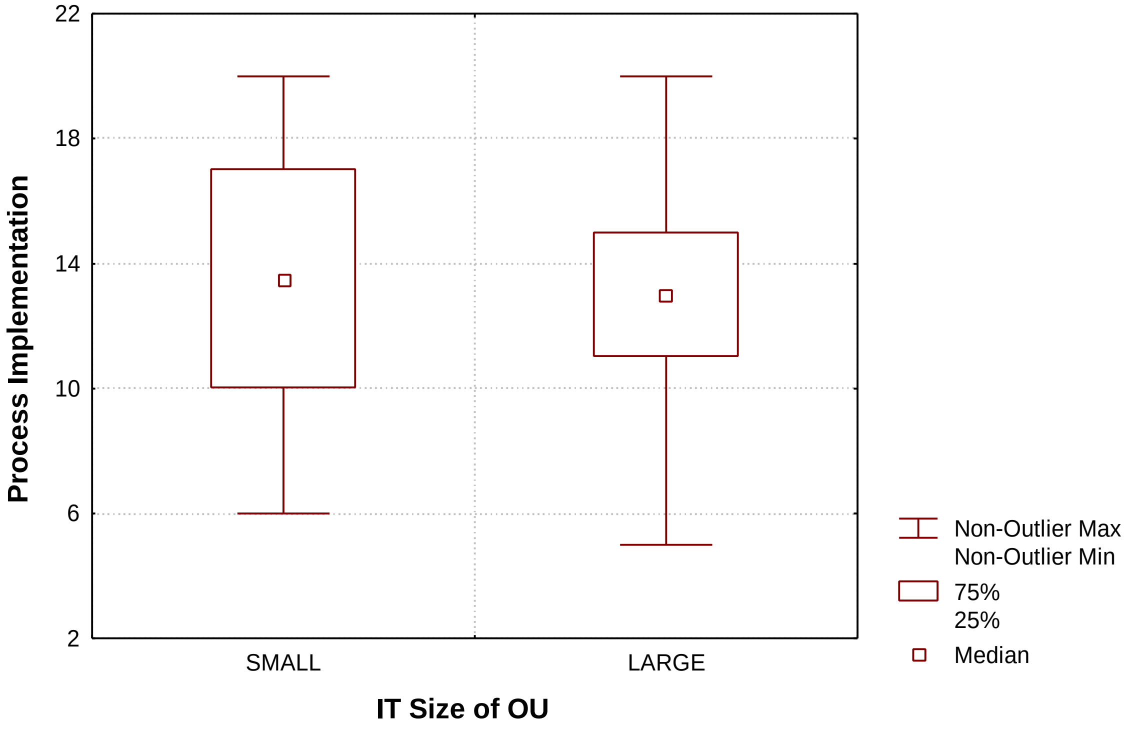

Figure 22: Box and whisker plot showing the variation in the "Process Implementation" capability dimension of the Integrate and Test Software process for the different OU sizes that were considered in our study (in terms of IT staff). A description of box and whisker plots and how to interpret them is provided in Appendix B (Section 8).

Figure 22 shows the variation along the D1 dimension for the two OU sizes that were considered during our study. Larger OUs tend to have slightly less implementation of Integrate and Test Software processes, although the difference is not marked.

5.1.4 Summary

To summarize the description of projects and assessments, we note the following:

- No organizations in the "Software Development" business sector assessed the Implement Software Design process when compared with all trials participants.
- For the Develop Software Design and Integrate and Test Software processes, the most frequently occurring types of organizations compared to the trials overall were the same.
- Organizations in the "Health and Pharmaceutical" business sector never assessed any of the development processes, even though some participated in the trials.
- For all three development processes, the median number of projects assessed per organization is one.
- In general, the larger projects tended to assess the Implement Software Design process more often.

- There was consistently little variation in the second dimension of process capability (Quantitative Process Management) for the three processes that were assessed, and at least 50% of assessed projects had no capability on that dimension.

- The capability on the Process Implementation dimension of process capability tended to be highest for the Implement Software Design process, and lowest for the Integrate and Test Software process.

- The differences in the Process Implementation dimension between large and small organizations tended not to be marked for the Develop Software Design and the Integrate and Test Software processes, but larger organizations tended to have higher capability on the Implement Software Design process.

- Variation in the Process Implementation dimension tended to be markedly smaller for larger organizations on the Implement Software Design and the Integrate and Test Software processes.

5.2 Reliability of the Software Development Process Capability Measures

The Cronbach alpha reliability coefficients for the "Process Implementation" variable are as shown in Table 7. For the purposes of our study, these values can be considered sufficiently large (see the interpretation guidelines in Section 4.5.5).

Process	**Cronbach alpha**
Develop Software Design	0.86
Implement Software Design	0.88
Integrate and Test Software	0.87

Table 7: Results of reliability evaluation for the process capability measure for the three development processes.

5.3 Affects of Software Development Process Capability

Below we provide the results for small organizations and large organizations for each of the three development processes. The results tables show the a_1 coefficient and its standard error for each imputed complete data set. The combined results include the average correlation coefficient across the complete data sets ($\bar{r}$), and the average a_1 coefficient ($\overline{Q}$) and its multiply imputed standard error $\sqrt{T}$. Values of $\bar{r}$ that are bolded indicate that it is larger than our 0.3 threshold. Values of $\overline{Q}$ that are bolded indicates that they are statistically significant at the Bonferonni adjusted alpha level.

5.3.1 Develop Software Design

Table 7 shows the results for small organizations. Here we see that the correlation between the Develop Software Design process and SCHEDULE variable is larger than our threshold, and the regression parameter is statistically significant. This indicates that higher capability increases the predictability of the project schedule and hence the ability of the project to meet their schedule commitments. However, no relationship was found with the BUDGET variable ($\bar{r} = 0.0625$), nor any of the remaining performance measures. This can be interpreted as an indicant that small organizations may be putting extra resources (budget) to meet their schedule commitments. The lack of other relationships may be due to a weak relationship between the Process Implementation dimension of process capability and the other performance measures in small organizations, perhaps due to a lack of applicability of the capability measure to small organizations.

Table 8 shows the results for larger organizations. Here we find strong relationships for all performance measures, and all except PRODUCTIVITY are statistically significant. The exception can be interpreted as a reflection of the fact that process capability does not necessarily imply a "good" design, only that the design process is implemented and managed. A design that has flaws may lead to rework and hence to reduced productivity. Another explanation is that the design process does not commonly consume a large proportion of a project's effort. Hence, even if efficiencies were realized during the process, they may not have a substantial impact on overall project productivity.

