ISSN:1353-7776

DESMET: A method for evaluating Software Engineering methods and tools

Barbara Kitchenham

Technical Report TR96-09

August 1996

Department of Computer Science
University of Keele
Keele
Staffordshire
ST5 5BG
U.K.

TEL: 01782 583075
FAX: 01782 713082

Abstract
This report describes the results of the DESMET project. The DESMET project was a collaborative project part funded by the U.K. DTI. Its goal was develop and validate a method for evaluating software engineering methods and tools. This report describes the guidelines developed by DESMET for

- selecting an appropriate evaluation method;
- performing a feature analysis;
- performing a quantitative case study.

It also describes some of the management and sociological issues that can influence evaluation exercise. Finally, the report describes briefly the attempt the project made to evaluate the DESMET method itself.

CONTENTS

1. Introduction

This report describes some of the work of the DESMET project. DESMET was a collaborative project part-funded by the UK Department of Trade and Industry which aimed to develop and evaluate a method for evaluating software engineering methods and tools. The project partner were the National Computing Centre, University of North London, GEC-Marconi and BNR-Europe.

This report gives an overview of the DESMET method, and describes in some detail procedures for undertaking Quantitative case studies and Qualitative Feature Analysis. DESMET also developed guidelines for formal experiments [1] but these are not presented in detail in this report.

Section 2 of the report provides an introduction to the DESMET project. Section 3 describes the nine different evaluation methods identified by the DESMET project. Section 4 discusses criteria to help you select the most appropriate evaluation method given your own specific circumstances. The DESMET project noted that many problems associated with an evaluation exercise were not strictly technical matters but were managerial and sociological issues. Section 5 gives a brief introduction to those concerns. Section 6 and 7 discusses specific evaluation methods in detail. Section 6 discusses Feature Analysis and Section 8 discusses quantitative case studies. Finally, Section 9 describes the work that the project undertook in order to evaluate the DESMET method itself.

2. The DESMET method: scope, terminology and limitations

This section describes the context in which you might want to use the DESMET results. It introduces some of the terminology that is used throughout the report and identifies some of the limitations of the method.

2.1 DESMET users

DESMET is concerned with the evaluation of methods or tools within a particular organisation. The term *organisation* is meant to apply to a software development group in a particular company/division performing broadly similar tasks under similar conditions. In addition, the DESMET method can be used by academic institutions interested in experimental software engineering.

The DESMET method is intended to help an evaluator in a particular organisation to plan and execute an evaluation exercise that is unbiased and reliable (e.g. maximises the chance of identifying the best method/tool). One of the problems with undertaking evaluations in industry is that as well as technical difficulties, there are sociological and managerial factors that can bias an evaluation exercise. For example, if staff undertaking an evaluation believe a new tool is superior to their current tool, they are likely to get good results; however, the favourable results might not carry over to other software staff who do not have the same confidence in the tool. We have attempted to address this problem in our evaluation guidelines.

2.2 Evaluation context

A DESMET evaluation exercise is comparative. That is, we assume that there are several alternative ways of performing a software engineering task and we want to identify which of the alternatives is best in specific circumstances. In some cases, comparisons will be made against an theoretical ideal (i.e. a gold standard).

With the exception of formal experiments, DESMET evaluations are context-dependent, which means that we do not expect a specific method/tool to be the best in all circumstances. It is possible that an evaluation in one company would result in one method/tool being identified as superior, but a similar evaluation in another company would come to a different conclusion. For example, suppose two companies were comparing the use of formal specification notations with the use of Fagan Inspections as a means of reducing the number of defects that reach the customer. If one of the companies had a mathematically sophisticated workforce, it might find formal specification superior in an evaluation; another company with a commercially-oriented workforce might find Fagan Inspections superior. Hence, the difference in the results of the evaluation might be due to the properties of the companies that perform the evaluations, not to the methods themselves.

However, as Pfleeger [1] has described, formal experiments are not context dependent. Under fully controlled conditions, we expect a formal experiment to give the same results irrespective of the particular evaluator or the particular organisation.

2.3 Evaluation Objects

DESMET assumes that you may want to evaluate any one of the following objects:

- a *generic method* which is a generic paradigm for some aspect of software development (e.g. structured design);
- a *method* which is a specific approach within a generic paradigm (e.g. Yourdon structured design);
- a *tool* which is a software application that supports a well-defined activity.

Sets of methods or tools being evaluated together are treated in the same way as an individual method or tool. A method/tool combination is usually treated as a method if different paradigms are being compared, and a tool, if different software support packages for the same paradigm are being compared.

2.4 Evaluation types

The DESMET evaluation method separates evaluation exercises into two main types:

- evaluations aimed establishing measurable effects of using a method/tool;
- evaluations aimed at establishing method/tool appropriateness i.e. how well a method/tool fits the needs and culture of an organisation.

Measurable effects are usually based on reducing production, rework or maintenance time or costs. DESMET refers to this type of evaluation as a quantitative or objective evaluation. Quantitative evaluations are based on

identifying the benefits you expect a new/method or tool to deliver in measurable terms and collecting data to determine whether the expected benefits are actually delivered.

The appropriateness of a method/tool is usually assessed in terms of the features provided by the method/tool, the characteristics of its supplier, and its training requirements. The specific features and characteristics included in the evaluation are based on the requirements of the user population and any organisational procurement standards. Evaluators assess the extent to which the method/tool provides the required features in a usable and effective manner based (usually) on personal opinion. DESMET refers to this type of evaluation as feature analysis and identifies such an evaluation as a qualitative or subjective evaluation.

Some methods involve both a subjective and an objective element. DESMET calls these **hybrid** methods. This are discussed in a later section.

2.5 Evaluation procedure

In addition to the separation between quantitative, qualitative and hybrid evaluations, there is another dimension to an evaluation: the way in which the evaluation is organised. DESMET has identified three rather different ways of organising an evaluation exercise:

- as **a formal experiment** where many subjects (i.e. software engineers) are asked to perform a task (or variety of tasks) using the different methods/tools under investigation. Subjects are assigned to each method/tool such that results are unbiased and can be analysed using standard statistical techniques;
- as **a case study** where each method/tool under investigation is tried out on a real project using the standard project development procedures of the evaluating organisation;
- as **a survey** where staff/organisations that have used specific methods or tools on past projects[1] are asked to provide information about the method or tool. Information from the method/tool users can be analysed using standard statistical techniques.

Although the three methods of organising an evaluation are usually associated with quantitative investigations, they can equally well be applied to qualitative evaluations.

2.6 Limitations of the DESMET method

The DESMET method is of limited use if:

1. you want to "mix and match" methods and tools;
2. your organisation does not have a controllable development process.

The DESMET method is aimed at evaluating specific methods/tools in specific circumstances. With the exception of hybrid methods such as the use of expert opinion, evaluation methods do not give any indication as to how different methods and tools interact.

For example, suppose a company undertakes two parallel, independent evaluations: one investigating Fagan Inspection and the other investigating the use of a formal specification language such as Z. Supposes the results of evaluation 1 suggest that Fagan Inspections are better than the company's current walkthrough standard for detecting specification defects (e.g. 80% of specification defects found before product release when Fagan Inspections are used compared with 45% using walkthroughs). In addition suppose that the results of evaluation 2 suggest that the use of Z specifications results in fewer residual specification defects than the company's current natural language specifications (e.g. 1 specification fault per 1000 lines of code found after release when Z is used compared with 10 specification faults per 1000 lines of code with current methods). The evaluator cannot assume that the results of changing the organisations process to using both Z specifications and Fagan Inspection will be independent. That is the DESMET method does not guarantee that the number of specification defects per 1000 lines of code after release will be 0.2 per 1000 lines instead of 4.5 per 1000 lines. It may be a reasonable assumption, but the combined effect of Z and Fagan Inspections would still need to be explicitly confirmed. In this case, the effect may not be what you expect because each technique might eliminate the same type of fault as the other.

Undertaking industrial evaluations is usually only worthwhile if the organisation intends to make widespread process changes. If you work for an organisation which undertakes a variety of different applications using

[1] In some circumstances it may be possible to organise a survey based on the assumption that respondents would be interested enough to set up a trial of a method/tool and return the information once they have finished the trial. However, this may result in a low response to the survey and a rather long timescale.

methods/tools specified by your customers (e.g. you produce custom-made software products using client-dictated development procedures), you may not have a standard development process. Thus, the results of an evaluation exercise undertaken on a specific project may not generalise to other projects because each future project is likely to use a different process. In such circumstances the value of any evaluation exercise is likely to be limited.

In general, the parts of the DESMET method you will be able to use depend on the organisation in which you work. You will need to assess the "maturity" of your organisation from the viewpoint of the type of evaluation it is capable of performing.

3. Evaluation methods

This section defines the nine evaluation methods identified by the DESMET project.

3.1 Quantitative evaluation methods

Quantitative evaluations are based on the assumption that you can identify some measurable property (or properties) of your software product or process that you expect to change as a result of using the methods/tools you want to evaluate.

Quantitative evaluations can be organised in three different ways: case studies, formal experiments, and surveys. DESMET has concentrated on developing guidelines for undertaking formal experiments and case studies of software methods/tools. Formal experiments are the basic scientific method of assessing differences between methods/tools. DESMET has tried to adapt as much as possible of the principles of formal experiments (i.e. terminology and techniques) to developing guidelines for performing case studies [2].

Shari Lawrence Pfleeger discusses the principles of formal experiments in a series of articles in SIGSOFT Notes [1], so they will not be discussed further in this report..

3.1.1 Case Studies

Case studies involve the evaluation of a method/tool after it has been used on a "real" software project. This means that you can be sure that the effects of the method/tool applies in your own organisation and scales-up to the particular type of projects you undertake.

Case studies are easier for software development organisations to perform because they do not require replication. However, you can have only limited confidence that a case study will allow you to assess the true effect of a method/tool.

In practice, there are a number of practical difficulties with case studies:

1. It is difficult to ensure that there is a means of interpreting the results of a case study. For example, a single productivity value of 0.02 function points per hour for a project using a new 4GL language cannot be interpreted unless there is some "baseline" productivity value for "normal" projects with which to compare it.
2. It is difficult to assess the extent to which the results of one case study can be used as the basis of method/tool investment decisions. For example, if we found the productivity of a project using a new method were twice that of other projects in our organisation, we would still not be advised to invest in that tool if the case study project was an in-house project management database and the projects we normally undertook for clients were embedded, real-time command and control systems.

3.1.1.1 Advantages of case studies

1. They can be incorporated into the normal software development activities.
2. If they are performed on real projects, they are already "scaled-up" to life size.
3. They allow you to determine whether (or not) expected effects apply in your own organisational and cultural circumstances.

3.1.1.2 Disadvantages of case studies

1. With little or no replication, they may give inaccurate results.
2. There is no guarantee that similar results will be found on other projects.
3. There are few agreed standards/procedures for undertaking case studies. Different disciplines have different approaches and often use the term to mean different things.

3.1.2 Surveys

Surveys are used when various method(s)/tool(s) have been used in an organisation (or group of organisations). In this situation, the users of the method(s)/tool(s) can be asked to provide information about any property of interest (e.g. development productivity, post-release defects, etc.).

