A Value-Based Framework for the Cost-Benefit Evaluation of Software Inspection Processes

S. Biffl, M. Halling

Abstract—In this paper we use the *value chain* concept, which links the level of value of the final product to the fitness of working products, for planning inspection activities. Following this value chain an inspection tackles the risk of a lower level of value due to defects in a working product from two sides: Inspection lowers the frequency of loss with removal of defects and the size of potential loss with a focus on defects that have high impact on the value chain. Thus it is important to focus inspection activities on working products and potential defects, which a) would have serious impact on the level of value of the final system, if they go undetected; and b) would be much more costly to remove during later stages of development.

We apply the framework to a simple example project to demonstrate the main concepts of inspection benefits and discuss how inspection costs and benefits can be compared for overall evaluation of inspection utility.

***Index Terms*—Software inspection, value chain, conditionally earned value, risk, uncertainty.**

I. Introduction

Project management is responsible for managing stakeholders' values during the course of a software development project: After the elicitation and ranking of success-critical stakeholders' objectives for the level of value in the finished system (e.g. with the *EasyWinWin* process [5]), the actual creation of these values has to be tracked during the project. We assume a rational development plan based on prioritized target levels of value to contain a value chain, which links the working products in the plan to reaching the planned levels of value for the final product (e.g. regarding the level of actual functionality and quality of the final system). This value chain enables project participants to assess whether a working product (any document that is created during the development process) supports the development towards a particular planned level of value (under certain assumptions regarding further development activities and interface contracts with other working products).

The risk of defects to prevent reaching the planned level of value for the project and specific development activities can be determined and managed with an appropriate risk management method, e.g. *Riskit [11]*.

Inspection is an instrument for risk management to determine product quality and mitigate risks from potential defects. The use of inspection should be planned and evaluated in an economic context of project and organizational goals. In a project plan the use of quality assurance activities should be justified with economic considerations, which support the choice of activities, e.g. using inspection of requirements rather than a larger amount of testing later on.

As inspection is an effective but resource-intensive approach to find defects, it is usually not cost-effective or may be even impossible to inspect all working products extensively. Information from risk analysis and the project value chain can be used to prioritize which working products to inspect to what extent.

Inspection with the support of a value chain and risk management can provide important benefits to project management and developers:

- *Saved rework.* Finding major defects saves potentially more costly rework and reduces the size and frequency for the risk of reduced levels of value in the target product.
- *Conditionally earned value.* Based on the value chain we introduce the concept of 'conditionally earned value' for each working product regarding its support for the target level of value in the final product. Earned value is a positive feedback for developers.
- *Reduction of planning uncertainty.* Inspection is a way to get feedback on the quality of key working products early in the development process: The information from inspection helps project management to assess risks from the working product under inspection, which reduces uncertainty on product quality and development process and thus aids project predictability.

These benefits address different aspects of software projects and require different measures for their quantification, while inspection costs are usually measured in person hours [4] or their monetary equivalent. For cost-benefit evaluation of inspection costs must be compared to benefits, which gets more complicated, if they are measured in different units.

In this paper we present an initial comprehensive framework, which extends a model presented in [4], to evaluate the impact of an inspection on the project including the above-mentioned benefit dimensions and a range of cost factors. Based on such an integrated framework for the evaluation of an inspection process, further work can develop models to optimize inspection process planning regarding two key questions of the project manager for inspection planning: a) How much inspection should be used at a given point in the project plan, and b), if inspection is used at a given point, which inspection design is likely to be most cost-effective?

Stefan Biffl is currently with the Fraunhofer Institute for Experimental Software Engineering, Sauerwiesen 6, D-67661 Kaiserlautern, Germany; on sabbatical leave from the Vienna University of Technology, Austria, (e-mail: Stefan.Biffl@tuwien.ac.at).

Michael Halling is with the Institute for Software Technology at Vienna University of Technology, Karlsplatz 13, A-1040 Vienna, Austria and with the Systems Engineering & Automation department at Johannes Kepler University Linz, Altenbergerstr. 69, A-4040 Linz(e-mail: halling@swt.tuwien.ac.at).

The rest of the paper is structured as follows: Section II presents a technical and organizational framework for inspection process evaluation. Section III introduces an economic framework for cost-benefit evaluation of inspection activities illustrated with a practical project example. Section IV summarizes the concepts on inspection cost and benefit presented and suggests further work.

II. Evaluating the Inspection Process

The traditional inspection process basically consists of the steps planning, individual defect detection, defect collection, and rework [12]. Figure 1 shows the process of inspection at a particular point in time during a software development project, e.g. inspection after development of the requirements specification.