	Results from Repeated Imputations										**Combined Results**		
	Imputation 1		**Imputation 2**		**Imputation 3**		**Imputation 4**		**Imputation 5**				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	-0.0134	0.0501	0.0235	0.0466	0.0605	0.0488	0.0050	0.0487	-0.0135	0.0501	0.0625	0.0124	0.0595
SCHEDULE	0.0831	0.0508	0.0646	0.0545	0.1200	0.0450	0.0831	0.0508	0.1570	0.0449	**0.4507**	**0.1016**	0.0638
CUSTOMER	-0.0097	0.0401	0.0161	0.0346	0.0088	0.0374	-0.0023	0.0374	0.0161	0.0346	0.0430	0.0058	0.0390
REQUIREMENTS	0.0659	0.0408	0.0160	0.0291	0.0344	0.0271	0.0160	0.0291	0.0215	0.0344	0.2206	0.0307	0.0398
PRODUCTIVITY	0.0115	0.0187	-0.007	0.0224	0.0115	0.0187	-0.007	0.0224	-0.007	0.0224	0.0139	0.0004	0.0237
MORALE	0.0093	0.0320	0.0278	0.0329	0.0093	0.0378	0.0278	0.0329	0.0093	0.0320	0.1240	0.0167	0.0354

Table 8: Repeated imputation results and combined results for *small* organizations: *Develop Software Design* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the ith imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the ith imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. A bolded $\bar{r}$ value indicates that it is larger than our threshold of 0.3. A bolded $\overline{Q}$ value indicates that the slope difference from zero is statistically significant. The results indicate that the correlation between process capability for this process and meeting schedule targets is meaningfully large and statistically significant. This means that projects in small organizations that have a larger "Process Implementation" value on the Develop Software Design process tend to be better able to meet their schedule targets.

	Results from Repeated Imputations										**Combined Results**		
	Imputation 1		**Imputation 2**		**Imputation 3**		**Imputation 4**		**Imputation 5**				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	0.0782	0.0280	0.0990	0.0252	0.0676	0.0292	0.0617	0.0298	0.0812	0.0277	**0.4827**	**0.0775**	0.0321
SCHEDULE	0.1551	0.0332	0.1551	0.0332	0.1551	0.0332	0.1551	0.0332	0.1551	0.0332	**0.6822**	**0.1551**	0.0332
CUSTOMER	0.1139	0.0314	0.0898	0.0321	0.0657	0.0316	0.0792	0.0319	0.0838	0.0328	**0.4718**	**0.0865**	0.0374
REQUIREMENTS	0.1201	0.0409	0.0964	0.0375	0.1056	0.0396	0.1294	0.0427	0.1653	0.0369	**0.5239**	**0.1234**	0.0492
PRODUCTIVITY	0.0660	0.0444	0.0736	0.0457	0.0660	0.0444	0.0766	0.0455	0.0977	0.0470	**0.3157**	0.0760	0.0476
MORALE	0.1693	0.0492	0.1488	0.0492	0.1591	0.0493	0.1841	0.0470	0.1515	0.0514	**0.5503**	**0.1626**	0.0517

Table 9: Repeated imputation results and combined results for *large* organizations: *Develop Software Design* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the ith imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the ith imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. A bolded $\bar{r}$ value indicates that it is larger than our threshold of 0.3. A bolded $\overline{Q}$ value indicates that the slope difference from zero is statistically significant. The results indicate that projects in large organizations tend to have improved performance on their ability to meet budget and target commitments, ability to achieve customer satisfaction, ability to satisfy specified requirements, and improve staff morale and job satisfaction as they increase the "Process Implementation" capability of their Develop Software Design process. This evidence is actually quite strong given that the $\bar{r}$ values are relatively large, and that the slope parameter is statistically significant despite the conservative analysis approach we used (see Section 5.4 for a further discussion of this conservatism).

5.3.2 Implement Software Design

Table 10 shows the results for small organizations. All correlations were below our threshold of 0.3, and hence no relationships with project performance were identified.

Table 11 shows the results for large organizations. Here, all correlations except with PRODUCTIVITY are large according to our criterion. However, only the relationship with BUDGET is statistically significant. The lack of statistical significance may also be influenced by the small sample size in this particular sub-group (with only 14 projects).

	Results from Repeated Imputations										Combined Results		
	Imputation 1		Imputation 2		Imputation 3		Imputation 4		Imputation 5				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	-0.0165	0.0550	-0.0511	0.0536	-0.0142	0.0524	-0.0315	0.0545	-0.0165	0.0550	-0.1183	-0.0259	0.0568
SCHEDULE	-0.0113	0.0612	0.0430	0.0586	0.0430	0.0586	0.0234	0.0606	0.0430	0.0586	0.1182	0.0282	0.0649
CUSTOMER	-0.0632	0.0357	-0.0196	0.0365	-0.0294	0.0316	-0.0459	0.0350	-0.0129	0.0345	-0.2338	-0.0342	0.0413
REQUIREMENTS	-0.0259	0.0316	-0.0357	0.0312	-0.0357	0.0312	-0.0184	0.0320	-0.0357	0.0312	-0.2331	-0.0302	0.0326
PRODUCTIVITY	-0.0144	0.0389	-0.0083	0.0368	0.0127	0.0402	0.0052	0.0403	0.0127	0.0402	0.0082	0.0016	0.0416
MORALE	0.0077	0.0422	-0.0096	0.0408	-0.0465	0.0443	-0.0096	0.0408	0.0077	0.0422	-0.056	-0.0101	0.0485

Table 10: Repeated imputation results and combined results for *small* organizations: *Implement Software Design* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the i^th^ imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the i^th^ imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. No evidence was found for a relationship between the "Process Implementation" capability on the Implement Software Design process and any of the six performance measures.

	Results from Repeated Imputations										Combined Results		
	Imputation 1		Imputation 2		Imputation 3		Imputation 4		Imputation 5				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	0.1467	0.0245	0.1467	0.0245	0.1467	0.0245	0.1535	0.0215	0.1535	0.0215	**0.8793**	**0.1495**	0.0237
SCHEDULE	0.1006	0.0579	0.0870	0.0505	0.0734	0.0384	0.0870	0.0505	0.0870	0.0505	**0.4534**	0.0870	0.0511
CUSTOMER	0.0666	0.0464	0.0529	0.0574	0.0597	0.0525	0.0666	0.0464	0.0597	0.0525	**0.3293**	0.0611	0.0516
REQUIREMENTS	0.0734	0.0384	0.0734	0.0384	0.0870	0.0505	0.0870	0.0505	0.0734	0.0384	**0.4679**	0.0788	0.0444
PRODUCTIVITY	0.0287	0.0558	0.0219	0.045	0.0219	0.0459	0.0355	0.0563	0.0355	0.0563	0.1558	0.0287	0.0528
MORALE	0.1293	0.0614	0.1293	0.0614	0.1225	0.0576	0.1293	0.0614	0.1293	0.0614	**0.5201**	0.1280	0.0608

Table 11: Repeated imputation results and combined results for *large* organizations: *Implement Software Design* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the ith imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the ith imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. A bolded $\bar{r}$ value indicates that it is larger than our threshold of 0.3. A bolded $\overline{Q}$ value indicates that the slope difference from zero is statistically significant. The results indicate that there is a "clinically significant" relationship between the "Process Implementation" capability of the Implement Software Design process and five of the six performance measures, but only the relationship with the ability to meet budget commitments is statistically significant.

5.3.3 Integrate and Test Software

Table 12 shows the results for small organizations. No relationships were found between process capability and any of the performance measures.

Table 13 shows the results for large organizations. Only the relationship with the PRODUCTIVITY variable was large and statistically significant, indicating that improvements in the capability of the Integrate and Test Software process can increase productivity. Intuitively this makes sense as integration and testing can commonly consume large proportions of a project's overall resources. Therefore any improvements in its efficiency can lead to substantial improvements in productivity.