Information from a number of different tool users implies replication. This means that we can use statistical techniques to evaluate the differences between method(s)/tool(s) with respect to the properties of interest. The major differences between a formal experiment and a survey is that a survey is usually "post-hoc" and less well-controlled than an experiment. This results in several problems with surveys:

1. They are only able to demonstrate association not causality. An example of the difference between association and causality is that it has been observed that there is a strong correlation between the birth rate and the number of storks in Scandinavia; however few people believe that shooting storks would decrease the birth rate. This limitation holds even if we believe a causal link is very plausible. For example, the basis for the tobacco industry's claim that a link between smoking and cancer is not-proven is the fact that scientists have only demonstrated an association not a causal link.
2. The results may be biased if the relationship between the population (i.e. the total number of persons in a position to respond to the survey) and the respondents is unknown. For example, in a survey aimed at obtaining information about the quality and productivity of projects using specific methods/tools, the project managers who reply are likely to be those that are well-organised enough to have extracted and stored such information. Projects run by well-organised project managers may gain more from the use of methods/tools that projects run by badly-organised project managers.

3.1.2.1 Advantages of surveys

1. They make use of existing experience (i.e. existing data).
2. They can confirm that an effect generalises to many projects/organisations.
3. They make use of standard statistical analysis techniques.

3.1.2.2 Disadvantages of surveys

1. They rely on different projects/organisations keeping comparable data.
2. They only confirm association not causality.
3. They can be biased due to differences between those who respond and those who do not respond.

Card et al [3] provide a good example of a quantitative survey (although they have not recognised the problem of association versus causality).

3.2 Qualitative Methods

DESMET uses the term Feature Analysis to describe a qualitative evaluation. Feature Analysis is based on identifying the requirements that users have for a particular task/activity and mapping those requirements to features that a method/tool aimed at supporting that task/activity should possess. This is the type of analysis you see in personal computer magazines when different Word Processors, or graphics packages are compared feature by feature.

In the context of software methods/tools, the users would be the software managers, developers or maintainers. An evaluator then assesses how well the identified features are provided by a number of alternative methods/tools. Feature Analysis is referred to as qualitative because it usually requires a subjective assessment of the relative importance of different features and how well a feature is implemented.

Feature analysis can be done by a single person who identifies the requirements, maps them to various features and then assesses the extent to which alternative methods/tools support them by trying out the method/tool or reviewing sales literature. This is the method that would normally be used when screening a large number of methods and tools. However, using an analogy with quantitative studies, DESMET suggests that it is also possible to organise Feature Analysis evaluations more formally using the same three ways:

1. Qualitative experiment: A feature-based evaluation done by a group of potential users. This involves devising the evaluation criteria and the method of analysing results using the standard Feature Analysis approach but organising the evaluation activity as an experiment (i.e. using a random selection of potential users to undertake the evaluation).
2. Qualitative case study: A feature-based evaluation undertaken after a method/tool has been used in practice on a real project.

3. Qualitative survey: A feature-based evaluation done by people who have experience of using or have studied the methods/tools of interest. This involves devising an evaluation criteria and method of analysing the results using the standard Feature Analysis approach but organising the evaluation as a survey.

3.3 Hybrid Evaluation Methods

When considering the different ways that methods/tools might be evaluated, we identified two specific evaluation methods that each seemed to have some quantitative and qualitative elements. We call these hybrid methods and describe them below as Qualitative Effects Analysis and Benchmarking. We do not claim these are the only hybrid methods, they were, however, the only ones we could think of! We have not provided any detailed guidelines for these types of evaluation.

3.3.1 Collated expert opinion - Qualitative Effects Analysis

A common way of deciding which method/tool to use on a project is to rely on the expert opinion of a senior software engineer or manager. Gilb [4] suggested assessing techniques in terms of the contribution they make towards achieving a measurable product requirement. This approach was used by Walker and Kitchenham [5] to specify a tool to evaluate the quality levels likely to be achieved by a certain selection of methods/techniques. In DESMET, we refer to this method of using expert opinion to assess the quantitative effects of different methods and tools as Qualitative Effects Analysis.

Thus, Qualitative Effects Analysis provides a subjective assessment of the quantitative effect of methods and tools. DESMET assumes that a knowledge base of expert opinion about generic methods and techniques is available. A user of the knowledge base can request an assessment of the effects of individual methods and/or the combined impact of several methods. This is quite a useful approach because the information held in a database containing expert opinion can be updated as and when the results of objective studies become available.

3.3.2 Benchmarking

Benchmarking is a process of running a number of standard tests/trials using a number of alternative tools/methods (usually tools) and assessing the relative performance of the tools in those tests. Benchmarking is usually associated with assessing performance characteristics of computer hardware but there are circumstances when the technique is appropriate to software. For example, compiler validation, which involves running a compiler against a series of standard tests, is a type of benchmarking. The compiler example illustrates another point about benchmarking: it is most likely to be useful if the method or tool does not require human expertise to use.

When you perform a benchmarking exercise, the selection of the specific tests is subjective but the actual performance aspects measured can be quite objective, for example, the speed of processing and/or the accuracy of results. Benchmarks are most suitable for comparisons of tools, when the output of the tool can be characterised objectively.

3.4 Summary of evaluation methods

This discussion has identified nine distinct evaluation types:

1. Quantitative experiment: An investigation of the quantitative impact of methods/tools organised as a formal experiment.
2. Quantitative case study: An investigation of the quantitative impact of methods/tools organised as a case study.
3. Quantitative survey: An investigation of the quantitative impact of methods/tools organised as a survey.
4. Qualitative screening: A feature-based evaluation done by a single individual who not only determines the features to be assessed and their rating scale but also does the assessment. For initial screening, the evaluations are usually based on literature describing the software method/tools rather than actual use of the methods/tools.
5. Qualitative experiment: A feature-based evaluation done by a group of potential user who are expected to try out the methods/tools on typical tasks before making their evaluations.
6. Qualitative case study: A feature-based evaluation performed by someone who has used the method/tool on a real project.
7. Qualitative survey: A feature-based evaluation done by people who have had experience of using the method/tool, or have studied the method/tool. The difference between a survey and an experiment is that participation in a survey is at the discretion of the subject.
8. Qualitative effects analysis: A subjective assessment of the quantitative effect of methods and tools, based on expert opinion.
9. Benchmarking: A process of running a number of standard tests using alternative tools/methods (usually tools) and assessing the relative performance of the tools against those tests.

All these methods of evaluation might be regarded an embarrassment of riches because they lead directly to the problem of deciding which one to use. The problem of selecting an appropriate evaluation method given your specific circumstances is discussed in the next section of this report.

4. Selecting an appropriate evaluation method

4.1 Technical selection criteria

4.1.1 Introduction

This section considers the way in which different evaluation requirements and organisation capabilities affect your choice of evaluation method. Your particular choice of evaluation method will be affected by your evaluation goals, the characteristics of the object you want to evaluate, the characteristics of the organisation you work in, and the limitations and constraints placed on the evaluation exercise. These different factors interact in complicated ways, so it is difficult to identify which evaluation method is the most appropriate.

The specific criteria that the DESMET method uses to determine your circumstances are:
1. The evaluation context.
2. The nature of the expected impact of using the method/tool.
3. The nature of the object (i.e. method/tool/generic method) to be evaluated.
4. The scope of impact of the method/tool.
5. The maturity of the method/tool.
6. The learning curve associated with the method/tool.
7. The measurement capability of the organisation undertaking the evaluation.

These issues are discussed in the following sections.

4.1.2 The evaluation context

There are (at least) four main contexts within which an industrial evaluation is performed[2]:

1. Choice of a set of methods/tools for an individual project. The approach that best fits this "mix and match" scenario is the expert opinion based Qualitative Effects Analysis.
2. Selection of methods and tools for an organisation which may involve initial screening of many alternatives followed by a more detailed evaluation of one or more short-listed methods/tools. Initial screening can be based on Feature Analysis but other criteria are needed to determine the appropriate evaluation method for a subsequent detailed evaluation.
3. Monitoring changes as part of process improvement program which can involve the evaluation of proposed change and/or evaluation of effect of adopting a new method/tool across an organisation. A specific process change can be treated as a new method/tool (depending on the extent to which it is automated). You would need to review the proposed process change and your particular situation as a whole before identifying an appropriate evaluation methods. However, if you are assessing the effect of an organisation-wide process change you should consider a survey. If you have been collecting quantitative data on your projects before and after the change a quantitative survey is possible if not you could use a Feature Analysis based survey.
4. Evaluating methods/tools for re-sale as part of a larger product or a product line. If you are selecting a specific product to sell-on as part of a product portfolio you would probably uses a Feature Analysis method of evaluation. If you are intending to purchase a software product for integration into one of your own products, you are in a formal acquisition situation which the DESMET method does not address.

The criteria discussed below are relevant when you have identified the constraints imposed by the basic context in which an evaluation is to be performed and need to identify an appropriate evaluation method for your specific evaluation requirements.

4.1.3 The nature of the impact

There are two major categories of impact:
1. Quantitative e.g. improved productivity (function points per hour), better maintainability (reduced staff effort to diagnose and correct released defects), better quality (reduced number of defects observed by customers);
2. Qualitative e.g. better visibility of progress, better usability of support tools, improve interoperability of tools, commonality of tool interfaces.

[2] Clearly, an evaluation performed by an academic organisation or a method/tool vendor would have rather different goals and context.

Quantitative effects such as improved productivity or quality are best assessed using quantitative methods, and qualitative effects such as better process visibility or controllability are best assessed using qualitative methods. However, some effects that appear to be quantitative are not easy to measure. For example:

- Some effects are only measurable when a number of projects/components are assessed together. For example, suppose we want to evaluate the extent to which a new development method reduces the risk of late delivery. Although *risk* can be defined quantitatively as the probability of some event occurring, it cannot be directly measured in terms of a single outcome. Suppose a method/tool is intended to reduce the risk of late delivery of products. If you undertake a trial of the method on a single project, the only observation that can be made is that a delivery date was achieved or not. Thus, the probability of meeting the delivery date can only be evaluated when a number of outcomes are compared i.e. when it is possible to calculate the proportion of successes in number of independent trials.
- Some quantitative effects are expected to affect projects/products other than the one on which a method/tool was used. For example, the impact of a method/tool which improved reusability could only be judged on subsequent projects, not the project on which it was first used.

Thus, quantitative effects must be broken down into two categories:

1. Directly measurable which means the effect can be measured on a single project or task. These type of effects can be evaluated using any of the quantitative evaluation methods.
2. Indirectly measurable which means the effect can only be measured on a set of project/tasks other than the project/task subjected to the method/tool being evaluated. The only quantitative method appropriate in these circumstance is a quantitative surveys. Any of the qualitative methods can be used.

For tools/methods that perform automatic transformations of inputs to outputs (e.g. compilers, code generators, test coverage analysers, natural language translation systems), improved quality is likely to encompass issues such as speed of processing, extent of input domain, quality of output. If these criteria can be quantified, benchmarks can be developed to evaluate such tools rather than undertaking other forms of quantitative evaluation.

4.1.4 Nature of evaluation object

The object being evaluated can be categorised using the following classification system:

Tool that is automated support for a well-defined activity (e.g. LOGISCOPE for structured testing, or a configuration management tool).

Method which has two categories:

1. A generic paradigm (e.g. structured analysis and design, object-oriented design, formal methods).
2. A specific approach within a generic paradigm (e.g. Jackson structured design, or Yourdon structured design).