The overall inspection process consists of 6 sub-processes, which are executed in sequence (1-6). The large box contains processes 1 and 6, which decide on whether to use inspection at all at this point (process 1) and on what further quality assurance (QA) activities to schedule for further development (process 6), depending on the feedback from inspection on the quality of the inspected product. Processes 1 and 2 determine the economical and organizational context of inspection, processes 3 and 4 carry out the inspection with individual defect detection and defect collection, processes 5 and 6 evaluate inspection quality and product quality.

In this paper we focus on evaluating inspections (processes 5 and 6) based on an economic cost-benefit model. In general, inspection is useful if it is feasible and more cost-effective than other quality assurance approaches in the project to find important potential defects in the working product. An important advantage of the inspection technique is its applicability to early life-cycle documents, like requirements specification and design documents, since ensuring the quality of working products offers better leverage in early stages of the project (see also [11]).

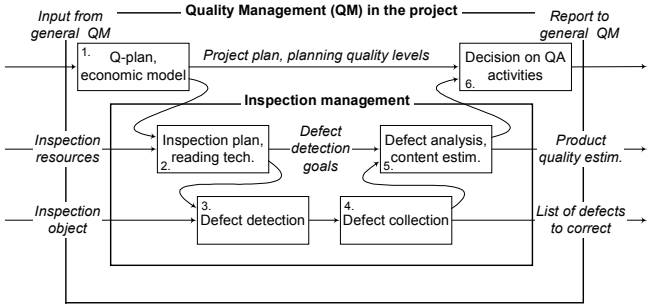

Figure 1: A hierarchical, technical view on software inspection in a project stage [3].

For a more rational discussion of the value of an inspection the following subsections introduce the notion of "defect potential" as starting point for a value/risk-based decision on the schedule/budget of quality assurance activities in a project context; and parameters of defect detection activities and their relationships to the effectiveness and efficiency of an inspection design. These technical dimensions of software inspection are important inputs to reasoning on the value of an inspection.

A. Defect Potential – Source of Risk for System Value

We view the software development process as a chain of sub-processes with the aim of creating final products or services, which provide pre-defined levels of value to stakeholders. The value chain for a particular value item in the target system is a set of conditions, which are to be fulfilled to reach a certain level of value. These conditions are determined during the definition of the requirements and refined and transformed for working products in the course of development.

While the development process creates value it also creates progressively more complex products, in which the entropy usually increases, e.g. spread from defects introduced at the requirements or design stage, unless quality assurance activities like inspection or testing are applied which help to remove defects and thus reduce unnecessary entropy[1].

However, as quality assurance in general and inspection in particular is costly, not all working products can usually be checked with the same care and therefore a prioritization must be made. We suggest using information from project risk analysis for this purpose. For risk management, e.g. with the *RiskIt* approach [11], the chain of products and processes, which work towards creating the final product, has some uncertainty of succeeding. The risk is to lose some of the expected level of value of the final system.

The *RiskIt* framework [11] defines a comprehensive qualitative model to visualize, formalize, identify, and manage risk scenarios. A rather quantitative approach to risk is represented by the *Risk Exposure measure*, which is defined as the product of the size of a loss and the probability that this loss occurs [6]. The notion of defect potential can be easily integrated with both approaches.

In the RiskIt framework, defect potential can be viewed as a parameter in risk management on project level, which describes the possible impact of potential defects in a product (part) on the development results, i.e. the utility of success-critical stakeholders. For major defect potential this implies first, that the product must be important for the final result, and second, that also the potential defects suspected in the product must be a major threat to the value of the final result. In the Risk Exposure context, defect potential in a similar way depends on the size of the potential loss (i.e. the impact of the working product on the final value of the project) and the probability of this loss (i.e. the defect density of the working product).

Therefore the context of risk exposure provides a basic guideline for deciding on whether a product qualifies to be important enough for inspection. In a well-structured project/system plan there are rather few working products on the value chain to create a certain part of the overall system value. In this case only these few products have to be

[1] Note that development activities can also be used for quality assurance, if they provide a different view on a development product and are used accordingly (see also [1] where inspection uses design activities to examine the consistency of a requirements specification).

checked for defects and the risk of loosing the value can be contained. For very tightly coupled complex systems the high number of components, which have to work together to provide the overall system value, may preclude the capability to show that a level of service is supported with a set of working products (e.g. functionality or performance features).

In practice the first aspect of defect potential, i.e. the document's importance for the project's total value, is often fairly straightforward to assess and mainly depends on the specific project situation and development process. The second aspect of defect potential is more difficult to assess *a priori* because usually the project manager does not know the number and type of defects within a working product before inspection, but has to estimate the defect density.