	Results from Repeated Imputations										Combined Results		
	Imputation 1		Imputation 2		Imputation 3		Imputation 4		Imputation 5				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	-0.0218	0.0495	-0.0008	0.0468	0.0389	0.0501	-0.0218	0.0495	-0.0008	0.0468	-0.0075	-0.0013	0.0556
SCHEDULE	-0.0092	0.0565	-0.0034	0.0546	0.0483	0.0602	0.0213	0.0563	0.0025	0.0558	0.0492	0.0119	0.0622
CUSTOMER	0.0025	0.0291	-0.0034	0.0330	-0.0069	0.0325	0.0025	0.0291	0.0025	0.0291	-0.0027	-0.0005	0.0310
REQUIREMENTS	0.0049	0.0284	-0.0104	0.0288	-0.0040	0.0434	0.0049	0.0284	-0.0040	0.0434	-0.0101	-0.0017	0.036
PRODUCTIVITY	-0.0188	0.0238	-0.0035	0.0217	-0.0282	0.0266	-0.0188	0.0238	-0.0188	0.0238	-0.1758	-0.0176	0.0259
MORALE	-0.0297	0.0342	-0.0146	0.0348	-0.0205	0.0363	-0.0205	0.0363	-0.0483	0.0369	-0.1813	-0.0267	0.0386

Table 12: Repeated imputation results and combined results for *small* organizations: *Integrate and Test Software* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the ith imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the ith imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. No evidence was found for a relationship between the "Process Implementation" capability on the Integrate and Test Software process and any of the six performance measures.

	Results from Repeated Imputations										Combined Results		
	Imputation 1		Imputation 2		Imputation 3		Imputation 4		Imputation 5				
	$\hat{Q}_1$	$\sqrt{U_1}$	$\hat{Q}_2$	$\sqrt{U_2}$	$\hat{Q}_3$	$\sqrt{U_3}$	$\hat{Q}_4$	$\sqrt{U_4}$	$\hat{Q}_5$	$\sqrt{U_5}$	$\bar{r}$	$\overline{Q}$	$\sqrt{T}$
BUDGET	-0.0228	0.0503	0.0743	0.0520	0.0569	0.0573	0.0457	0.0572	0.0847	0.0551	0.2039	0.0477	0.0715
SCHEDULE	-0.0538	0.0453	-0.0391	0.0511	-0.0592	0.0404	-0.0058	0.0466	0.0085	0.0516	-0.1612	-0.0299	0.0574
CUSTOMER	-0.0379	0.0494	-0.0611	0.0526	0.0255	0.0623	0.0012	0.0552	0.0232	0.0587	-0.0524	-0.0098	0.0699
REQUIREMENTS	0.0402	0.0587	0.0128	0.0561	-0.0410	0.0599	0.0348	0.0514	0.0782	0.0591	0.1076	0.025	0.0746
PRODUCTIVITY	0.1374	0.0486	0.1401	0.0528	0.0557	0.0457	0.1548	0.0498	0.1060	0.0490	**0.5020**	**0.1188**	0.0655
MORALE	-0.0062	0.0388	-0.0491	0.0374	-0.0375	0.0431	-0.0638	0.0399	-0.0151	0.0477	-0.2030	-0.0344	0.0490

Table 13: Repeated imputation results and combined results for *large* organizations: *Integrate and Test Software* process. The $\hat{Q}_i$ values are the estimated slope parameters of the regression model for the ith imputed data set. The $\sqrt{U_i}$ values are the standard errors of the estimated slope parameter for the ith imputed data set. The combined results consist of the $\bar{r}$ value, which is the mean correlation coefficient across the five imputed data sets, the $\overline{Q}$ value, which is the combined slope parameter across the five imputed data sets, and the $\sqrt{T}$ value, which is the combined standard error of $\overline{Q}$. Section 7.8 describes how values can be combined across imputed data sets. A bolded $\bar{r}$ value indicates that it is larger than our threshold of 0.3. A bolded $\overline{Q}$ value indicates that the slope difference from zero is statistically significant. The results indicate that there is a relationship between the "Process Implementation" capability of the Integrate and Test process and productivity.

5.3.4 Summary and Discussion

An overall summary of the results from this study is provided in Table 14. This table shows in a succinct manner which processes were found to be associated with each of the performance measures for small and large organizations. Due to the conservatism that is noted below, we also include in that summary processes that had an association with performance measures whose magnitude was greater than 0.3, but that was not statistically significant.

A number conclusions can be drawn:

- We found weak evidence supporting the verisimilitude of the predictive validity premise for small organizations. This may be an indicatent that the process capability measure is not appropriate for small organizations, or that the capabilities stipulated in ISO/IEC 15504 do not necessarily improve project performance in small organizations.
- The productivity of projects in large organizations is associated with the capability of the integration and testing process. Such a relationship makes intuitive sense since this process commonly consumes large proportions of project effort.

- The association of the Develop and Implement Software Design processes and the remaining performance measures in large organizations have relatively large magnitudes, although statistical significance is only attained for the Develop Software Design process. For the Implement Software Design process the sample size within that subset may have been too small (hence, low statistical power).
- Given these results, we can confidently remark that the Develop Software Design process is a key one for large organizations, and its assessment and improvement can provide substantial payoff.

5.4 Limitations

Two potential limitations of our results concern their conservatism and generalizability.

Our results can be considered conservative due to the Bonferroni procedure that we employ for statistical significance testing. Another factor leading to conservatism is that our performance measures used single item scales. Single item scales are typically less reliable than multiple item scales. Nunnally (Nunnally, 1978) notes that a correction for attenuation in the correlation coefficient that considers the observed reliability of the performance variables will estimate the “real validity” of the predictor. In our study, we could not estimate the reliability of our criterion. Therefore our predictive validity coefficients are likely smaller than they would be with multi-item performance measures with a correction for attenuation. This conservatism means that when we identify a meaningfully large (“clinically significant”) and statistically significant result then the evidence is quite substantial since a significant result is found despite the conservatism. However, it also means that subsequent studies (say with larger sample sizes and more reliable criterion measures) may find more of the processes related to performance.

There may also be limitations on the generalizability of our results. Specifically, the extent to which our findings can be generalized to assessments that are not based on the emerging ISO/IEC 15504 International Standard. The emerging ISO/IEC 15504 International Standard defines requirements on assessments. Assessments that satisfy the requirements are claimed to be compliant. Based on public statements that have been made thus far, it is expected that some of the more popular assessment models and methods will be consistent with the emerging ISO/IEC 15504 International Standard. For example, Bootstrap version 3.0 claims compliance with ISO/IEC 15504 (Bicego et al., 1998), and the future CMMI product suite is expected to be consistent and compatible (Software Engineering Institute, 1998b). The assessments from which we obtained our data are also considered to be compliant. The extent to which our results, obtained from a subset of compliant assessments, can be generalized to all compliant assessments is an empirical question and can be investigated through replications of our study. The logic of replications leading to generalizable results is presented in (Lindsay and Ehrenberg, 1993).