Method/tool combinations that is a tool which is integrated with a specific method (e.g. TEAMWORK). When several alternatives are being investigated, method/tool combinations should be treated in two different ways:

1. Comparisons of several tools which support the same basic method. This is equivalent to evaluating a tool.
2. Comparison of tools which support quite different methods. This is equivalent to evaluating a specific method.

Project support environments (PSE) which refer to an integrated set of tools. Comparisons of PSEs can be of two types:

1. Comparisons of PSEs that support similar development paradigms. This is equivalent to evaluating tools.
2. Comparisons of PSEs that support completely different development paradigms. This is equivalent to evaluating different methods.

This categorisation of evaluation objects can be reduced to three distinct categories:

1. Tool
2. Method
3. Generic method

In general, the DESMET method makes no distinction between evaluating a single method/tool and evaluating a set of methods/tools. The exception is that the Qualitative Effects Analysis (Expert opinion) offers the capability of assessing the effect of a "mix and match" set of generic methods. In addition, Feature Analysis can be used to define the support requirements for many different software development activities that could be satisfied by several different methods/tools.

If the object(s) being evaluated are tools and a comparison is being made between two or more alternative tools, either Feature Analysis or Benchmarking are likely to be appropriate. Benchmarking would be preferable, if the tool undertakes automatic transformation of data with little or no interaction with the tool user (e.g. a speech synthesiser, or a compiler). However, if a comparison is being made between introducing a tool to perform a task and doing the task manually, a quantitative evaluation may be necessary (e.g. comparing ad hoc unit testing methods with structural methods based on a dynamic analysis tool).

If the object(s) being evaluated are generic methods, Qualitative Effects Analysis or Feature Analysis are likely to be appropriate. This is because any quantitative evaluation would be based on a specific instantiation of the generic method. However, an expert might be able to judge the increase in productivity or quality of one generic method relative to another, or to identify the features of an "ideal" method against which the generic methods can be assessed. If the objects(s) being evaluated are specific methods, you could use quantitative evaluation methods or Feature Analysis.

4.1.5 The scope of impact

The scope of impact of the method/tool has two major dimensions:
1. Product granularity;
2. Extent of impact.

4.1.5.1 Product granularity

Product granularity identifies whether the method/tool applies to:

1. The development (or maintenance) of a software product as a whole (e.g. specification methods or system test tools).
2. Individual parts of the product such as modules or documents (e.g. test coverage tools, or structured programming techniques).

If a method/tool applies to the product as a whole, it limits the type of quantitative case studies that can be performed and makes a formal experiment very difficult. To see a product level effect, a quantitative case study would need to have an external "baseline" for comparison purposes (e.g. values from a "sister" project(s) using another method/tool). For example, suppose we are interested in a requirements capture tool that is intended to reduce the number of specification defects that reach the customer. If we run the tool on one project and observe 1 specification defect per 1000 lines of code during acceptance testing, we would need to know the specification defect rate for other similar projects in order to assess whether the new tool was decreasing the number of specification errors.

If a method/tool works at the module level, we could use the method/tool on some modules and not on other and compare the results. For example, if we were evaluating the benefits of using detailed design inspections, we could apply the detailed design inspection method to a random subset of the modules and the current method to the remaining modules and look at the number of defects found in the modules in subsequent testing. We would then have an internal control and would be able to make an assessment of the benefit of the design inspection method using results from a single project.

4.1.5.2 Extent of impact

The extent of impact identifies how the effect of the method/tool is likely to be felt over the product/project lifecycle. In this context, we assume that a product development involves a number of different stages:

1. Concept
2. Requirements analysis and definition
3. System specification
4. System design
5. Code and unit test (depending on the development methods used)
6. Integration and system test
7. Operational use and maintenance.

In practice the development activities that relate to the different stages will overlap somewhat in time. Different methods and tools are used to perform the different software development activities. However, some methods/tools apply to many stages, for example configuration control tools and project management methods. There are 5 possibilities:

1. The effect is restricted to an activity associated primarily with a specific stage and is observable/measurable immediately from the output of the activity, e.g. the number of defects found by using alternative unit test methods.
2. The effect applies to (or is visible after) two or more stages of product development, e.g. a common language for software specifications and software design which is expected to reduce the number of design defects entering the coding stage.
3. The effect applies to the entire development phase, e.g. a full integrated software development environment which is expected to reduce development effort.
4. The effect applies to the entire lifecycle, e.g. a method/tool that is expected to reduce the costs of post release defect correction and subsequent product enhancements.
5. The effect applies to other products, e.g. a method/tool that promotes reusability and is expected to reduce development costs of subsequent projects.

If the effect applies over a wider scale than a single phase, a formal experiment is likely to be difficult to perform because it will be difficult to control all the other software development activities taking place between the stimulus (i.e. use of the method/tool being evaluated) and observation of the response (i.e. measurement of the response variable(s)). It will also be difficult to undertake anything but a toy project given the timescale and resource limitations that you are likely to experience in a software development organisation. However, you may be in a position to undertake a case study.

If the effect applies to the entire lifecycle, or to other products, a quantitative evaluation of a new method/tool on anything but toy projects may not be feasible within your timescales. However, it may be possible to perform a quantitative survey, if the method/tool has been in use for some time.

4.1.6 The maturity of the item

The maturity of the method or tool indicates the extent to which there is likely to be information about it readily available. The concept can be assessed in terms of the following categories[3]:

1. Not in use on commercial projects (i.e. still being developed)
2. Used in a few state-of-the-art products produced by your own organisation.
3. In widespread use in your own organisation.

If the object is still being developed, there would not sufficient information about the object to warrant a survey (quantitative or qualitative). However, for tools/methods that do not support human-intensive tasks, it may be possible to develop benchmarks to assess the performance of the new tools/methods.

If the object is in widespread use, a quantitative or qualitative survey is possible. However, a quantitative survey is possible within an organisation only if the organisation has an existing metrics programme.

4.1.7 Learning time

If you want to evaluate an unfamiliar method or tool, the time it would take someone to become familiar enough with the method/tool to access its capabilities or to use it effectively influences the type of evaluation that can be undertaken.

Learning time can be divided into two aspects:
1. Time required to understand the principles underlying the method/tool.
2. Time required to become proficient in its use.

This information is particularly important when deciding whether or not an experimental evaluation is feasible. If the time required to understand the principles is more than a day, or the time required to be able to use the method/tool on small examples is more than 2-3 days, a quantitative experiment is usually not likely to be viable.

4.1.8 Evaluation maturity of an organisation

DESMET assumes that the evaluation capability of your organisation can be assessed on a 4-point ordinal scale. The evaluation capability determines the type of evaluation your organisation is able to undertake. The scales points are defined as follows:

[3] This classification ignores the possibility that the method/tool is in use outside your own organisation. If the method/tool were used externally, it might be possible to organise a survey of external users. This article assumes that you are concerned with evaluations in your own organisation.

Level 1: Severely limited evaluation capability
The organisation does not have well-defined standards for software development (or has many different standards). It has no standards for using metrics. It would be difficult to generalise the results of any evaluations to other projects. Therefore, the organisation would not be able to benefit very much from the use of the DESMET method.

Level 2: Qualitative evaluation capability
The organisation has well-defined standards for software development, and adherence to those standards is monitored. However, the organisation does not have any standards for metrics and does not monitor in quantitative terms. Therefore, the organisation is able to use qualitative evaluation methods, but would need major process changes to be able to use quantitative evaluation methods effectively.

Level 3: Quantitative and qualitative evaluation capability
The organisation has well-defined standards for software development, and adherence to those standards is monitored. In addition, the organisation is accustomed to monitoring and planning individual projects in quantitative terms. However, there are limited organisational standards for metrics, and a long-term store of reliable measurements on past projects is not available. Therefore, the organisation is able to use qualitative and quantitative evaluation methods, but would need major process changes to be able to monitor trends across the whole organisation.

Level 4: Full evaluation capability
The organisation has well-defined standards for software development, and adherence to those standards is monitored. In addition, the organisation is accustomed to monitoring and planning individual projects in quantitative terms. There are organisational standards for metrics, and there is a long-term store of reliable measurements on past projects which includes information on use of methods, productivity, quality, together with other information to characterise projects. Therefore, the organisation is able to use qualitative and quantitative evaluation methods, and is able to monitor trends across the whole organisation.

Note. The only evaluation method which is independent of organisation maturity is benchmarking since it is applied to methods/tools that perform automatic transformations of inputs to outputs with little or no human intervention.

Since organisations have a tendency to over-estimate their capability, DESMET provided a questionnaire to help organisations assess their evaluation maturity level as objectively as possible [6].

4.1.9 Summary of Evaluation criteria

DESMET identified seven criteria that would influence your choice of an evaluation method. From the viewpoint of the different evaluation methods, Table 1 summarises the conditions that would favour the use of each evaluation method.

Table 1 Evaluation Method Selection

Evaluation Method	Conditions favouring method
Quantitative Experiments	Benefits clearly quantifiable. Staff available for taking part in experiment (i.e. performing non-productive work). Method/tool related to a single task/activity. Benefits directly measurable from task output. Relative small learning time. Desire to make context independent method/tool assessments.
Quantitative Case Studies	Benefits quantifiable on a single project. Benefits quantifiable prior to product retirement. Stable development procedures. Staff with measurement experience. Timescales for evaluation commensurate with the elapsed time of your normal size projects.
Quantitative Surveys	Benefits not quantifiable on a single project. Existing database of project achievements including productivity, quality, tool/method data. Projects with experience of using the method/tool.
Feature Analysis - Screening mode	Large number of methods/tools to assess. Short timescales for evaluation exercise.
Feature Analysis	Benefits difficult to quantify.

Case Study	Benefits observable on a single project. Stable development procedures. Tool/method user population limited. Timescales for evaluation commensurate with the elapsed time of your normal size projects.
Feature Analysis Experiment	Benefits difficult to quantify. Benefits directly observable from task output. Relatively small learning time. Tool/method user population very varied.
Feature Analysis Survey	Benefits difficult to quantify. Tool/method user population very varied. Benefits not observable on a single project. Projects with experience of using the method/tool, or projects prepared to learn about the method/tool.
Qualitative Effects Analysis	Availability of expert opinion assessments of methods/tools. Lack of stable development procedures. Requirement to mix and match methods/tool. Interest in evaluation of generic methods/tools.
Benchmarking	Method/tool not human-intensive. Outputs of method/tool able to be ranked in terms of some "goodness" criteria.

4.2 Practical issues

In principle, you should select the simplest evaluation method appropriate for your specific evaluation requirements and organisational capability. However there are a number of constraints that can influence your final choice of evaluation method:

- the *elapsed time* that is needed for the different evaluation options;
- the *confidenc*e that a user can have in the results of an evaluation;
- the *cost* of an evaluation;

This information may be useful in two circumstances:

1. When you want to decide between a quantitative case study and a quantitative formal experiment and there are no clear technical reasons for preferring one to the other.
2. When you want to ensure that the evaluation you have decided to undertake is feasible.

These issues are discussed in this section.

4.2.1 Evaluation Timescales

The DESMET method ranks the nine evaluation methods in order of the timescales likely to be needed to perform an evaluation, as shown in Table 2. You should ensure that your technical choice of evaluation method conforms with your external timescale constraints.