This defect density estimate should reflect the project manager's assumptions on the document complexity, the document author's qualification, and the general risk attitude towards the project outcome. In practice case studies with and without inspection can provide data on defect densities in typical working products and the overall impact of defined classes of defects on development results.

We want to consider a project example throughout the paper to explain our concepts and approaches. The example project is a web-based legal information system: Main value for the client is to provide a sophisticated query engine for a large amount of legal documents stored in a database. Of course, the product must provide appropriate viewing and printing facilities and a state-of-the-art user interface. We follow a standard software engineering process with requirements specification, design, implementation and testing stages. Due to a limited amount of resources for quality assurance we cannot verify the quality for all working products and must therefore assess the defect potential of working products for prioritization.

We use the *RiskIt* framework for risk management and identify two major risks for reduced client utility: 1. Not representing the data appropriately and 2. Not providing sufficiently sophisticated search functionality. Therefore we identify working products that are important in this context and find at the requirements stage two important documents: The data representation and the query engine requirements.

From past project experience we know that the authors are familiar with the project context. While the specification of the sophisticated search functionality is complex and extensive (we assume a defect density of 3 to 5 defects per page), the data representation is well established and evaluated from former projects. Therefore we assign the highest defect potential to the requirements specification of the query engine, which is accordingly selected for inspection.

B. Inspection Process Designs

Apart from the prioritization of working products the defect potential assessment discussed in the previous section determines an inspection goal with respect to defect types in a document. Therefore project managers must select a specific inspection process design that fits the inspection goal in general.

Important parameters for the effectiveness and the efficiency of a particular inspection design are the number and sequence of process steps (e.g. individual defect detection, team meeting, rework), the team size, defect detection techniques applied, and the capabilities of the inspectors in the team to fulfill their roles. As far as defect detection techniques are concerned, state-of-the-art defect detection techniques, like checklists [8] or reading techniques [1][12] can look for a range of defect types, which may be present in a particular type of product.

As we outlined above, for development products of typical size and complexity we assume that only a limited number of quality assurance checks can be executed exhaustively for all parts of the product. Therefore some working products or even parts of working products are selected for inspection based on their defect potential. Then an appropriate inspection design must be selected in order to achieve the inspection goal of removing certain critical target defect types from the selected working products effectively. Accordingly the technical dimension of defect detection process design influences the total value contribution of an inspection considerably.

Another aspect of inspection design is the relationship to inspection costs: different inspection designs of course induce different costs. Therefore it is important to determine the inspection design, which fulfills the specified inspection goals with minimal costs.

In practice a project manager can select an appropriate inspection design from past project experience or published empirical data. There are a number of reports on experiments with different inspection designs, defect detection techniques, and team sizes [1][2][8][12]. From these reports the effectiveness and efficiency of different inspection process designs with respect to different document types and other influencing variables can be assessed. The project manager can the use this data from such experiments or from appropriate local assessments at a company to evaluate the effectiveness of hypothetical inspection designs (see e.g. [2]) and determine the appropriate design for his particular context.

Continuing the example from the previous section the project manager's goal is to inspect the query engine requirements document in order to guarantee that the final product provides appropriate searching functionality. Suppose further that the project manager can assign at most 4 developers to inspection for a maximal duration of 2 days. As far as defect detection techniques are concerned both checklists [8] and perspective-based reading [1] are possible, as all developers have received appropriate training before. As the project manager estimates the requirements document to have a large defect potential, he decides to chose the following inspection process design: take all 4 developers; inspection duration equals two days; one inspector uses a checklist and the remaining 3 inspectors apply 3 different perspectives [12]. As empirical data indicates that meetings are inefficient he further decides to skip the inspection meeting.

III. An Economic Framework for Software Inspection

Often software engineering and quality assurance of software development focus on technical processes. However, software development is a very competitive business and therefore project and quality managers have to make decisions on a technical and especially on an economic basis. In this paper we see inspection as an investment into quality assurance, which should yield a positive contribution to the overall net gain of the project. If this is not the case then inspections should not be done. A major challenge in this context is to adequately measure inspection benefits and costs.

Figure 2 shows the relationships between the previously discussed technical views on defect potential and the defect detection process on one hand and the corresponding economic notion of costs and benefits, which enable the economic evaluation of inspection processes on the other hand.

The assessment of *defect potential* determines the documents to be inspected influencing the inspection *costs* mainly through document size and importance. Furthermore it has an impact on the selection of the *inspection process* as it provides information on target defects. The *benefit* from inspecting a document with a large defect potential is the removal of defects and a related information gain.

The *inspection process* is a major determinant of inspection costs through different efforts for defect detection techniques and team sizes.