Performance Measure	**Process(es)**
Small Organizations	
Ability to meet budget commitments	
Ability to meet schedule commitments	**Develop Software Design**
Ability to achieve customer satisfaction	
Ability to satisfy specified requirements	
Staff productivity	
Staff morale / job satisfaction	
Large Organizations	
Ability to meet budget commitments	**Develop Software Design** **Implement Software Design**
Ability to meet schedule commitments	**Develop Software Design** Implement Software Design
Ability to achieve customer satisfaction	**Develop Software Design** Implement Software Design
Ability to satisfy specified requirements	**Develop Software Design** Implement Software Design
Staff productivity	**Integrate and Test Software** Develop Software Design
Staff morale / job satisfaction	**Develop Software Design** Implement Software Design

Table 14: Summary of the findings from our predictive validity study. In the first column are the performance measures that were collected for each project. In the second column are the development processes whose capability was evaluated. The results are presented separately for small (equal to or less than 50 IT staff) and large organizations (more than 50 IT staff). For each performance measure we show the software development processes that were found to be related to it. A process was considered to be associated with a performance measure if it had a correlation coefficient that was greater than or equal to 0.3 (i.e., “clinically significant”), and that was statistically significant at a one-tailed (Bonferonni adjusted) alpha level of 0.1. These processes are shown in bold. The processes that are not bolded are only “clinically significant” but not statistically significant. The lack of statistical significance may be a consequence of small sample size.

6 Conclusions

In this paper we have presented an empirical study that evaluated the predictive validity of the ISO/IEC 15504 measures of software development process capability (i.e., Develop Software Design, Implement Software Design, and Integrate and Test Software). Predictive validity is a basic premise of all software process assessments that produce quantitative results. We first demonstrated that no previous studies have evaluated the predictive validity of these processes using the ISO/IEC 15504 measure, and then described our study in detail. Our results indicate that higher development process capability is related with increased project performance for large organizations. In particular, we found that the "Develop Software Design" process capability to be associated with five different project performance measures, indicating its importance as a target for process improvement. The "Integrate and Test Software" process capability was also found to be associated with productivity. However, the evidence of predictive validity for small organizations was rather weak.

The results suggest that improving the design and integration and testing processes may potentially lead to improvements in the performance of software projects in large organizations. It is by no means claimed that development process capability is the only factor that is associated with performance. Only that some relatively strong associations have been found during our study, suggesting that these processes ought to be considered as potential targets for assessment and improvement.

It is important to emphasize that studies such as this ought to be replicated to provide further confirmatory evidence as to the predictive validity of ISO/IEC 15504 development process capability. It is known in scientific pursuits that there exists a "file drawer problem" (Rosenthal, 1991). This problem occurs when there is a reluctance by journal editors to publish, and hence a reluctance by researchers to submit, research results that do not show statistically significant relationships. One can even claim that with the large vested interests in the software process assessment community, reports that do not demonstrate the efficacy of a particular approach or model may be buried and not submitted for publication. Therefore, published works are considered to be a biased sample of the predictive validity studies that are actually conducted. However, by combining the results from a large number of replications that show significant relationships, one can assess the number of studies showing no significant relationships that would have to be published before our overall conclusion of there being a significant relationship is put into doubt (Rosenthal, 1991). This assessment would allow the community to place realistic confidence (or otherwise) in the results of published predictive validity studies.

7 Appendix A: Multiple Imputation Method

In this appendix we describe the approach that we used for imputing missing values on the performance variable, and also how we operationalize it in our specific study. It should be noted that, to our knowledge, multiple imputation techniques have not been employed thus far in software engineering empirical research, where the common practice has been to ignore observations with missing values.

7.1 Notation

We first present some notation to facilitate explaining the imputation method. Let the raw data matrix have i rows (indexing the cases) and j columns (indexing the variables), where $i = 1 \ldots n$ and

$j = 1 \ldots q$. Some of the cells in this matrix may be unobserved (i.e., missing values). We assume that there is only one outcome variable of interest for imputation (this is also the context of our study since we deal with each dependent variable separately), and let y_i denote its value for the i^{th} case. Let $Y = (Y_{mis}, Y_{obs})$, where Y_{mis} denotes the missing values and Y_{obs} denotes the observed values on that variable. Furthermore, let X be a scaler or vector of covariates that are fully observed for every i. These may be background variables, which in our case were the size of an organization in IT staff and whether the organization was ISO 9001 registered, and other covariates that are related to the outcome variable, which in our case was the process capability measure (i.e., "process implementation" as defined in the main body of the text).

Let the parameter of interest in the study be denoted by Q. We assume that Q is scaler since this is congruent with our context. For example, let Q be a regression coefficient. We wish to estimate $\widehat{Q}$ with associated variance U from our sample.

7.2 Ignorable Models

Models underlying the method of imputation can be classified as assuming that the reasons for the missing data are either ignorable or nonignorable. Rubin (Rubin, 1987) defines this formally. However, here it will suffice to convey the concepts, following (Rubin, 1988).

Ignorable reasons for the missing data imply that a nonrespondent is only randomly different from a respondent with the same value of X. Nonignorable reasons for missing data imply that, even though respondents and nonrespondents have the same value of X, there will be a systematic difference in their values of Y. An example of a nonignorable response mechanism in the context of process assessments that use a model such as that of ISO/IEC 15504 is when organizations assess a particular process because it is perceived to be weak and important for their business. In such a case, processes for which there are capability ratings are likely to have lower capability than other processes that are not assessed.

In general, most imputation methods assume ignorable nonresponse (Schaefer, 1997) (although, it is possible to perform, for example, multiple imputation, with a nonignorable nonresponse mechanism). In the analysis presented in this paper there is no a priori reason to suspect that respondents and nonrespondents will differ systematically in the values of the outcome variable, and therefore we assume ignorable nonresponse.

7.3 Overall Multiple Imputation Process

The overall multiple imputation process is shown in Figure 23. Each of these tasks is described below. It should be noted that the description of these tasks is done from a Bayesian perspective.