Table 2 The relative timescales associated with each evaluation method

Relative Timescales	Evaluation Method	Comments
Long (3 plus months)	Case study (Quantitative & Feature Analysis)	A case study may last for the entire duration of a project.
Medium (several months)	Feature Analysis Survey	Surveys of tool/users take time for questionnaires to be circulated and returned.
Short (Several weeks)	Experiment (Quantitative & Feature Analysis)	Experiments usually need to be constrained to relatively small well-defined tasks that can be done in a short time or they are unlikely to be viable. Staff will not be available for long experiments, and extended tasks are difficult to control properly.
	Benchmarking	Developing appropriate tests will take some time. Thereafter running the tests should be relatively straightforward (since we assume that benchmarks are used for method/tools that are

Relative Timescales	Evaluation Method	Comments
		not human-intensive).
	Feature analysis -screening mode	Time will be required to assess the features that will be the basis of the evaluation and to collate material about a number of different methods and tools. An actual evaluation based on review of marketing and technical documentation rather than trials of the tools/methods themselves need not take very long
Very short (a few days)	Quantitative Survey	Assuming a database of quantitative data is available extracting and analysing data about a specific method/tool should not take very long.
	Qualitative Effects Analysis	Assuming a database of expert opinion about method/tool impacts is available, extracting data about a specific method/tool should not take very long.

4.2.2 Confidence in results

There is always a risk that an evaluation results in an incorrect conclusion about the method/tool being evaluated. This risk has two elements:

1. The results may imply that a method or tool is not beneficial when it is beneficial (false negative).
2. The results may imply that a method or tool is beneficial when it is not (false positive).

The extent of the risk differs for the different methods. You should ensure that the evaluation method which you have been advised to use will give sufficient confidence that the results of the evaluation are correct. In this context, "sufficient" must be assessed in relation to the potential cost of an incorrect decision. An incorrect decision would be regarded as serious if a large investment decision were affected, or the method/tool were to be used to produce safety-critical applications. The different methods can be ranked as follows with respect to the risk that they will deliver incorrect results as shown in Table 3.

Table 3 The relative risk associated with each evaluation method

Relative risk	Evaluation Method	Comments
Very High	Qualitative Effects Analysis	This is based on subjective view of a single expert, so has a very high risk of giving a misleading result.
	Feature analysis - Screening mode	The evaluation is based on one person's evaluation of third party information and the evaluation criteria are subjective.
High	Quantitative Case study with cross-project baseline	The evaluation of impact is based on a comparison of two different projects. No statistical analysis is possible, so there is a high risk of a misleading result.
	Feature Analysis Case study	The evaluation is based on one person's experience of using the method/tool and the evaluation criteria are subjective.
Medium	Quantitative Case Study with organisation baseline	The evaluation of impact can use information about the average and scatter of `normal' projects. This gives a better chance of a valid result.
	Feature Analysis Survey	The evaluation is based on the practical experience of a number of projects and staff.
Low	Quantitative Case Study with within-project baselines	Replication within the case study project allows statistical methods to be used. However, there is still a risk that the case study project had unusual properties so could give a misleading result.
	Feature Analysis Experiment	Replication and randomisation reduce the risk of systematic bias, but the evaluations are still subjective.
	Quantitative Survey	Replication across projects allows statistical methods to be used to assess impact. However, results will only show association not causality. Thus, the results may be due to some unknown factor rather than the

Relative risk	Evaluation Method	Comments
		method/tool being investigated.
Very Low	Quantitative Experiment	The whole of the "scientific method" supports the reliability formal experiments. In addition, the probability of incorrect results is itself quantifiable when formal experimental procedures are followed.

4.2.3 Costs of an evaluation

In principle, evaluation costs should be proportional to the consequential benefits if the method or tool were judged beneficial and introduced, less the expected investment and introduction costs. Evaluation costs should therefore, as a rule of thumb, be no more than 10% to 20% of expected net benefits. However, the costs of not evaluating and making a bad choice may be more than just the loss of the potential benefits since consequential loss may ensue!

As an initial crude guideline, the relative costs of the various evaluation methods (allowing for the constraints on method selection) are shown in Table 4.

In order to assess the actual costs of a specific evaluation, you must consider the factors that affect the costs of each evaluation type. These cost factors are identified in the following paragraphs.

Table 4 The relative cost associated with each evaluation method

Relative cost	Evaluation Method	Comments
High	Experiment	The cost of experiments is due to a number of different staff undertaking what may be non-revenue-producing work. (Note. An academic institution may find experiments less costly if they are performed as part of student course work.)
Medium	Case Study	A case study involves staff reporting on the use of a method/tool. Costs will relate to training and additional work undertaken for evaluation rather than project purposes.
	Feature Analysis Survey	Planning and running a survey can take a good deal of effort, and each respondent will have to take the time to understand and complete the survey questionnaire.
	Benchmarking	The main staff effort associated with benchmarking is designing the benchmark tests and running them on each method/tool of interest.
	Feature Analysis - Screening mode	This would be done by a single person in a matter of weeks. The cost would increase if it were necessary to undertake a detailed requirements specification of the potential method/tool users requirements
Low	Quantitative Survey	Assuming a database of quantitative information is available, analysing the data to evaluate the impact of a method/tool involves relatively little effort although data analysis might require statistical expertise. Note. Establishing such a database in the first place would require substantial investment.
Very Low	Qualitative Effects Analysis	Assuming a database of expert opinion about the impact of methods/tools is available, extracting the information requires relatively little effort. Note. Establishing such a database in the first place would require substantial investment.

4.2.3.1 Costs of Feature Analysis

The three different types of feature analysis have a similar cost basis. The difference will be related to the number of evaluators taking part.

Staff costs arise from the effort required to:

- define the requirements of the user population

- design and produce a questionnaire;
- devise a scoring method;
- familiarise the evaluator(s) with tool/method;
- complete the Feature Analysis questionnaire (summed for all evaluators);
- collate results;
- produce evaluation report.

In addition, when the tools/methods are being used in the evaluation, direct costs arise from:
- licensing/buying/hiring computer equipment to run the tools;
- licensing/buying/hiring the methods and tools themselves;
- training and support/consultancy.

These costs are usually avoided during initial screening where the evaluator is likely to use marketing and technical documentation rather than trials of the methods/tools themselves.

4.2.3.2 Costs of Qualitative Effects Analysis

If the method being investigated is already included in the Qualitative Effects Analysis knowledge base, the costs of an evaluation will be restricted to the staff effort needed to:
- run the evaluation;
- prepare an evaluation report.

If the method does not exist in the knowledge base, the costs of the evaluation will include additional staff effort to:
- elicit knowledge about the method;
- update the knowledge base.

In addition, there may be additional consultancy costs if knowledge elicitation requires access to external experts.

4.2.3.3 Costs of Benchmarking

The costs of a benchmarking exercise include the initial staff effort required to design and construct the specific tests that will be used (e.g. for a compiler this might be a set of source code seeded with syntactic errors of different types). Then for each tool evaluated using the benchmark, there will be staff effort to:
- run the benchmarks;
- analyse benchmark results;
- report on the results.

Initial costs may be high and timescales lengthy to develop the benchmark(s), but after that the actual evaluation cost per tool is relatively low.

4.2.3.4 Costs of a Quantitative Case Study

The evaluation costs are based on the staff effort required to:
plan the evaluation;
- extend the project plan of an existing project to integrate case study requirements;
- collect data required for the case study. (The amount of effort is inversely proportional to the extent of existing data collection procedures);
- analyse the results;
- prepare an evaluation report.

For the methods and tools direct costs arise from:
- licensing/buying/hiring computer equipment to run the tools;
- licensing/buying/hiring the methods and tools themselves;
- training and support/consultancy.

4.2.3.5 Costs of a Quantitative Formal Experiment

The evaluation costs are based on the staff effort required to:
plan the evaluation, including design of the experiment;
take part in the experiment (for each subject);
analyse the results;
prepare an evaluation report.

For the methods and tools direct costs arise from:
- licensing/buying/hiring computer equipment to run the tools;
- licensing/buying/hiring the methods and tools themselves;
- training and support/consultancy.

4.2.3.6 Costs of a Quantitative Survey

Assuming that a database of quantitative project data already exists, the costs of performing a quantitative survey are based primarily the staff effort required to:
- extract and analyse the data;
- prepare an evaluation report.

4.2.4 External Evaluation

If the evaluation method that is most appropriate is not viable given other constraints, you might consider the option of having an evaluation performed externally. Conditions when an organisation might opt for an external evaluation are:
- When shortage of human resources (not finance) prevents an internal evaluation from taking place.
- For very novel methods or tools, when there is little internal knowledge about the methods/tools. Here, an academic evaluation might be feasible.
- When information about methods/tools is available external to your own organisation.

In the latter case, you might be able to organise a survey of results in other organisations directly, rather than using an external agency to undertake the evaluation. However, it is difficult for a commercial organisation to organise a survey of other commercial organisations, particularly if quantitative information is requested. Under these circumstances it would be better:
- to approach an independent body to obtain qualitative data;
- to approach an organisation that maintains a large-scale database of quantitative information.

However, external evaluation of any sort is likely to be a costly and risky method of evaluation. One risk is that the results may not generalise to your organisation. There is always the likelihood that one organisation's experience of a method or tool will not be replicated in another organisation, so the results of an external evaluation might not apply in the commissioning organisation. Another risk is that the basic evaluation is flawed by poor data. Large-scale cross-company databases may not have very reliable data. Definitions of metrics such as “lines of code”, “function points”, “defects per 1000 lines of code” are very difficult to standardise among different organisations.

5. The influence of human factors

This section was prepared in collaboration with Chris Sadler of University of North London.

5.1 Introduction

In the previous section, I described the range of methods available if you want to evaluate a software engineering method/tool and the criteria you need to consider to select a method appropriate to your individual circumstances. Before discussing some of the evaluation methods in more detail, this section will review some of the human factors issues that can affect an evaluation exercise.

5.2 Impact of human factors

The attitude and motivation of participants in an evaluation exercise can distort its results. In general, *distortion effects* come about because the behaviour of staff participating in an evaluation exercise is *at variance with their normal day-to-day working practices*. This implies that productivity or quality measurements obtained from evaluation exercises may not be realistic and so can not be reliably generalised or scaled-up. Thus, the success of a specific evaluation project depends on recognising any possible causes of distortion effects and taking appropriate steps during evaluation project planning to avoid or minimise them.

Distortion effects can go either way: excessive enthusiasm or a desire to make a positive impression can lead to an over-optimistic assessment; hostility, anxiety or unfamiliarity can lead to unduly pessimistic outcomes. These issues are usually less of a problem for surveys which are based on the results obtained from a number of different projects with little or no intrusion into the projects by the evaluator. However you should expect problems running formal experiments where scale-up is usually critical or case studies where evaluations are based on single projects.

Case study evaluations cause additional problems because they centre on observations obtained from a "typical" production project. This can cause a conflict of goals between the evaluation project and the production project(s) that host the evaluation exercise[4]. For example, the need to adopt unfamiliar practices and to make measurements, as required for the evaluation, may detract from the pilot project's goals to deliver on time and to budget. This might lead to the evaluation being abandoned in order to rescue the pilot project, or the pilot project delivering a sub-optimal product. Either way no valid conclusions can be drawn about the method/tool under investigation. Thus, it is important that the relationship between the evaluation exercise and the pilot project are properly defined. At the minimum you need to ensure:

- An evaluation exercise is managed separately from the pilot project(s) with a separately administered budget.
- Successful completion of the evaluation project is a formal success criterion for the pilot project managers as well as the evaluation exercise manager.
- The pilot project should take priority over the evaluation exercise. If the method or tool is evaluated on a project of such low priority that managers do not care whether it succeeds or not, the results of the evaluation are likely to be biased.
- The roles and responsibilities of the evaluation manager and the pilot project manager(s) is defined and understood.
- The roles and responsibilities of project staff with respect to the evaluation management and the pilot project management is defined and understood.