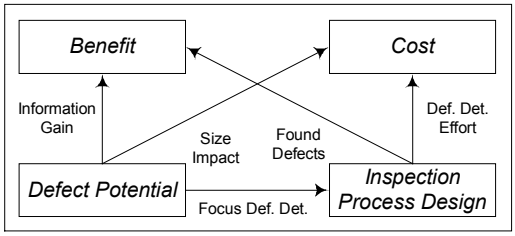

Figure 2: Framework overview and relationships.

A. Model Assumptions

For the following cost-benefit discussion certain assumptions are important in order to facilitate the presented concepts:

- We discuss the inspection benefits and costs from the viewpoint of the project manager responsible for the development of the project.
- We assume that inspecting a document with an appropriate focus is deterministic and yields the goals expected by the project manager (we ignore uncertainty from inspection). That means we view inspection as an investment possibility with certain costs and certain future benefits. Of course, in practice inspection uncertainty is a real problem, but its discussion exceeds the scope of this paper.
- We assume that the project manager has the power, ability, and environment to remove any detected quality deficiencies after inspection. That means, we ignore cases where the project manager and his team are unable to complete a project.
- We assume that the project manager has limited resources for inspection in the document. That means, he cannot simply buy additional resources for additional inspections. We focus on the case where the project manager has to optimally use a limited amount of resources for quality management.

These assumptions simplify aspects of project reality to enable us to describe our approach to cost-benefit analysis in an understandable way. Of course, we know that they do not represent the situation in practice. However, we think that they adequately model reality for our purposes in this work. Further work in this area must then relax the most stringent assumptions and adjust the base model in an appropriate way.

B. Benefits of Inspection

The benefit framework models the potential benefits of inspecting a working product during software development. From a technical point of view, software inspections find defects early, so they can be removed from the working product, which reduces system entropy and the spreading of defects to other working products.

Table 1 summarizes the main benefits associated with software inspection of software development products. It provides a short description and information on the required context and the unit of measure. The required context determines additional information that must be available in order to quantify the benefit. The unit of measures indicates how the benefit is usually measured.

Table 1: Software Inspection Benefits.

Name	Description	Required Context	Unit
Saved Rework	Early removal of defects reduces rework later on	Defect Classification	Person Hours
Conditionally Earned Value	Conditional assessment of achieved value level	Working product prioritization; set of conditions	Percentage of project value (in monetary units) completed.
Reduction of Planning Uncertainty	Variability of uncertain project variables is reduced	Risk Framework; System Interdependencies	Variability in Percentage of base project variable.

In addition to the benefits mentioned in table 1 a fourth category of benefits often enters discussion: S*oft inspection benefits*, which denote potential benefits from improved communication and teaching of system and domain information in the development team. This benefit is hard to quantify; further, if communication and teaching are important in the project, they should be provided, but not necessarily as part of an inspection. Thus we exclude soft issues from further analysis.

In the following we are going to explain and discuss the different benefits summarized in table 1 in more detail.

1) Saved Rework

The first, most obvious and often discussed benefit is *Saved Rework* from the early removal of defects. We measure this benefit in working hours saved. A prerequisite for calculating the saved rework is a defect classification that determines the *severity* of a detected and removed defect.

In this context defect severity measures the amount of re-

work caused by a defect of a certain severity level, if it is not removed early during inspection, but later on during the testing phase or operation. Defect severity may vary with the development phase in which the defect would have surfaced. The number of defect severity classes should follow the rationale to use enough defects classes to allow expressing the magnitude of impact they have on development, while restricting the number of defect classes to a variety that can readily be understood, used, and retained by an inspector who has to classify defects.

The benefit differences between the defect severity levels should be in a range that helps inspectors and managers to consistently assign a defect to a severity level, e.g., benefits of neighboring classes should differ at least by a factor of 2.

There are several approaches to determine the benefit for a defect of a given severity class [4].

- The simplest approach is to assign each defect class a single benefit value.
- Another approach is to assume for each class a probability distribution of benefits. The expected benefit for a given defect is determined from this benefit distribution.
- A more sophisticated approach includes estimates on the benefit for several development phases, e.g., an early phase, where the impact of a defect is rather low (e.g., in-house design). While in a later phase the impact is much higher, since more rework is necessary and more people are involved (e.g., operation at the customer).

Continuing our example, the web-based legal information project, suppose that an inspection was done and that all defects detected during inspection were classified using a 2 level severity classification. Based on past project experience or empirical studies the project manager assigns certain values of saved rework benefit to each severity level. Suppose that finding a major defect saves on average 8 hours and finding a minor defect saves on average 1 hour of rework [8]. In total 50 minor and 20 major defects were found during inspection, yielding a total saved rework benefit of 50*1+20*8=210 person hours.