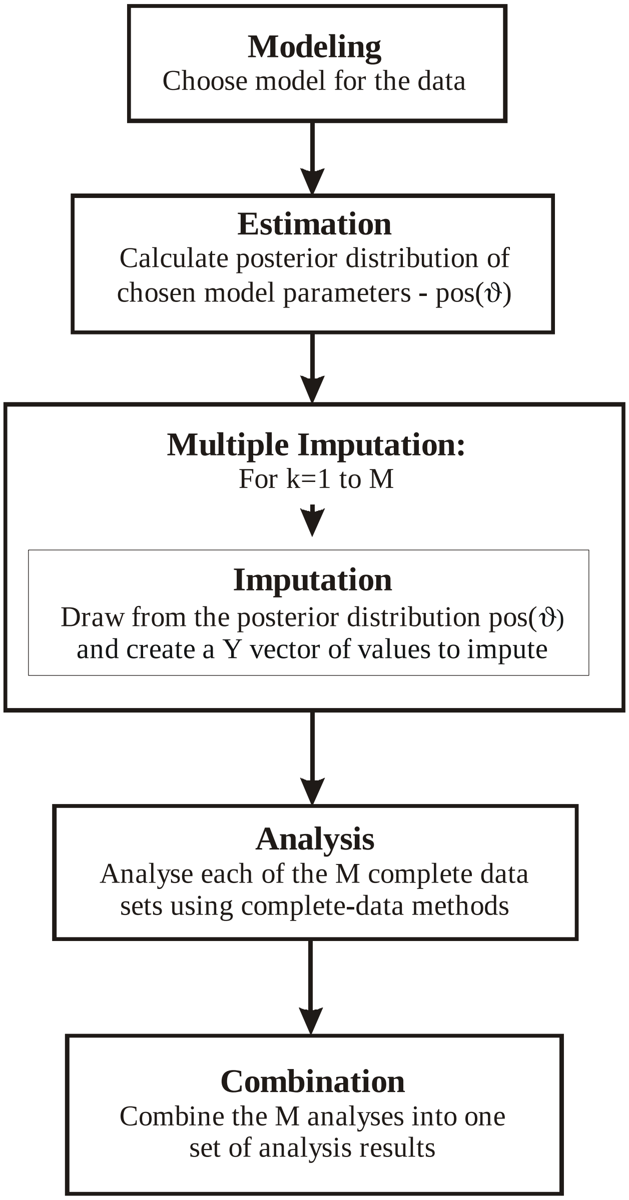
Modeling
Choose model for the data
Estimation
Calculate posterior distribution of
chosen model parameters - pos(ϑ)
Multiple Imputation:
For k=1 to M
Imputation
Draw from the posterior distribution pos(ϑ)
and create a Y vector of values to impute
Analysis
Analyse each of the M complete data
sets using complete-data methods
Combination
Combine the M analyses into one
set of analysis results

Figure 23: Schematic showing the tasks involved in multiple imputation.

7.4 Modelling Task

The objective of the modelling task is to specify a model $f_{Y|X}\left(Y_i \middle| X_i, q_{Y|X}\right)$ using the observed data only where $q_{Y|X}$ are the model parameters. For example, consider the situation where we define an ordinary least squares regression model that is constructed using the observed values of Y and the predictor variables are the covariates X, then $q_{Y|X} = (b, s^2)$ are the vector of the regression parameters and the variance of the error term respectively. This model is used to impute the missing values. In our case we used an implicit model that is based on the hot-deck method. This is described further below.

7.5 Estimation Task

We define the posterior distribution of q as $\Pr\left(q \mid X, Y_{obs}\right)$.[20] However, the only function of q that is needed for the imputation task is $q_{Y|X}$. Therefore, during the estimation task, we draw repeated values of $q_{Y|X}$ from its posterior distribution $\Pr\left(q_{Y|X} \mid X, Y_{obs}\right)$. Let's call a drawn value $q^{*}_{Y|X}$.

7.6 Imputation Task

The posterior predictive distribution of the missing data given the observed data is defined by the following result:

$$\Pr\left(Y_{mis} \mid X, Y_{obs}\right) = \int \Pr\left(Y_{mis} \mid X, Y_{obs}, q\right) \Pr\left(q \mid X, Y_{obs}\right) dq \qquad \textbf{Eqn. 1}$$

We therefore draw a value of Y_{mis} from its conditional posterior distribution given $q^{*}_{Y|X}$. For example, we can draw $q^{*}_{Y|X} = (b^{*}, s^{*2})$ and compute the missing y_i from $f\left(y_i \middle| x_i, q^{*}_{Y|X}\right)$. This is the value that is imputed. This process is repeated M times.

7.7 Analysis Task

For each of the M complete data sets, we can calculate the value of Q. This provides us with the complete-data posterior distribution of Q: $\Pr\left(Q \mid X, Y_{obs}, Y_{mis}\right)$.

7.8 Combination Task

The basic result provided by Rubin (Rubin, 1987) is:

$$\Pr\left(Q \mid X, Y_{obs}\right) = \int \Pr\left(Q \mid X, Y_{obs}, Y_{mis}\right) \Pr\left(Y_{mis} \mid X, Y_{obs}\right) dY_{mis} \qquad \textbf{Eqn. 2}$$

This result states that the actual posterior distribution of Q is equal to the average over the repeated imputations. Based on this result, a number of inferential procedures are defined.

The repeated imputation estimate of Q is:

$$\overline{Q} = \sum \frac{\widehat{Q}_m}{M} \qquad \textbf{Eqn. 3}$$

which is the mean value across the M analyses that are performed.

The variability associated with this estimate has two components. First there is the within-imputation variance:

[20] We use the notation $\Pr(\cdot)$ to denote a probability density.

$$\overline{U} = \sum \frac{U_m}{M} \qquad \textbf{Eqn. 4}$$

and second the between imputation variance:

$$B = \frac{\sum \left(\hat{Q}_m - \overline{Q}\right)^2}{M-1} \qquad \textbf{Eqn. 5}$$

The total variability associated with $\overline{Q}$ is therefore:

$$T = \overline{U} + \left(1 + M^{-1}\right)B \qquad \textbf{Eqn. 6}$$

In the case where Q is scaler, the following approximation can be made:

$$\frac{\left(Q - \overline{Q}\right)}{\sqrt{T}} \sim t_v \qquad \textbf{Eqn. 7}$$

where t_v is a t distribution with v degrees of freedom where:

$$v = (M-1)\left(1 + r^{-1}\right)^2 \qquad \textbf{Eqn. 8}$$

and

$$r = \frac{\left(1 + M^{-1}\right)B}{\overline{U}} \qquad \textbf{Eqn. 9}$$

If one wants to test the null hypothesis that $H_0 : Q = 0$ then the following value can be referred to a t distribution with v degrees of freedom:

$$\frac{\overline{Q}}{\sqrt{T}} \qquad \textbf{Eqn. 10}$$

7.9 Hot-Deck Imputation: Overview

We will first start by presenting the hot-deck imputation procedure in general, then show the particular form of the procedure that we use in our analysis, and how this is incorporated into the multiple imputation process presented above.

Hot-deck procedures are used to impute missing values. They are a duplication approach whereby a *recipient* with a missing value receives a value from a *donor* with an observed value (Ford, 1983). Therefore the donor's value is duplicated for each recipient. As can be imagined, this procedure can be operationalized in a number of different ways.

A basic approach for operationalizing this is to sample from the n_{obs} observed values and use these to impute the n_{mis} missing values (Little and Rubin, 1987), where $n = n_{mis} + n_{obs}$. A simple sampling scheme could follow a multinomial model with sample size n_{mis} and probabilities $\left({}^{1}\!/_{n_{obs}}, \ldots, {}^{1}\!/_{n_{obs}} \right)$. It is more common, however, to use the X covariates to perform a poststratification. In such a case, the covariates are used to construct C disjoint classes of observations such that the observations within each class are as homogeneous as possible. This also has the advantage of further reducing nonresponse bias.

For example, if X consists of two binary vectors, then we have 4 possible disjoint classes. Within each class there will be some observations with Y observed and some with Y missing. For each of the missing values, we can randomly select an observed Y value and use it for imputation. This may result in the same observation serving as a donor more than once (Sande, 1983). Here it is assumed that within each class the respondents follow the same distribution as the nonrespondents.

7.10 Metric-Matching Hot-Deck

It is not necessary that the X covariates are categorical. They can be continuous or a mixture of continuous and categorical variables. In such a case a distance function is defined, and the l nearest observations with the Y value observed serve as the donor pool (Sande, 1983).