5.3 Sociological effects

When introducing a new method or tool, you should expect the behaviour of the project participants to be affected. Indeed the goal of the evaluation project is to assess the nature and degree of those effects. However, it is important to distinguish those effects that are *inherent* in the application of the method/tool, from those sociological effects which arise from the *novelty* brought about by the evaluation project or the *expectations* that have been placed upon it.

5.3.1 Novelty Effects

Novelty effects arise when the subjects' behaviour is changed because of what they are required to do or because the situation they find themselves in is unfamiliar. The two major novelty effects are:

1. The learning curve effect.
2. The Hawthorne effect.

5.3.1.1 Learning curve effect

The learning curve is the term given to the observation that people develop a facility, and ultimately an expertise, with methods/tools as they gain familiarity with their application. Initially, therefore, they are likely to use new methods/tools more clumsily or ineffectively than they might after a period of familiarisation. Thus the learning curve effect will tend to counteract any positive effects inherent in the new method/tool.

In the context of evaluating methods/tools there are two basic strategies to minimise the learning curve effect (that are not mutually exclusive):
1. Provide appropriate training before undertaking an evaluation exercise;
2. Separate pilot projects aimed at gaining experience of using a method/tool from pilot projects that are part of an evaluation exercise.

5.3.1.2 The Hawthorne effect

When an evaluation is performed, staff working on the pilot project(s) may have the perception that they are working under more management scrutiny than normal and may therefore work more conscientiously. Industrial psychologists call this the Hawthorne effect because it was first observed during a series if studies at the Hawthorne aircraft factory in the 1930s [7]. Factory workers were observed to perform better when the factory lighting was improved. However, when the lighting was reduced again, the performance of the factory workers again improved. Their performance was not primarily linked to lighting levels but to the attention which management was giving them in conducting the experiments. Thus, the Hawthorn effect would tend to exaggerate positive effects inherent in a new method/tool.

[4] In order to avoid over-using the word "project", we shall refer to production projects that host an evaluation exercise as pilot projects and the evaluation exercise as the evaluation project.

A strategy to minimise the Hawthorne effect is to ensure that a similar level of management scrutiny is applied to control projects in your case study (i.e. project(s) using the current method/tool) as is applied to the projects that are using the new method/tool.

5.3.2 Expectation effects

The behaviour of evaluation project participants can be modified by virtue of the expectations about the method/tool they hold themselves or by the expectations of the people responsible for the evaluation. There are two main types of expectation effect:

1. The placebo effect
2. The doctor effect.

5.3.2.1 The placebo effect

In medical research, some patients who are deliberately given ineffectual treatments (i.e. placebos) recover if they believe that the treatment will cure them. Likewise a software engineer who believes that adopting some practice (for example, wearing a baseball cap) will improve the reliability of his code may succeed in producing more reliable code. However, such a result could not be generalised because there would be no guarantee that the specific believe was universally held, nor that the believe itself is sufficient to modify the behaviour of all software engineers.

In medicine, researchers minimise the placebo effect by running *blind* experiments where subjects are not informed whether they are in the trial group (receiving the "real" treatment, or the control group who receive the placebo). This works in drug trials because in this situation subjects are passive recipients of the treatment. This is not the case in the context of method/tool evaluations.

When evaluating methods and tools the best you can do is to:
- Assign staff to pilot projects using your normal project staffing methods and hope that the actual selection of staff includes the normal mix of enthusiasts, cynics and no-hopers that normally comprise your project teams.
- Make a special effort to avoid staffing pilot projects with staff who have a vested interest in the method/tool (i.e. staff who developed or championed it) or a vested interest in seeing it fail (i.e. staff who really hate change rather than just resent it).

This is a bit like selecting a jury. Initially the selection of potential jurors is at random, but there is additional screening to avoid jurors with identifiable bias.

5.3.2.2 The doctor effect

In a blind experiment, patients may infer which group they belong to because those administrating the treatments convey their expectations about who should benefit and who should not. To counter this effect, double blind experiments are conducted where neither the doctor nor the patient knows which treatment is given to whom.

There are two particular areas where a doctor-type effects can manifest themselves in project-based evaluation exercises:

Unbalanced target effect. In order to establish the nature and degree of any inherent effect of the new method or tool, you will need to make some initial measurements. If your projects do not usually use software measures this can give rise to a variety of novelty effects, but in any case, measures are likely to result in *explicit* project targets related to product size, defect rate or development effort. Pilot project staff will then strive to achieve the explicit targets even if it means trading-off against other *implicit* targets such as documentation quality. Any targets highlighted by the measurement process give a clear, if unintentional, indicator of expectation.

Intervention effects. When you perform an evaluation exercise you will need staff to act as investigators as well as the pilot project staff who act as subjects. The investigators can easily compromise and evaluation exercise by encouraging pilot project staff to modify normal working practices or by conveying their expectations about the final outcome of the evaluation exercise.

The approaches you can take to reduce these effects are to:
- Ensure that an evaluation exercise includes controls for example projects using the old technology and that the controls are treated in the same was as the projects using the new technology. That is the controls also have explicit targets based on measurements and must interface with the evaluation team.
- Ensure your evaluators are not particularly enthusiastic or cynical about the new technology being evaluated. They need to be motivated by the requirements to ensure the successful completion of the evaluation exercise not the adoption or not of the technology. Often companies have *technology champions* who assess and

promote new technology. A champion in this sense should not be in charge of an evaluation exercise he/she will have too much vested interest in specific "pet" technologies. An evaluation needs a *process champion* who is not concerned with particular technology but with ensuring the best overall development process for the organisation. A process champion should be as concerned to avoid bad process changes (i.e. non-beneficial or unnecessary changes) as to adopt good process changes.

5.3.3 Summary

This section has identified a number of human factors and sociological issues that can distort the results of an evaluation exercise. When you undertake an evaluation exercise, you need to be sure that you have identified all such effects and put in place specific processes to minimise their impact.

6. Principles of Feature Analysis

This section was prepared in collaboration with Lindsay Jones

6.1 Introduction

This section discusses the principles of feature analysis and describes how to identify features and establish a means of scoring products against those features. It also considers how to plan a feature-based evaluation and how to analyse the results of an evaluation.

In its simplest form, feature analysis provides a list of "yes/no" responses to the existence of a particular property in a product. If, for example, you were trying to decide which personal computer to buy, you might list all the properties that you believed to be requirements of the computer and then allocate a "tick" or "cross" for each property for each candidate PC, according to whether it possessed that property. You would then count the number of ticks that each candidate had received. Those with the highest counts would provide you with a short list of candidates on which to carry out a more detailed evaluation to decide their relative value for money or some other criteria for finally deciding which one to buy.

When evaluating products such as software packages, the requirements become "fuzzier" (consider package "usability" for example), and the products tend to "conform" in degrees to each requirement. In such cases you need a means of assessing this degree of conformance as objectively as possible so you can include the information in the evaluation process. This "fuzziness" of requirements applies to most of the properties of software methods and tools.

The objective of the evaluation is usually to provide input into a decision about whether to purchase a method or tool for use by the organisation. The method or tool will normally be intended to carry a specific purpose, thus, the results of the evaluation need to provide information in the following areas:

1. *Suitability for purpose* - will it do the job we want it to?
2. *Economic issues* - can we afford it?
3. *Drawbacks* - are there any *gotchas*? Is there any aspect that makes the tool or method less attractive though it does the job?
4. *Other advantages* - these are things other than the purpose for which the tool or method is intended but make the tool more attractive; e.g., it helps get into another market.

A feature analysis type of evaluation is an attempt to put rationale and structure behind a "gut feeling" for the right product. Therefore an important part of carrying out the evaluation is to:

- help clarify the important features of the method or tool in the context of the organisation and the environment;
- help identify the differences between the methods and tools;
- provide an explanation of why a decision was made - an audit trail, as it were.

The main processes involved in carrying out a feature analysis are, therefore, to:

1. Select a set of candidate method/tools to evaluate.
2. Decide the required properties or features of the item being evaluated.
3. Prioritise those properties or features with respect to the requirements of the method/tool users.
4. Decide the level of confidence that is required in the results and therefore select the level of rigour required of the feature analysis.
5. Agree on a scoring/ranking system that can be applied to all the features.

6. Allocate the responsibilities for carrying out the actual feature evaluation.
7. Carry out the evaluation to determine how well the products being evaluated meet the criteria that have been set.
8. Analyse and interpret the results.
9. Present the results to the appropriate decision-makers.

6.2 Features of feature analysis

6.2.1 Iterative procedures

Feature analysis is a shortlisting activity that may be performed iteratively. Each iteration is intended to reduce the number of candidate methods/tools. Different iterations may vary the set of features being assessed, the individual making the assessment, or the evaluation criteria. The starting point and the number of iterations depend upon the number of candidate methods and tools and the amount of effort available for performing the activity. It is unlikely that the number of iterations can be fixed before starting the evaluation, as it will depend on the outcome of each.

6.2.2 Comparative Framework

A common framework is necessary to make any sort of comparative evaluation of methods and tools. Feature analysis allows the framework to be expressed in terms of a set of common (mandatory and/or desirable) properties, qualities, attributes, characteristics or features (terms which are often used interchangeably) for each type of method or tool. Choosing a set of attributes or features is essentially a subjective activity and depends on the experience, viewpoints, and interests of the individuals or groups concerned.

6.2.3 Subjective Judgement Scales

A method or tool has to be judged as to how much support it actually appears to provide for a feature, i.e., to what degree support is present. You will usually need to devise a scale for assessing the degree of support a method/tool provides for a specific feature. This assumes that those methods and tools that do possess a feature, or score highly on a support scale for it, are "good" and that those that do not are "less good".

6.2.4 Method/tool assessment

Once each method or tool has been "scored" for each feature in the framework using some common judgement scale, the results for the methods or tools have to be compared to decide their relative order of merit. Combining the scores of multiple features (in whatever way) must be treated with caution, because many different combinations of numbers can produce the same aggregate score. More direct means of comparison are often useful such as comparing the scores obtained for key features across candidates.

Ultimately, almost every manager wants a single figure of merit. Although it is possible to produce an aggregate score, you must exercise caution when providing and interpreting such numbers.

6.2.5 Feature complexity

Features chosen for an evaluation may be quite complex concepts in themselves - again consider what notions "usability" might cover. They may be decomposed into sub-features which are conceptually simpler items, and the sub-features further decomposed. Unfortunately, there is no rule which says when to stop decomposing, so it is easy to generate a large number of features.

6.2.6 Advantages of feature analysis

Feature analysis is an extremely flexible method of method/tool evaluations:

1. It does not impose many pre-requisites upon the infrastructure of an organisation wanting to use the method (for example you do not need to have a measurement programme in place);
2. It can be done to any required level of detail. Simple screening evaluations can be done quickly and cheaply, but it is always possible to perform more detailed evaluation to improve the reliability of an evaluation exercise.
3. It can be applied to any type of method or tool from Word processors to Lifecycles. This is particularly important if you want to evaluate integrated toolsets or complex methods. Quantitative evaluation methods are usually inappropriate for evaluating the effect of multiple process changes.