2) Conditionally Earned Value

In general, an important benefit of inspecting a document is the associated information gain for the project manager. However, this information gain is in some sense multidimensional and therefore we define two different benefit factors to deal with it.

The *Conditionally Earned Value* benefit factor covers the information gain related to the quality of a specific, inspected working product. Every working product adds a certain contribution to the total value of the software product developed for the client. The ideal case would be to inspect and fully quality assure every working product. In this case the project manager would have full information on the project status. Assuming that he can take appropriate actions to remove any quality deficiencies, he could successfully complete the project.

In practice, however, projects have to deal with scarce resources and therefore this value maximizing approach is not realizable. A reasonable approach to deal with this problem is to prioritize working products with respect to their potential value for the total project. The most important artifacts can then be inspected (see section II.A on defect potential for details on the prioritization).

Each inspection of such an important working product adds earned project value conditional on the defect potential of:

- Other working products on the same development stage and their influence on the inspected document.
- Working products in later development stages.

Before inspection the quality of a specific working product can in general be only assessed based on general context information, e.g. the credibility and knowledge of the authors. However, the project manager can usually not be sure whether the product actually delivers its designated value or not.

Now assume that in the case of the web-based legal information system the project manager faces the following probability matrix with respect to final product value (see table 2). This matrix shows the different components of the product and how much they add to the total client value (first column). From this you can see that the searching functionality represents the most important system part. The first row shows the three development stages where inspection could be possibly done and how much of end value is created in each phase. Now multiplying each row probability with the appropriate column probability completes the matrix in table 2.

Each percentage value in the shaded area indicates the percentage of total value created in this situation conditional on the assumption that everything else (on the same and later development stages) works fine. However, if the project manager does not verify the quality of a working product in a specific situation he cannot be sure that the value in this situation is really earned.

In our example the project manager has done an inspection of the query engine specification and therefore the conditionally earned value is 28% of the total project value[2].

TABLE 2: DISTRIBUTION OF FINAL PROJECT VALUE ON WORKING PRODUCTS.

System Part / Development Process		Specification (40%)	Design (30%)	Implementation (30%)
Graphical Representation	(5%)	2%	1.5%	1.5%
Viewing and Printing	(25%)	10%	7.5%	7.5%
Searching	(70%)	28%	21%	21%

3) Reduction of Planning Uncertainty

The benefit from '*Reduction of Planning Uncertainty'* or *'Better Predictability'* represents another dimension of information gain for the project manager from inspection. Con-

[2] Note that the above example is an extreme simplification of the derivation of the conditionally earned value. In further work we will focus on applying a Bayesian probability framework to this problem in order to appropriately measure the information gain due to inspecting a working product. However, the simple example is sufficient to show the intuition behind the concept of conditionally earned value.

trary to the previous benefit, it does not focus on the already created conditional value but on the reduction of project risk, which translates to project planning uncertainty. In practice every project plan faces a large amount of uncertainty resulting in deviations from the envisioned project path. This uncertainty usually involves cost, schedule, and quality plans. The quality uncertainty is obviously reduced by inspecting working products.

At project initiation the project manager can only rely on past experience for an initial project plan. Figure 3 shows a simplified illustration for the narrowing range of uncertainty of a variable like effort or functionality (y-axis) over the course of several project stages (x-axis). With each project stage the probable range of the variable values diminishes (see three possible paths) until the value in the end is fixed.

With appropriate information at an early stage of development, see e.g. the three fat points at the requirements stage, project control is relatively easy and flexible, while the fat points on the three paths (without specific action from project management) in later stages are much further apart (ellipses get more spread out), and thus harder to change to a more favorable path.

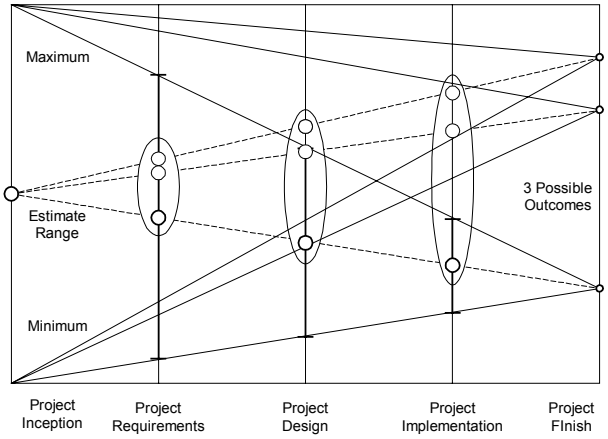

Figure 3: Information Gain for Project Planning.