An allied area where such metric-matching has received attention is the construction of matched samples in observational studies (Rosenbaum and Rubin, 1985). This is particularly relevant to our case because we cannot ensure in general that all the covariates that will be used in all analyses will be categorical. For the sake of brevity, we will only focus on the particular metric-matching technique that we employ.

7.11 Response Propensity Matching

In many observational studies[21] (see (Cochran, 1983)) a relatively small group of subjects is exposed to a treatment, and there exists a larger group of unexposed subjects. Matching is then performed to identify unexposed subjects who serve as a control group. This is done to ensure that the treatment and control groups are both similar on background variables measured on all subjects.

Let the variable R_i denote whether a subject i was exposed ($R_i = 1$) or unexposed ($R_i = 0$) to the treatment. Define the propensity score, $e(X)$ as the conditional probability of exposure given the covariates (i.e., $e(X) = \Pr(R = 1 \mid X)$). Rosenbaum and Rubin (Rosenbaum and Rubin, 1983) prove some properties of the propensity score that are relevant for us.

First, they show that the distribution of X is the same for all exposed and unexposed subjects within strata with constant values of $e(X)$. Exact matching will therefore tend to balance the X distributions

[21] These are studies where there is not a random assignment of subjects to treatments. For example, in the case of studying the relationship between exposure to cigarette smoke and cancer, it is not possible to deliberately expose some subjects to smoke.

for both groups. Furthermore, they also show that the distribution of the outcome variable Y is the same for exposed and unexposed subjects with the same value of $e(X)$ (or within strata of constant $e(X)$).

David et al. (David et al., 1983) adopt these results to the context of dealing with nonresponse in surveys. We can extrapolate and let $R_i = 1$ indicate that there was a response on Y for observation i, and that $R_i = 0$ indicates nonresponse. Hence we are dealing with response propensity as opposed to exposure propensity. We shall denote response propensity with $p(X)$. It then follows that under ignorable nonresponse if we can define strata with constant $p(X)$ then the distribution of X and Y are the same for both respondents and nonrespondents within each stratum.

To operationalize this, we need to address two issues. First, we need to estimate $p(X)$. Second, it is unlikely that we would be able to define sufficiently large strata where $p(X)$ is constant, and therefore we need to approximate this.

If we take the response indicator R to be a Bernoulli random variable independently distributed across observations, then we can define a logistic regression model (Hosmer and Lemeshow, 1989):

$$p(X) = \frac{e^{\left(a_0 + a_1 X_1 + \cdots + a_{q-1} X_{q-1}\right)}}{1 + e^{\left(a_0 + a_1 X_1 + \cdots + a_{q-1} X_{q-1}\right)}}$$

This will provide us with an estimate of response propensity for respondents and nonrespondents.

We can then group the estimated response propensity into C intervals, with bounding values $0, p_1, p_2, \ldots, p_{C-1}, 1$. Strata can then be formed with observation i in stratum c if $p_{c-1} < p_i < p_c$ with $c = 1 \ldots C$. Therefore, we have constructed strata with approximately constant values of response propensity. In our application we set $C = 5$, dividing the estimated response propensity score using quintiles.

7.12 An Improper Hot-Deck Imputation Method

Now that we have constructed homogeneous strata, we can operationalize the metric-matching hot-deck imputation procedure by sampling with equal probability from the respondents within each stratum, and use the drawn values to impute the nonrespondent values in the same stratum. However, doing so we do not draw q from its posterior distribution, and then draw Y_{mis} from its posterior conditional distribution given the drawn value of q. Such a procedure would be improper because it does not take into account the uncertainty introduced by the imputation itself. Therefore some alternatives are considered, namely the approximate Bayesian bootstrap.

7.13 The Approximate Bayesian Bootstrap

A proper imputation approach that has been proposed is the Approximate Bayesian Bootstrap – ABB – (Rubin and Schenker, 1986; Rubin and Schenker, 1991). This is an approximation of the Bayesian Bootstrap (Rubin, 1981) that is easier to implement. The procedure for the ABB is, for each stratum, to draw with replacement z_{obs} Y values, where z_{obs} is the number of observed Y values in the stratum. Then, draw from that z_{mis} Y values with replacement, where z_{mis} is the number of observations with missing values in the stratum. The latter draws are then used to impute the missing values within the stratum. The drawing of z_{mis} missing values from a possible sample of z_{obs} values rather than from the actual observed values generates the appropriate between-imputation variability. This is repeated M times to generate multiple imputations.

7.14 Summary

The procedure that we have described implements multiple-imputation through the hot-deck method. It consists of constructing a response propensity model followed by an Approximate Bayesian Bootstrap.

This procedure is general and can be applied to impute missing values that are continuous or categorical. We have described it here in the context of univariate Y, but it is generally applicable to multivariate Y (see (Rubin, 1987) for a detailed discussion of multiple-imputation for multivariate Y).

8 Appendix B: Understanding Box and Whisker Plots

In this paper, box and whisker plots are used quite frequently. This brief appendix is intended to explain how to interpret such a diagram.

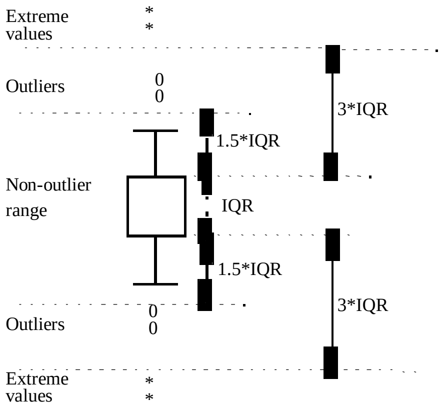

Figure 24: Description of a box and whisker plot.

Box and whisker plots are used to show the variation in a particular variable. Figure 24 shows how such a plot is constructed. The box represents the inter-quartile range (IQR). The IQR bounds the 25th and 75th percentiles. The 25th percentile is the value of the variable where 25% or less of the observations have equal or smaller values. The same is for the 75th percentile. The whiskers are the largest values within 1.5 times the size of the box. This value of 1.5 is conventional. Outliers are within 1.5 times the size of the box beyond the whiskers, and extremes are beyond the outliers. Finally, usually there is a dot in the box. This dot would be the median, or the 50th percentile.

The box and whisker plot provides a rather versatile way for visualizing the obtained values on a variable.

9 Acknowledgements

We wish to thank all the participants in the SPICE Trials, without whom this study would not have been possible. In particular, our gratitude is given to Inigo Garro, Angela Tuffley, Bruce Hodgen, Alastair Walker, Stuart Hope, Robin Hunter, Dennis Goldenson, Terry Rout, Steve Masters, Ogawa Kiyoshi, Victoria Hailey, Peter Krauth, and Bob Smith. We also wish to thank the *JSS* anonymous reviewers for their comments.

10 References

Baker, B., Hardyck, C., Petrinovich, L., 1966. Weak Measurements vs. Strong Statistics: An Empirical Critique of S. S. Stevens' Proscriptions on Statistics. *Educational and Psychological Measurement*, 26, 291-309.

Benno, S., Frailey, D., 1995. Software Process Improvement in DSEG: 1989-1995. *Texas Instruments Technical Journal*, 12(2), 20-28.