In addition, feature analysis also has the major advantage that it is not restricted to technical evaluations, acquisition issues such as the viability of the suppler can also be evaluated. In fact, in the case of a major

acquisition exercise, you would almost certainly need to perform a feature analysis as well as any more objective quantitative evaluation exercise.

6.2.7 Limitations of feature analysis

Feature analysis can be useful in carrying out shortlisting activities and for undertaking evaluations when no process metrics are available to undertake quantitative evaluations on the effects of using such methods and tools. However, feature analysis does have several significant drawbacks. This section discusses these limitations.

6.2.7.1 Subjectivity

Feature analysis is been based on judging methods against some "evaluation criteria" which are identified subjectively. Such evaluations are therefore likely to be biased and context dependent.

6.2.7.2 Inconsistency

There is also a problem of inconsistency in scoring between different assessors. If different assessors evaluate different tools, they may have different degrees of familiarity with and understanding of the method/tool. In addition, they may interpret the scale for a particular feature in different ways.

6.2.7.3 Collating Scores

Producing a set of scores for all the features for a specific method or tool is, unfortunately, not the end of the story. The various score have to be collated and compared to decide the relative order of merit of the methods or tools.

Producing a single number from the individual scores (e.g. by some arithmetic combination formula such as weighted ranking) may be misleading because many different combinations of numbers can produce the same aggregate score. Furthermore, certain features may attract higher average scores than others because an assessor may understand them better and be more able to recognise support in the method or tool. There are also deeper reasons (connected with the nature of the ordinal scales that are usually used to assess features - for instance, a score of 4 is not necessarily twice as good as a score of 2) why you need to be very careful assessing the meaning of scores if they are combined into single numbers.

6.2.7.4 Generating too many Features

Another problem of feature analysis is that hundreds of features may be identified. This makes performing an assessment of a specific method/tool very time consuming and makes it difficult to analyse all the scores. You need to strike a balance between the depth of understanding required to achieve the desired level of confidence in the evaluation and the practical difficulties in handling large numbers of features.

6.3 Identifying and Scoring Features

6.3.1 Deriving feature lists

6.3.1.1 General Principles

There are several approaches to generating sets of features for a particular evaluation exercise. They include:

1. Treating the system development process as a system in its own right and analysing it using the standard techniques of system analysis to obtain a model of the tasks, processes, data (documents) and interactions involved. A method or tool can then be assessed for its support for features of this software process model. This approach is particularly useful if you need to identify a set of methods/tools to support your entire development process.
2. Taking an expert view distilled from experience or based on the theories of computer science to define the key features of a method: for example, effective ways of managing complexity, techniques for locating the system boundary, design based on the principle of levels of abstraction, or having an underlying mathematical formalism.

In most significant evaluation exercises you will need to use both approaches. Initially you will need to make some assessment of the technology you are interested in to identify an initial feature list. You will then need to refine the feature list and determine how the features should be assessed. This should be done in consultation with potential users of the technology.

For an important investment decision you should expect to:

1. Identify any user groups likely to have different requirements. For example, for general IT technology such as PC software you might identify users groups as administrative staff, managers, and technical staff, For software engineering technology you might identify user groups as systems analysts, designers, developers, testers, quality engineers.
2. Review your initial feature list with representatives of each user group to compose a refined feature list that addresses all their requirements.
3. Review the importance and support requirements for each feature in your refined feature list with representatives of each users group.
4. Identify the an acceptable score for each feature allowing for different requirements for different requirements for each distinct user population group.

6.3.1.2 Preparing an initial feature list

It is important that you assess a balanced set of features, not only looking at the *technical* aspects, but also the *economic*, *cultural* and *quality* aspects. Bear this in mind when compiling your list.

A specific feature may be further decomposed into sub-features, and sub-features further decomposed into conceptually simpler items. There is no rule for ceasing decomposition, except that the greater the number of sub-features, the greater the investment required in the evaluation, and the more difficult it becomes to assess the results. (You could apply a pragmatic cut-off rule - if you cannot present the results on one sheet of paper, then the list of features is too large.)

When you decompose features into subfeatures and subsubfeatures, you are developing a hierarchy of features. A set of features at the same level of decomposition and in the same part of the hierarchy can be assessed together as a *feature set*. You will need to keep track of how feature sets are derived to assist you analysis of the evaluation results.

A top level feature set including non-technical as well as technical features might be:
1. *Supplier* assessment;
2. *Maturity* of method or tool;
3. *Economic* issues in terms of purchase, technology transfer costs and cost of ownership;
4. *Ease of introduction* in terms of cultural, social and technical problems;
5. *Eligibility* for required application areas;
6. *Reliability* of the tool software;
7. *Robustness* against erroneous use;
8. *Effectiveness* in current working environment;
9. *Efficiency* in terms of resource usage;
10. *Elegance* in the way certain problem areas are handled;
11. *Usability* from the viewpoint of all the target users in terms of learning requirements and “user-friendliness”;
12. *Maintainability* of the tool;
13. *Compatibility* of the method and tool with existing or proposed methods and tools.

Level 2 of the feature hierarchy are created by expanding each top level feature. For example, “Usability” could be expanded into two features: “Learnability” and “Graphical User Interface Features”. The second level features can be further expanded to produce level 3 features if necessary. For example Learnability might be further expanded into the following features:
1. Quality of documentation;
2. Learning curve;
3. Training effort;
4. Training quality;
5. On-line help.

6.3.2 Assessment scales for scoring features

6.3.2.1 Feature Types

There are two types of feature:
1. Simple features that are either present or absent. These are assessed by a simple YES/NO nominal scale;
2. Compound features where the degree of support offered by the method/tool must be measured or judged on an ordinal scale.

Each simple feature can be accompanied by a degree of *importanc*e assessment. Each compound feature must be accompanied by an assessment of its importance and an appropriate judgement scale associated with the degree

of existence or *conformance* to a particular feature or characteristic. Scales for measuring importance and conformance are discussed below.

6.3.2.2 Importance

A good method or tool is one that includes the features that are most important to its users. The importance of a feature can be assessed by considering whether it is mandatory or only desirable. A method/or tool that does not possess a mandatory feature is, by definition, unacceptable (for example a word processor that did not have an spell-checker feature). Non-mandatory features are classed as "desirable" (for example a thesaurus on a word processor). Coverage of non-mandatory features allows you to judge the relative order of merit of a group of otherwise acceptable methods/tools.

This view of importance leads to two assessment criteria: one of which identifies whether or not a feature is mandatory; the other of which assesses the extent to which a non-mandatory feature is desired. There may be many gradations of "desirability", but for practical purposes, we recommend restricting yourself to no more than 3. For assessing a feature these two criteria can be considered as one ordinal scale with following scale points:

M Mandatory
HD Highly Desirable
D Desirable
N Nice to have - "icing on the cake".

6.3.2.3 Conformance

A judgement scale for conformance performs two distinct purposes:
1. It defines what level of support of a particular feature is required.
2. It provides the assessor with a consistent measurement scale against which to "score" the feature of a particular candidate.

When assessment of the user population support requirements, or evaluation candidate methods/tools, is to be performed by persons other than those who assembled the feature list, it is usually necessary to define the scale points explicitly for each features. Since evaluation of judgement scales is subjective, the more accurately the scale points are defined, the more likely it is that different assessors will use the scales consistently.

When defining the meaning of the scale points for a particular feature, you must take care to ensure that
- there is minimum overlap between the descriptions of each point;
- the scale covers the complete range of possibilities.

The granularity of the scale points depends upon the feature that is being looked at, and your actual requirements. Table 5 is an example of a "generic" judgement scale for tool support for a particular facility. The table shows the generic definition of the scale point that applies to all compound features, a more detailed (but still reasonably generic) definition of what the scale point means in terms of defining the level of support a tool provides for a specific feature and a mapping from the scale point to the integers.

Table 5 Example judgement scale to assess tool support for a feature

Generic scale point	Definition of Scale point	Scale Point Mapping
Makes things worse	Cause Confusion. The way the feature is implemented makes it difficult to use and/or encouraged incorrect use of the feature	-1
No support	Fails to recognise it. The feature is not supported nor referred to in the user manual	0
Little support	The feature is supported indirectly, for example by the use of other tool features in non-standard combinations.	1
Some support	The feature appears explicitly in the feature list of the tools and user manual. However, some aspects of feature use are not catered for.	2
Strong support	The feature appears explicitly in the feature list of the tools and user manual. All aspects of the feature are covered but use of the feature depends on the expertise of the user	3
Very strong support	The feature appears explicitly in the feature list of the tools and user manual. All aspects of the feature are covered and the tool provides tailored dialogue boxes to assist the user.	4

Full support	The feature appears explicitly in the feature list of the tools and user manual. All aspects of the feature are covered and the tool provides user scenarios to assist the user such as "Wizards".	5

6.3.2.4 Acceptance Thresholds and Criteria

Feature analysis is intended to help you decide whether or not a specific method/tool meets the requirements of its potential users. Representatives of the user population(s) need to assess the importance of each feature and assess their support requirements (i.e. *the acceptance threshold*) for each compound feature. You need to collate their requirements into an overall *acceptance criteria* that you can use to assess the order of merit of each method/tool being evaluated.

For example, you might decide on a two-stage acceptance criteria as follows:

Firstly, you would assess all candidate methods/tools for basic acceptability. You might decide that a method/tool will be acceptable if:

- all simple mandatory requirements are met;
- all compound mandatory requirements meet their required level (i.e. acceptance threshold);

Secondly, if that acceptance criterion resulted in a number of methods/tools being judged acceptable, you might decide to base your final choice of a specific method/tool on:

- the number and importance of other simple features supported;
- the number and importance of other compound features that meet their acceptance level.

The acceptance criteria might need to be refined to allow consideration of:

1. the different levels of "desirability" of the different features;
2. the extent to which each method/tool provides coverage of the different high level factors;
3. the extent to which the different requirements of different user groups impacts the assessment of the order of merit of the methods/tool.

In a screening mode evaluation, it is usual for the evaluator to derive acceptance criteria either by assessing the importance of each features from the viewpoint of the user population and setting a minimum acceptable value (i.e. 2) for all compound features. For other types of feature analysis, the evaluator needs to elicit the acceptance threshold information from representative users.

6.3.3 Documentation

The techniques described above lead to the development of two documents that should be part of any evaluation exercise

6.3.3.1 The Evaluation Criteria

This document comprises:

- A list of all the required features, along with their definitions.
- For each compound feature, a judgement scale with appropriate levels of importance.
- For each user group:
  - an assessment of the importance of each feature
  - for each compound feature, the minimum acceptable level of support required (i.e. the acceptance threshold)

The purpose of the document is to enable you to analyse the results of the evaluation against your requirements.

6.3.3.2 The Score Sheet

This comprises:

- A list of all the simple features.
- Every compound feature that is to be included along with the judgement scale - but without the level of importance. (If you want the evaluators to be objective it is preferable not to tell them what the required conformance level is. N.B. If the evaluation organiser and the method/tool assessor are the same person, the method/tool assessment may be influenced by the level of importance.)
- A space for comment with every feature. These comments may lead to the development of further simple or compound features. Simple scores on their own are not usually enough.

6.4 Planning a Feature Analysis Evaluation

When planning a feature analysis based evaluation of a method or tool the following areas should be addressed and included in the evaluation project plan:

1. Scope of the evaluation
2. Basis of the evaluation
3. Roles and responsibilities
4. Procedures
5. Assumptions and Constraints made
6. Timescale and effort involved.