Thus one benefit of inspecting a working product is a resulting reduction of variability of project variables, e.g. cost, schedule and quality plans.

A risk framework including system interdependencies and relationships represents the required context for this benefit. In this sense inspections represent a way of risk reduction and monitoring providing the project manager with information to get project estimates closer to the true value at an earlier time than without inspection.

Suppose that the project manager of the web-based legal information system considers a project cost variability of 20% of the expected total costs. After the inspection of the query engine specification he can compare the costs for creating the product with his estimated costs. Furthermore he can consider the working product's quality and the necessary effort to fix all detected defects. The project manager can use this information to adjust his project plans and to reduce variability. Suppose that the query engine specification took 5% more effort than planned and that after inspection there is another 10% of originally planned effort required to remove all defects. Based on this information, the project manager can assess that his original project cost estimates were too optimistic because it was not possible to create a working product with the required quality in the planned time. Based on this conclusion he can increase the expected total costs for the project and remove some uncertainty because he knows at least that the project will most probably not end in the lower cost range.

C. *Costs of Inspection*

In the previous section we presented various benefits associated to software inspection and also discussed that project managers must calculate and monitor project costs. Therefore for actually deciding whether and how to inspect, inspection cost factors must be taken into consideration as well.

As far as inspection costs are concerned, we basically distinguish direct and indirect costs [4]. While direct costs are directly related to the inspection process (e.g. inspection preparation, inspection meeting), indirect costs are only causally related to inspection but cannot be directly attributed to the software inspection process (e.g. project delay, slower project progress).

The determination and quantification of *indirect costs* is in general very difficult and depends on various project and company variables, e.g. software development process model, project pressure, project-planning approach. Therefore we suggest using a qualitative model in order to consider them.

The *direct costs* are in most cases *variable costs,* meaning that they change from inspection run to inspection run. We measure direct costs in person hours, which in our opinion is the relevant and appropriate unit. In practice person hours can simply be transformed into monetary units by considering the appropriate wage costs attributed to involved employees.

In general, the variable direct costs depend on the complexity and quality of the inspected document (i.e. how many defects are there to be found with which effort) and the defect detection process (i.e. how do inspectors detect defects). It is important not to forget the cost factor of removing defects detected during inspection.

As outlined in the section on inspection process design, the project manager of the web-based legal information system has chosen to allow 4 inspectors to inspect for two days. Now assume that these developers were only inspecting in these two days, then the direct costs of defect detection are 4 (team size)*8 (hours per day)*2 (number of days) = 64 person hours. Then we have to add costs for removing the defects (remember that we assumed 50 minor and 20 major defects) where we expect to need 5 hours for a major defect and 0.5 hours for a minor defect (this totals 50*0.5 + 20*5 = 125 person hours). Finally we add 21 person hours for inspection planning, management and evaluation. Therefore the total direct inspection costs are 210 person hours. In this simple example we ignore indirect costs.

D. Cost-Benefit Evaluation Approaches

In the previous sections we discussed benefits and costs of software inspection. This section presents a framework for combining benefits and costs to evaluate inspection from an economic point of view. Of course, software inspection can follow different goals like to cover a particular defect potential (document coverage), or to find as many defects as possible or to optimize inspection net gain. However each of these goals only considers a subset of available inspection benefits. Therefore we want to present a comprehensive framework that integrates all benefit and cost aspects. The challenge in this context is to relate the different benefit and cost factors to each other.

Our approach is to define the *project utility*, i.e. the utility of the project manager or the software developing company, respectively. This utility function must fulfill the following requirements in order to be a rational utility function:

- *Monotonically increasing with the product value for the client:* ensuring that the developed product satisfies client needs increases the project utility.
- *Monotonically decreasing with rising project risk:* If the project suffers from large potential risk, the project manager's risk is reduced.
- *Monotonically decreasing with rising project costs:* obviously the project utility is reduced by project costs.

Table 3 summarizes the variables required to derive an exemplary project utility function, which is denominated in monetary units (i.e. all variables are denominated in monetary units).

TABLE 3: VARIABLES OF COST-BENEFIT EVALUATION APPROACH

Variable	Description
U	Project Utility
p	Conditionally Earned Value in percent of the total value
TV	Fair price determined with the client for the Total Value
TC	Total Estimated Project Costs (including total inspection costs)
SR	Saved Rework in monetary units
k	Risk Based Discount Rate

Based on the assumptions presented in section III.A, we propose the following utility function, as it satisfies the above requirements for the project utility and can be interpreted as a *Net-Present-Value*-oriented approach [7]. Note, that for equation 1, the most important assumption is that the inspection process quality is known *a priori*, meaning that we exclude any uncertainty from the inspection process.

$$U = \frac{p \cdot TV + SR}{1 + k} - TC \qquad \text{(eqn. 1)}$$

The numerator of this utility function describes the conditionally earned project gain, i.e. project benefit minus project costs, in monetary units based on the current project status. The project benefits are the conditionally earned value and the saved rework appropriately transformed into monetary units in the project context. The project costs are the expected total project costs including inspection costs.