Bicego, A., Khurana, M., Kuvaja, P., 1998. Bootstrap 3.0: Software Process Assessment Methodology. *Proceedings of SQM'98*.

Bohrnstedt, G., Carter, T., 1971. Robustness in Regression Analysis. In: Costner, H. (ed.), *Sociological Methodology*, Jossey-Bass.

Briand, L., El Emam, K., Morasca, S., 1996. On the Application of Measurement Theory in Software Engineering. *Empirical Software Engineering: An International Journal*, 1(1), 61-88.

Brodman, J., Johnson, D., 1994. What Small Businesses and Small Organizations Say about the CMM. *Proceedings of the 16th International Conference on Software Engineering*, 331-340.

Brodman, J., Johnson, D., 1995. Return on Investment (ROI) from Software Process Improvement as Measured by US Industry. *Software Process: Improvement and Practice*, Pilot Issue.

Butler, K., 1995. The Economic Benefits of Software Process Improvement. *Crosstalk*, 8(7), 14-17.

Card, D., 1991. Understanding Process Improvement. *IEEE Software*, July, 102-103.

Clark, B., 1997. The Effects of Software Process Maturity on Software Development Effort. PhD Thesis, University of Southern California.

Cochran, W., 1983. *Planning and Analysis of Observational Studies*. John Wiley & Sons.

Cohen, J., 1988. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates.

Cronbach, L., 1951. Coefficient Alpha and the Internal Structure of Tests. *Psychometrika*, September, 297-334.

Curtis, B., 1996. The Factor Structure of the CMM and Other Latent Issues. Paper presented at the Software Engineering Process Group Conference, Boston.

David, M., Little, R., Samuhel, M., Triest, R., 1983. Imputation Models Based on the Propensity to Respond. *Proceedings of the Business and Economics Section*, American Statistical Association, 168-173.

Deephouse, C., Goldenson, D., Kellner, M., Mukhopadhyay, T., 1995. The Effects of Software Processes on Meeting Targets and Quality. *Proceedings of the Hawaiian International Conference on Systems Sciences*, vol. 4, 710-719.

Dion, R., 1992. Elements of a Process Improvement Program. *IEEE Software*, 9(4), 83-85.

Dion, R., 1993. Process Improvement and the Corporate Balance Sheet. *IEEE Software*, 10(4), 28-35.

El Emam, K., 1998. The Internal Consistency of the ISO/IEC 15504 Software Process Capability Scale. *Proceedings of the 5th International Symposium on Software Metrics*, 72-81.

El Emam, K., Goldenson, D. R., 1995. SPICE: An Empiricist's Perspective. *Proceedings of the Second IEEE International Software Engineering Standards Syposium*, 84-97.

El Emam, K., Madhavji, N. H., 1995. The Reliability of Measuring Organizational Maturity. *Software Process: Improvement and Practice*, 1(1), 3-25.

El Emam, K., Drouin, J-N , Melo, W. (eds.), 1998. *SPICE: The Theory and Practice of Software Process Improvement and Capability Determination*. IEEE CS Press.

El Emam, K., Briand, L., 1999. Costs and Benefits of Software Process Improvement. In: Messnarz, R., Tully, C. (eds.), *Better Software Practice for Business Benefit: Principles and Experience*. IEEE CS Press.

El Emam, K., Goldenson, D., 2000. An Empirical Review of Software Process Assessments. Advances in Computers (to appear).

Flowe, R., Thordahl, J., 1994. *A Correlational Study of the SEI's Capability Maturity Model and Software Development Performance in DoD Contracts*. MSc Thesis, The Air Force Institute of Technology.

Ford, B., 1983. An Overview of Hot-Deck Procedures. In: Madow, W., Olkin, I., Rubin, D. (eds.), 1983. *Incomplete Data in Sample Surveys, Volume 2: Theory and Bibliographies*. Academic Press.

Fusaro, P., El Emam, K., Smith, B., 1997. The Internal Consistencies of the 1987 SEI Maturity Questionnaire and the SPICE Capability Dimension. *Empirical Software Engineering: An International Journal*, 3, 179-201.

Galletta, D., Lederer, A., 1989. Some Cautions on the Measurement of User Information Satisfaction. *Decision Sciences*, 20, 419-438.

Gardner, P., 1975. Scales and Statistics. *Review of Educational Research*, 45(1), 43-57.

Goldenson, D., Herbsleb, J., 1995. After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success. Technical Report, CMU/SEI-95-TR-009, Software Engineering Institute.

Goldenson, D., El Emam, K., Herbsleb, J., Deephouse, C., 1999. Empirical Studies of Software Process Assessment Methods. In: El Emam, K., Madhavji, N. H. (eds.), *Elements of Software Process Assessment and Improvement*. IEEE CS Press.

Gopal, A., Mukhopadhyay, T., Krishnan, M., 1999. The Role of Software Processes and Communication in Offshore Software Development. Communications of the ACM (to appear).

Harter, D., Krishnan, M., Slaughter, S., 1999. Effects of Process Maturity on Quality, Cycle Time, and Effort in Software Product Development. Under review for Management Science.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., Zubrow, D., 1994. Benefits of CMM-Based Software Process Improvement: Initial Results. Technical Report, CMU-SEI-94-TR-13, Software Engineering Institute.

Hosmer, D., Lemeshow, S., 1989. *Applied Logistic Regression*. John Wiley and Sons.

Humphrey, W., 1988. Characterizing the Software Process: A Maturity Framework. *IEEE Software*, March, 73-79.

Humphrey, W., Snyder, T., Willis, R., 1991. Software Process Improvement at Hughes Aircraft. *IEEE Software*, July, 11-23.

Ives, B., Olson, M., Baroudi, J., 1983. The Measurement of User Information Satisfaction. *Communications of the ACM*, 26(10), 785-793.

Jones, C., 1996. The Pragmatics of Software Process Improvements. *Software Process Newsletter*, IEEE Computer Society TCSE, 5, Winter, 1-4, Winter. . (available at http://www-se.cs.mcgill.ca/process/spn.html)

Jones, C., 1999. The Economics of Software Process Improvements. In: El Emam, K., Madhavji, N. H. (eds.), *Elements of Software Process Assessment and Improvement*, IEEE CS Press.

Jones, C., 1994. *Assessment and Control of Software Risks*. Prentice-Hall.

Kerlinger, F., 1986. *Foundations of Behavioral Research*, Holt, Rinehart, and Winston.

Krasner, H., 1999. The Payoff for Software Process Improvement: What it is and How to Get it. In: El Emam, K., Madhavji, N. H. (eds.), *Elements of Software Process Assessment and Improvement*, IEEE CS Press.

Krishnan, M., Kellner, M., 1998. Measuring Process Consistency: Implications for Reducing Software Defects. Submitted for Publication.

Labovitz, S., 1967. Some Observations on Measurement and Statistics. *Social Forces*, 46(2), 151-160.

Labovitz, S., 1970. The Assignment of Numbers to Rank Order Categories. *American Sociological Review*, 35, 515-524.