These are discussed in the following sections.

6.4.1 Scope

The scope of the evaluation should be defined and should state the purpose of the evaluation. There are two main reasons why evaluations are undertaken:

1. To select a tool to carry out a specific function in the organisation, that is, where there is a specific requirement. A special case of this is selecting a tool to support an already selected method.
2. To undertake a market survey to establish product capability and availability where there is no specific requirement.

This article concentrates on assisting evaluations to carry out the former, although the underlying process is the same in both cases. The major difference is that it is more difficult to address the notion of conformance of specific features of a product when there are no specific requirements.

In addition, you need identify:

1. the starting point of the evaluation: either the list of candidate tools for evaluation, or how the candidate list should be established (this is discussed in the next article);
2. the specific software engineering functions that require support;
3. expected scope of usage of the selected tool; (which type of projects, which type of staff);
4. how the results of the evaluation will be used and by whom.

6.4.2 Basis of evaluation

The depth of investigation required for the candidates will be linked to the level of confidence required in the evaluations. It may vary as follows:

- survey of publicity literature;
- attendance at organised demonstrations or trade fairs;
- study of published evaluations in the technical literature;
- study of technical documentation;
- interviews of current users;
- detailed evaluation of technical documentation plus “hands-on” usage of demonstration or evaluation versions of software (if appropriate);
- practical experience of using the candidate method or tool in a wide range of situations.

6.4.3 Roles and responsibilities

There are likely to be four major roles in an evaluation exercise (although some may be performed by the same person). They are:

1. The sponsor.
2. The evaluator (or evaluation team)
3. The technology user(s)
4. The method/tool assessors.

6.4.3.1 The sponsor

This is the person who wants to know what purchasing decision to make given a requirement for investment in some technology. He/she is responsible for:

- setting the context for the evaluation;
- identifying any constraints (i.e. the timescales within which the results are required);
- providing the resources for performing the evaluation;
- receiving and acting on the evaluation report.

6.4.3.2 The evaluator

This is the person or team responsible for running the evaluation exercise. He/she is responsible for:

- preparing the evaluation plan
- identifying candidate method/tools;
- identify each distinct group in the potential user population;
- eliciting the requirements of each user population group;
- identifying the features to be assessed and the method by which they are scored;
- organising the assessment whereby each of the method(s)/tool(s) is scored;
- collation and analyse of the scores;
- preparation of the evaluation report.

6.4.3.3 The technology users

These are the people who will use the method/tool chosen as a result of the evaluation exercise. The user population for method/tools can be a mixed population (i.e. be composed of groups that have different requirements). The evaluator needs to identify each group in a mixed user population and extract the requirements of each group.

Representatives of each group will need to:

- discuss their technology requirements with the evaluator;
- review the features identified by the evaluator for completeness with respect to their requirements
- assess the importance and support required for each of the different features

6.4.3.4 The method/tool assessors

These are the people who will score each feature for the candidate methods/tools. They are responsible for reviewing/using specific methods or tools in accordance with the defined evaluation procedure. Depending on the type of evaluation undertaken, the method/tool assessor(s) can be

- the evaluator (or evaluation team);
- representatives of the technology users;
- current users of the methods/tools.

6.4.4 Evaluation procedures

You should also define how the evaluations are to be carried out. This includes:

- how candidate method/tools will be identified;
- how the feature-sets and criteria of rejection or acceptability will be derived;
- how the results will be arrived at and presented;
- how the evaluations are to be organised.

It is necessary to decide the level of confidence required in the results of the evaluation and how that confidence is to be attained. This is often linked to budgetary considerations, and to the criticality of the success of the selection.

There is are four ways of organising evaluations:

1. Screening mode
2. Case Study
3. Formal Experiment
4. Survey

6.4.4.1 Screening mode Approach

This is the cheapest but least reliable form of Feature Analysis. The evaluation is performed by a single person based on documentation only. It is the best approach for screening a large number of methods/tools and is often used as a first stage in a more complex evaluation exercise involving reducing a large number of possible methods/tool to a short-list of one or two candidate method/tools that can be evaluated in more detail.

In this case the evaluator is responsible for:

- identifying the candidate methods/tools;
- devising the assessment criteria (i.e. the features to be assessed and the judgement scales) with or without specific help from potential tools users;
- compiling information about the method/tools;
- scoring each feature for each method/tool;

- analysing the scores;
- presenting a report on the evaluation.

Pros
This is the fasted and least costly form of feature analysis.
The number of different roles is minimised.

Cons
The entire evaluation is based on the subjective evaluation of a single person (or team), so it may be biased.
The evaluator may not be representative of potential users of the method/tool.

6.4.4.2 The Case Study Approach

This is likely to be fairly inexpensive but provides only limited confidence in the reliability of the evaluation. DESMET defines a case study evaluation exercise to be one where a method or tool tried out on a real project. In the context of Feature Analysis, this requires a separation of roles between evaluation staff and software development staff who will act as assessors:

- the evaluator (or evaluation teams) selects the methods/tools to be evaluated;
- the evaluator devises the assessment criteria and produce an evaluation form working in conjunction with the potential users;
- the evaluator selects one or more pilot projects that will try-out the methods/tools usually one method/tool per pilot project. The pilot projects should be selected to be typical of the projects undertaken by the organisation
- a software developer (or a development team) in each pilot project then tries out one of the methods/tools;
- the software developer(s) who tried out the method/tool score each feature of the method/tool;
- the evaluator analyses the scores and prepares and evaluation report.

Pros
The methods/tools are assessed on a real project, so their scalability can be factored into the evaluation.
The assessment is made by members of the user population.

Cons
Different developers will evaluate different methods/tools on different projects. Thus some of the different scores could be due to:

- the different projects stressing different feature in different ways;
- the assessors having different abilities and different learning curves and so assessing the methods/tools from different perspectives;
- different assessors interpreting the evaluation scales differently

6.4.4.3 The Formal Experiment Approach

This is likely to give the most trustworthy results, but is also often the most expensive approach to carry out. It involves a separation between evaluation staff and experimental subjects:

- the evaluator (or evaluation teams) selects the methods/tools to be evaluated;
- the evaluator needs to devise the assessment criteria and produce an evaluation form working in conjunction with the potential users;
- the evaluator then needs to plan and run the evaluation experiment. The procedures for running the experiment are the same as those discussed by Shari Lawrence Pfleeger [1]. They include among other things:
  - identifying an appropriate experimental design;
  - identifying subjects (preferably by a random selection of potential users from each class of user). Note the experimental subjects should not include users who participated in the development of the feature list;
  - identifying the tasks to be performed by the experimental subjects using the method/tool;
  - organising any training for experimental subjects;
  - running the required experimental trials according to the experimental plan
- the experimental subjects will need to try out the candidate method(s)/tool(s) according to the experimental instructions and score each feature;
- the evaluator analyses the forms and prepares and evaluation report.

Pros
A well-designed experiment minimises the effect of individual assessor differences.

Cons

It is difficult to ensure that the tasks performed in an experimental situation properly reflect "real-life" tasks, so there is often a question as to whether results will scale-up.

6.4.5 The Survey Approach

This is similar to the formal experiment approach in the sense that it solicits an assessment from a number of different people. However, in this case the assessors are not the potential users but people who have some experience of the candidate methods/tools. These may be people external to your company or people from other parts of your company.

In this case the evaluator (or evaluation team) is responsible for the following activities:

- identifying the candidate methods/tools;
- devising the assessment criteria and judgement scales in conjunction with the potential method/tool users;
- designing and running the survey. This includes:
  - deciding on the type of survey (e.g. a postal survey v. personal interview)
  - creating the survey documentation (e.g. questionnaire, instructions etc.)
  - identifying who will be asked to participate in the survey;
  - running the survey according to the survey design;
- analysing the replies from the individuals surveyed and preparing an evaluation report. In the case of a postal survey, this needs to include an assessment of the response rate and an evaluation of the representativeness of the respondents as well as collating the completed evaluation forms.

Pros

You can obtain information about the methods/tools on real-life projects.
A survey can provide results in shorter timescales than a case study.
A well-designed survey where you understand the population, sample that population at random and obtain replies from all the sampled individuals has all the rigour and validity of a formal experiment.
A survey is less human intensive than a formal experiment.

Cons

This approach is only viable if you can identify individuals with experience of the methods/tools you want to evaluate.
You cannot control the experience, background and capability of the assessors in the same way that you can in a formal experiment.
With a postal survey results can be biased because:

- the respondents are not typical software developers or managers. For example people who have time to reply to a survey may be better organised than most managers/developers.
- the respondents may have an axe to grind. For example, they may have had a method/tool imposed on them and want an opportunity to gripe about it.
- you may get patchy responses (i.e. lots of information about some of the methods/tool of interest but none about others).

6.4.6 Assumptions and constraints

Any assumptions and constraints understood to apply should be explicitly stated. These might address issues such as:

- the maximum effort and funding that will be available for evaluation activities;
- the maximum duration of the activities;
- the installation and training support that will be available;
- the availability of the environment that is required for tool evaluation.

6.4.7 Timescales and effort

The plan needs to identify all the tasks required to complete the evaluation and identify timescales and staff effort.

6.4.7.1 Example

Table 6 shows an example breakdown of the tasks and effort required to carry out a "hands on" evaluation of a fairly complex analysis and design tool i.e. an Analysis and Design tool to support SSADM. This was a mixture of the screening mode and case study approach where a single person performed the evaluation. Although the evaluator tried out the tools on example tasks, the tools were not tested on a real project. The feature list comprised 20 feature sets which totalled over 120 individual features. A paper-based evaluation had already been carried out in order to reduce the number of candidates so that a "hands on" evaluation would be cost effective

Table 6 Example of tasks and effort for a Feature Analysis

Cost Factor	Staff days
Produce the project plan	The time required to prepare the plan will depend on the depth of evaluation required of the tool and indeed the nature of the tool under evaluation.
Negotiate the loan by the supplier of the tool	2.0 per tool
Set up the evaluation environment	2.5 per tool
Initial meeting with the supplier after receiving the tool	0.5 per supplier
Initial pre-reading on tool	0.5 per tool
Evaluation of the tool (including familiarisation period). This does not include performance and stress testing	5 to 15 per tool. The number of criteria to be checked is one of the main determinants of effort
Write report(including the outcome from the post evaluation meeting, below)	5.0 (based on two tools)
Post evaluation meeting with supplier	0.5 per tool
QA report, including consistency checks, re-work, etc.	2.5 (based on two tools)
Printing, copying, binding, dispatching report	1.0
Total	15 - 25 days per tool

6.5 Analysing a Feature Analysis Evaluation

The aim of a feature analysis is to identify which of several candidate method/tools best fits the requirements of potential users. After set of method/tools have been assessed you will need to analyse the score sheets and determine which of the methods/tools is best. There are many different approaches to this problem. The two main approaches depend on whether or not you have an explicit acceptance threshold for each compound feature.

If you have explicit acceptance threshold, your analysis should be based on the difference between the acceptance threshold for each feature set by the users and the score each method/tool obtained for the feature. If you do not have an explicit acceptance threshold (which often happens in a screening mode evaluation), you must base your assessment on the scores of the methods/tools relative to one another. In both of these cases the results can be presented as a profile of score against a particular feature set.

Another problem occurs when you have more than one score per method/tool. This happens if you have organised the evaluation as an experiment or a survey. In this case the different assessments of each method/tool must be collated before the methods/tools can be compared.