In order to measure the benefits and costs of project uncertainty we define the variable *k*, which represents a risk-adjusted discount rate in the range of 0% to 100%. If there is no uncertainty in the project then *k* has a value of 0%, if there is much uncertainty *k* has a value of 100% (basically it would be possible to allow for unlimited *k*). As the project benefits are conditional on further development and are going to be realized in the future, it is appropriate to discount them (i.e. divide them by *1+k*). Harrison *et al. [10]* present a similar way to quantify the benefit of better project predictability due to process improvement. They argue that the discount factor *k* can be motivated from the *capital asset pricing model* (CAPM). We follow this approach, as the general idea of discounting uncertain future benefits is reasonable, however, we doubt in our context the simple applicability of concepts like the CAPM, which are well-established in finance, and therefore prefer to base our evaluation approach on utility theory.

This project utility is defined in terms of economic variables. However, the technical aspects of software inspection, i.e. defect potential of working products and different inspection designs, are all appropriately taken into consideration. Variable *p* covers the inspected document's defect potential, variable *SR* depends on the inspected document's defect density and variable *TC* is influenced by the inspection process design. The following simple example shows how the results from previous sections are combined and used to derive the project utility.

Consider the web-based legal information system as an example. In order to evaluate the benefit of inspection we compare the situation with and without inspection to each other. Suppose that before doing the software inspection the project manager faces the following project variables: we normalized the total project value and expected total project cost without inspection equal to 1 (TV=TC=1); the risk-adjusted discount rate equals 20% based on the experience and expertise of the project team. Furthermore we need an *a priori* probability for the earned value after the requirements phase, which we set equal to 20%. Then the project utility is [(0.20*1)/1.20-1], which is equal to –0.83.

If we now consider the situation with inspection then the following variables change: the total project costs are increased by 210 person hours; the saved rework benefit also amounts to 210 person hours; the conditionally earned value probability equals 28% and the risk-adjusted discount rate is reduced to 15%. Note, that we further assume that the benefits from saved rework and the inspection costs offset. Therefore the project utility with inspection is as follows: [(0.28*1)/1.15-1]= –0.75.

In both situations the utility is negative which is quite understandable as the conditionally earned benefit is rather low after the requirements stage. However, the project utility with inspection is higher than the utility without inspection. That is the case even though the saved rework benefits and the inspection costs just offset, which is a rather conservative example. See for example the report based of an empirical study [4], where saved rework on average more than offset inspection costs even with conservative benefit assumptions.

Before we conclude our cost-benefit evaluation, it is important to point out that the presented utility function is just one possibility and a very important aspect of further work in this area is to establish a well-founded utility theory for stakeholders participating in the software development process. Therefore we view this framework as a first approximation for a functional relationship that can be used in the future for optimization and decision-making. Basically it offers the possibility to either maximize project utility with respect to product value or with respect to risk reduction.

IV. Summary and Further Work

In this paper we introduced a framework to evaluate the cost-benefit of an inspection process, which is based on the value of a software system and its parts, for more rational approaches to inspection planning. The approach presented is not operational in the sense that it does not provide practical guidelines on how to optimize inspections. The main contribution of this analysis to inspection research is that it presents a simplified but in our opinion illustrative model of inspection benefits and costs that allows for qualitative evaluation of inspection processes.

We assume that a rational project plan contains a value chain, which links the level of value of the final product to the fitness of working products, for planning inspection activities. Following this value chain an inspection tackles the risk of a lower level of value from defects in the working product from two sides: Inspection lowers the frequency of risk with removal of defects and the size of potential loss through a focus on defects with high impact on the value chain.

Thus it is important to focus inspection activities on working products and potential defects, which a) would have serious impact on the level of value of the target system, if they go undetected; and b) would be much more costly to remove later during development.

This paper discussed costs and benefits, namely saved rework, conditionally earned value, and reduction of planning uncertainty, to evaluate the cost-benefit of an inspection process with a project example.

The concept of the 'conditionally earned value' of a working product to support a certain level of value is helpful to determine the value of inspecting a good-quality product, where only little benefit can come from finding defects. The concept is by itself interesting as it expresses a paradigm shift from finding defects to showing value of a working product in the value chain with explicit consideration of benefits of a system under development and not only defect count. This positive view is well-suited to motivate developers to actively take part in quality assurance.