Lawlis, P., Flowe, R., Thordahl, J., 1996. A Correlational Study of the CMM and Software Development Performance. *Software Process Newsletter*, IEEE TCSE, 7, Fall, 1-5. (available at http://www-se.cs.mcgill.ca/process/spn.html)

Lebsanft, L., 1996. Bootstrap: Experiences with Europe's Software Process Assessment and Improvement Method. *Software Process Newsletter*, IEEE Computer Society, 5, Winter, 6-10. (available at http://www-se.cs.mcgill.ca/process/spn.html)

Lee, J., Kim, S., 1992. The Relationship between Procedural Formalization in MIS Development and MIS Success. *Information and Management*, 22, 89-111.

Lindsay, R., Ehrenberg, A., 1993. The Design of Replicated Studies. *The American Statistician*, 47(3), 217-228.

Lipke, W., Butler, K., 1992. Software Process Improvement: A Success Story. *Crosstalk*, 5(9), 29-39.

Little, R., Rubin, R., 1987. *Statistical Analysis with Missing Data*. John Wiley & Sons.

McGarry, F., Burke, S., Decker, B., 1998. Measuring the Impacts Individual Process Maturity Attributes Have on Software Projects. *Proceedings of the 5th International Software Metrics Symposium*, 52-60.

McIver, J., Carmines, E., 1981. *Unidimensional Scaling*, Sage Publications.

Nunnally, J., 1978. *Psychometric Theory*. McGraw Hill.

Nunnally, J., Bernstein, I., 1994. *Psychometric Theory*. McGraw Hill.

Paulk, M., Curtis, B., Chrissis, M-B , Weber, C., 1993. Capability Maturity Model, Version 1.1. *IEEE Software*, July, 18-27.

Paulk, M., Konrad, M., 1994. Measuring Process Capability versus Organizational Process Maturity. *Proceedings of the 4th International Conference on Software Quality*.

Rice, J., 1995. *Mathematical Statistics and Data Analysis*. Duxbury Press.

Rosenbaum, P., Rubin, D., 1983. The Central Role of the Propensity Score in Observational Studies for Causal Effects. *Biometrika*, 70(1), 41-55.

Rosenbaum, P., Rubin, D., 1985. Constructing a Control Group Using Multivariate Matched Sampling Methods that Incorporate the Propensity Score. *The American Statistician*, 39(1), 33-38.

Rosenthal, R.. 1991. Replication in Behavioral Research. In: Neuliep, J. (ed.), *Replication Research in the Social Sciences*. Sage Publications.

Rubin, D., 1981. The Bayesian Bootstrap. *The Annals of Statistics*, 9(1), 130-134.

Rubin, D., 1987. *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons.

Rubin, D., 1988. An Overview of Multiple Imputation. *Proceedings of the Survey Research Section*, American Statistical Association, 79-84.

Rubin, D., Schenker, N., 1986. Multiple Imputation for Interval Estimation from Simple Random Samples with Ignorable Nonresponse. *Journal of the American Statistical Association*, 81(394), 366-374.

Rubin, D., Schenker, N., 1991. Multiple Imputation in Health Care Databases: An Overview. *Statistics in Medicine*, 10, 585-598.

Rubin, H., 1993. Software Process Maturity: Measuring its Impact on Productivity and Quality. *Proceedings of the International Conference on Software Engineering*, 468-476.

Rubin, D., Stern, H., Vehovar, V., 1995. Handling ‘Don’t Know’ Survey Responses: The Case of the Slovenian Plebiscite. *Journal of the American Statistical Association*, 90(431), 822-828.

Sande, I., 1983. Hot-Deck Imputation Procedures. In: Madow, W., Olkin, I. (eds.), *Incomplete Data in Sample Surveys, Volume 3: Proceedings of the Symposium*. Academic Press.

Schaefer, J., 1997. *Analysis of Incomplete Multivariate Data*. Chapman & Hall.

Sethi, V., King, W., 1991. Construct Measurement in Information Systems Research: An Illustration in Strategic Systems. *Decision Sciences*, 22, 455-472.

Siegel, S., Castellan, J., 1988. *Nonparametric Statistics for the Behavioral Sciences*, McGraw Hill.

Software Engineering Institute, 1995. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison Wesley.

Software Engineering Institute, 1997. *C4 Software Technology Reference Guide – A Prototype*. Handbook CMU/SEI-97-HB-001, Software Engineering Institute.

Software Engineering Institute, 1998a. Top-Level Standards Map”. Available at
http://www.sei.cmu.edu/activities/cmm/cmm.articles.html, 23rd February.

Software Engineering Institute, 1998b. CMMI A Specification Version 1.1. Available at http://www.sei.cmu.edu/activities/cmm/cmmi/specs/aspec1.1.html, 23rd April.

Spector, P., 1980. Ratings of Equal and Unequal Response Choice Intervals. *Journal of Social Psychology*, 112, 115-119.

The SPIRE Project, 1998. *The SPIRE Handbook: Better Faster Cheaper Software Development in Small Companies*. ESSI Project 23873.

Stevens, S., 1951. Mathematics, Measurement, and Psychophysics. In: Stevens, S. (ed.), *Handbook of Experimental Psychology*, John Wiley.

Subramanian, A., Nilakanta, S., 1994. Measurement: A Blueprint for Theory-Building in MIS. *Information and Management*, 26, 13-20.

Treiman, D., Bielby, W., Cheng, M., 1988. Evaluating a Multiple Imputation Method for Recalibrating 1970 U.S. Census Detailed Industry Codes to the 1980 Standard. *Sociological Methodology*, vol. 18, 309-345.

Velleman, P., Wilkinson, L., 1993. Nominal, Ordinal, Interval, and Ratio Typologies Are Misleading. *The American Statistician*, 47(1), 65-72.

Wohlwend, H., Rosenbaum, S., 1993. Software Improvements in an International Company. *Proceedings of the International Conference on Software Engineering*, 212-220.

Khaled El Emam is currently at the National Research Council in Ottawa. He is the current editor of the IEEE TCSE Software Process Newsletter, the current International Trials Coordinator for the SPICE Trials, which is empirically evaluating the emerging ISO/IEC 15504 International Standard world wide, and co-editor of ISO's project to develop an international standard defining the software measurement process. Previously, he worked on both small and large research and development projects for organizations such as Toshiba International Company, Yokogawa Electric, and Honeywell Control Systems. Khaled El Emam obtained his Ph.D. from the Department of Electrical and Electronics Engineering, King's College, the University of London (UK) in 1994. He was previously the head of the Quantitative Methods Group at the Fraunhofer Institute for Experimental Software Engineering in Germany, a research scientist at the Centre de recherche informatique de Montreal (CRIM), and a research assistant in the Software Engineering Laboratory at McGill University.

Andreas Birk is a consultant and researcher at the Fraunhofer Institute for Experimental Software Engineering (Fraunhofer IESE). He has received a masters degree in computer science and economics (Dipl.-Inform.) from the University of Kaiserslautern, Germany, in 1993. He was workpackage leader in the ESPRIT projects PERFECT (no. 9090) and PROFES (no. 23239). As consultant, he has been working with European software companies in the build up and extension of their process improvement programmes. His research interests include software process improvement, knowledge management, technology transfer, and empirical methods in software engineering. He is a member of the IEEE Computer Society, GI, and ACM.