The approaches presented in this paper, the value chain and the economic framework for inspection, provide added value from inspection for project managers and developers as they help to make inspection better planable in the project context and support project control.

Using this rather complex model compared to "simple" guesses enables more accurate evaluation of inspection runs. Actually, simple evaluation estimates can only be based on defects reported and effort spent during inspection, i.e. measures that are easily observed after inspection. However, these simple evaluation measures do not adequately describe inspection benefits and ignore less obvious but most important benefits, like conditionally added value and improved predictability. In order to further motivate application of inspections in practice we think that it is important to qualitatively assess the different dimensions of inspection costs and benefits. Of course, usage of our model in practice suffers from problems of estimating certain model parameters for the value chain and the project predictability. However, expert opinion, historic data, company standards or Monte Carlo simulation of parameter values can be used to estimate these parameters.

Therefore further work in this direction will focus on providing practical approaches to estimate model parameters and to develop models to optimize inspection process planning. There are two main questions the project manager has to decide for inspection planning:

- Which working products should be inspected at what point during development? How much inspection is enough for these working products?
- Which inspection approach is likely to be most effective in the particular development situation?

Next steps are to conduct empirical investigations on inspection which consider project context of inspection as well as assumptions on spreading of defects and increase of rework effort under 'usual' development conditions and the effectiveness range of inspection designs in a range of given circumstances.

Based on empirical data from these investigations as well as expert opinion, historical data, and assessments from similar local projects (related work, experiments in literature, industry information from literature) a long-term goal can be to provide models for the simulation of inspection in order to determine important factors for inspection effectiveness for given context parameters.

Acknowledgements

Michael Halling has been supported by the Austrian Science Fund, Grant P-14128-COSIMIS. Stefan Biffl has been supported in part under the Austrian Science Fund, Grant J-1948-INF.

References

[1] Basili V., Green S., Laitenberger O., Lanubile F., Shull F., Soerumgaard S., and Zelkowitz M., "The Empirical Investigation of Perspective-Based Reading", *Empirical Software Engineering: An International Journal*, vol. 1, no. 2, 1996, pp. 133-164.

[2] Biffl St. and Gutjahr W., "Influence of Team Size and Defect Detection Methods on Inspection Effectiveness", *Proc. of IEEE Int. Software Metrics Symposium,* London, April 2001.

[3] Biffl St., "Hierarchical Economic Planning of the Inspection Process", *Proc. of the 3rd Int. Workshop on*

Economics-Driven Software Engineering Research (EDSER-3) at the Int. Conf. on Software Engineering, Toronto, Canada, Comp. Soc. Press, May 2001.

[4] Biffl St., Freimut B., and Laitenberger O., "Investigating the Cost-Effectiveness of Reinspections in Software Development", *Proc. of Int. Conf. on Software Engineering, Toronto, Canada, Comp. Soc. Press*, May 2001.

[5] Boehm B., Grünbacher P., and Briggs R., "Developing Groupware for Requirements Negotiation", *IEEE Software May/June 2001*, p. 46-55.

[6] Boehm B. and Port D., "Risk-Based Strategic Software Design: How Much COTS Evaluation is Enough?", *Proc. of the 3rd Int. Workshop on Economics-Driven Software Engineering Research (EDSER-3)* at the Int. Conf. on Software Engineering, Toronto, Canada, Comp. Soc. Press, May 2001.

[7] Brealey R. and Myers S., *"Principles of Corporate Finance"*, 5th Edition, McGraw-Hill, 1996.

[8] Gilb T. and Graham D., "*Software Inspection"*, Addison-Wesley, 1993.

[9] Halling M. and Biffl St., (2001) „Using Reading Techniques to Focus Inspection Performance", *Proc. Euromicro 2001 Workshop on Software Product and Process Improvement, Warsaw,* IEEE Comp. Soc. Press, Sept. 2001.

[10] Harrison W., Settle J., and Raffo D., "Assessing the Value of Improved Predictability Due to Process Improvements", *Proc. of the 3rd Int. Workshop on Economics-Driven Software Engineering Research (EDSER-3)* at the Int. Conf. on Software Engineering, Toronto, Canada, Comp. Soc. Press, May 2001.

[11] Kontio J., "The Riskit Method for Software Risk Management, Version 1.00", *Tech. Rep. CS-TR-3782, Dept. of Comp. Sci., University of Maryland*, College Park, MD, 1997.

[12] Laitenberger O., *"Cost-effective Detection of Software Defects through Perspective-based Inspections. PhD thesis"*, University of Kaiserslautern, Germany, www.iese.fhg.de, May 2000.