6.5.1 Method/tool Evaluation Profiles

An Evaluation Profile represents the score achieved by a specific method or tools against a specific feature set. A particular evaluation profile can reflect a specific iteration. An evaluation profile can be represented either numerically or graphically.

6.5.1.1 Difference based evaluation assessments

Table 7 presents an example of a numerical representation, for a particular feature set composed of compound features.

Table 7 Example Evaluation Profile for the Learnability feature set

SubFeature	Importance	Conformance Acceptability Threshold	Conformance score obtained for candidate method/tool	Difference
Quality of Documentation	H	3	2	-1
Learning Curve	H	2	3	1
Training Quality	H	4	3	-1
Training Effort	D	3	4	1
On-line help	N	2	3	1
Total				(-2)(+3)

In this example all the features are non-mandatory ones. If you have done your evaluation or analysis iteratively any method/tool not providing mandatory features would have already have been eliminated from the list of candidate methods tools.

If you have defined the importance of each feature and the acceptance threshold based on user requirements you can determine the difference between the score achieved by a particular method/tool. The example in Table 7 gives you an overall assessment for learnability for this candidate of (-2)(+3) if you ignore importance. If you restricted yourself to an assessment of highly desirable features only the score would be (-2)(+1)

NEVER subtract the 2 from the 3 to give a combined score of 1! This will give you results that you will be unable to analyse - assuming that you believe that success in one dimension can never cancel out failure in another. You should also be aware that you could obtain a result of (-2)(+3) for the learnability of another candidate method and tool but the result could be due to a different profile for example due to "Quality of Documentation" scoring 4 and "On-line help" scoring 1.

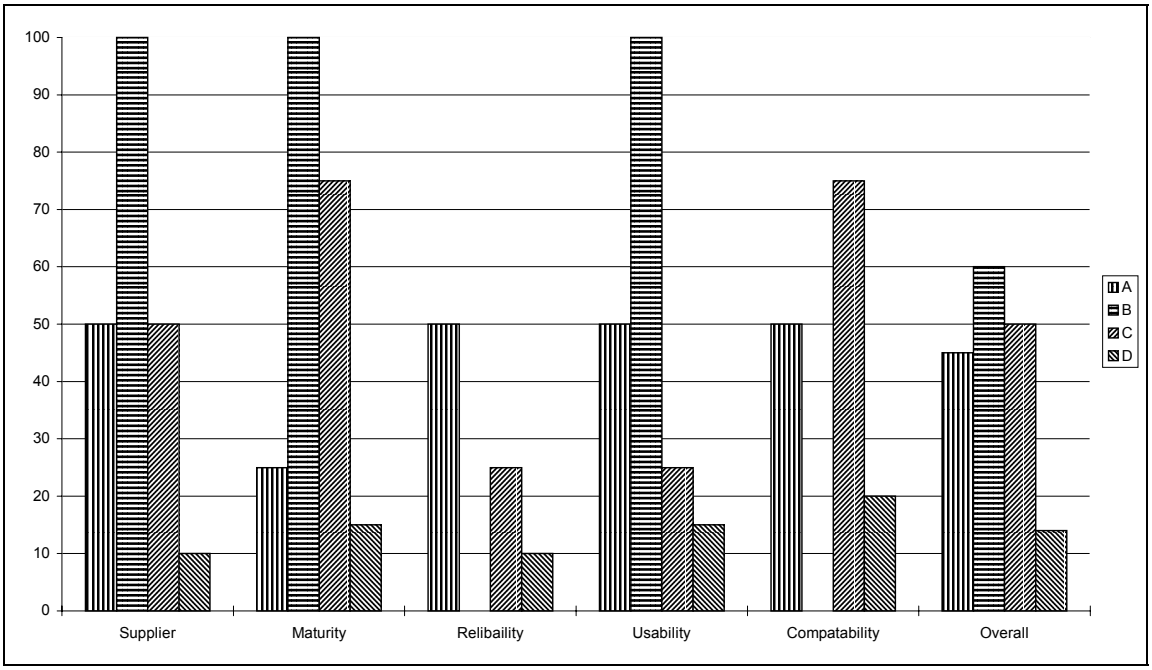
If you want a single number that allows you to rank a set of methods/tool, you should consider the percentage of features achieving acceptance levels (i.e. in this case 3/5=60%).

A useful way to visualise the scores of set of methods/tools is to show the percentage achievement of each method/tool for the top level feature set as an overall profile as shown in Figure 1. Figure 1 shows the results of assessing four tools against a top level feature set comprising Supplier features, Maturity Features, Reliability features, Usability features and Comparability features. In addition, the overall assessment is shown. Of the 4 tools assessed it is clear that tool D is inferior to the other tools. However, a single overall assessment would give a mistaken view of the merit of tool B. It obtains the highest overall score but this score is obtained by excellent scores in three out of the five features sets and bad scores in the other two. If you want to select a tool that provides some support for all dimensions, tools A and C are preferable.

As discussed above, the score differ depending on whether or not you include all features or only features of a particular level of importance. For example, if tool B obtained its high score by fulfilling all highly desirable features (as well as all mandatory ones), whereas tools A and C obtained their scores by failing to achieve some highly desirable features but achieving some less desirable features, tool B would be preferable. This indicates that even if you do not organise your assessment in iterations you should organise your analysis of the results iteratively. You should screen out tool/methods that do not achieve mandatory features before undertaking more detailed analyses.

Finally, you need to be aware, that if you have a mixed user population with different requirements, you will need to analyse the scores against the acceptance threshold for each group separately (unless by chance they are all the same). It is quite possible to find the method/tool that is best for one subset of the user population is not the best for another. In this case it is up to the evaluator to report the results and the sponsor to decide whether to accept different solutions for the different user groups or impose a single method/tool on all groups.

Figure 1 Evaluation Profile



6.5.1.2 Absolute scores

If you did not have information about the importance of features or the acceptance level required of compound features, your analysis must be based on the accumulating the absolute scores. (Note this means treating an ordinal conformance judgement scale as if it were an interval scale measure, so treat such procedures with care!) For example, the combined score for the learnability feature set in Table 7 would be 15. This can also be represented as a percentage of the maximum score which in this is 25, giving a percentage score of 60%. Again you can obtain the aggregate score for each top level feature set and you can sum the scores obtained for all the feature sets into an overall score. Note when aggregating scores, you should aggregate the raw score before converting to percentages.

If you have simple as well as compound features you need to assign a score to the YES and a score to NO. It is probably inappropriate to score 1 for providing a simple feature because that will bias your assessment towards methods/tools that provide some support for compound features. We suggest provision of a simple feature should score 5, failure to provide a simple feature should score 0.

If you are able to assess the relative importance of features but not specific acceptance criteria, you can use the importance assessment as a weighting factor. We suggest the following weights:

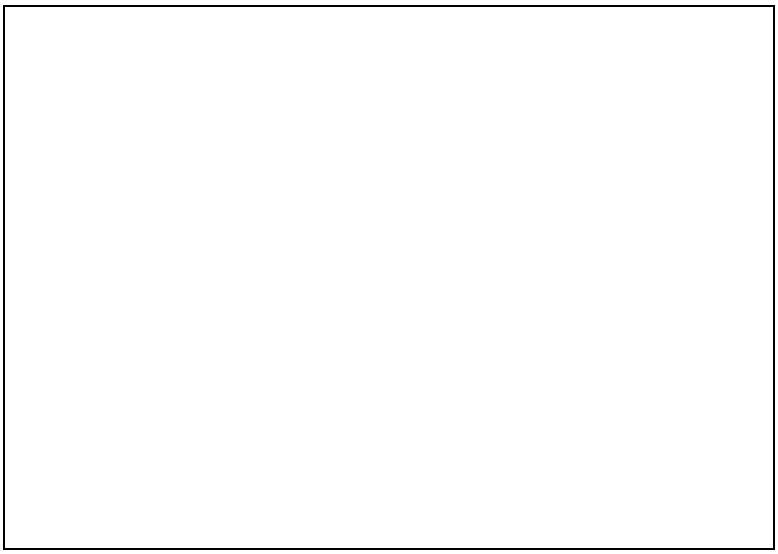
Mandatory features: 10
Highly desirable: 6
Desirable: 3
Nice: 1.

This would give a score for the feature set in Table 7 of 63 out of a possible 110 giving a percentage score of 57%. Note there is no defined rationale for determining appropriate weights. Again when considering the overall score you need to be careful about way in which the scores are aggregated over the different high level features. It is usually preferable to select methods/tools that cover top level feature sets in preference to tools that score highly on some features and low on others.

6.5.2 Graphical Representation of the Results

One way of representing results is to use "multiple metrics graphs" which are a variation on Kiviat diagrams that depict characteristics as slices of a large, circular pie: In a multiple metrics graph [8], the pie is divided into unequal slices, so that the size of the slice (the number of degrees of the arc), represents the importance of the feature relative the other features displayed. Thus the larger the slice, the more important the feature. A series of multiple metrics graphs could be drawn to represent each feature set for each candidate.

Figure 2 A multiple metric graph



The basic principals of creating multiple metrics graphs are as follows:

1. Form two concentric circles:
   - the outer circle represents the worst case value;
   - the inner circle represents your acceptance threshold;
   - the centre represents the best possible situation.

2. Draw lines from the centre to outer circle to divide the pie into "slices"
   - each slice represents a feature or subfeature;
   - the size of each pie slice represents the importance of the feature relative to the overall set of features.
3. The centre of each slice (a line equidistant from both slice edges) represents the judgement scale onto which the value of the feature is scored. A polygon is then created by joining the scores together intersecting the inner circle at all the slice boundaries.
4. The smaller the area of the polygon, the better the match the outcome of the candidate feature set.

In this case, feature A is more important than feature C which is more important than feature B. The Polygon depicts the results, being that features A and B have failed to meet the threshold, whilst feature C is well inside the requirements.

6.5.3 Combining the Results of Experiments or Surveys

If you have undertaken an iteration using the formal experiment or survey approach, then additional techniques will be required in order to combine the results from the various assessors that have scored the same method or tool.

The simplest approach is to plot the distribution of the scores on frequency diagram for each feature, and to choose the modal (most frequently occurring) score. **Do not** use the average score it is inappropriate for ordinal scale measures. We suggest the mode rather than the median because you should be looking for the highest consensus value not the mid-point of the sample.

If you find that scores for the same feature for the same method/tool are distributed evenly across a number of different scale points you need to review your subjects, your scales and the specific method/tool for the reason why the subjects cannot come to a consensus view. Possible reasons include

1. a lack of understanding of the purpose of the method or tool.
2. misunderstanding of one or more of the evaluators about the definition or scale of a particular feature.
3. evaluations being performed for different underlying requirements, due to:
   - the requirements being unclearly specified;
   - the evaluators being poorly selected;
   - the requirements correctly including a wide scope that is not catered for in a particular candidate.
4. ambiguous scale points on the judgement scale, or scale points defined in too subjective a way.

You need to assess whether there is a systematic problem with the assessment of the particular feature for the particular method/tool, or the specific method/tool, or the evaluation exercise as a whole. In the first case, you can omit assessment of the specific feature, in the second case you can omit the analysis of the specific tool, in the final case you will need to reject the whole evaluation exercise.

In the event that there is no systematic problem, we advise you to score the feature as the worst case. So for example, if 50% of respondents score a feature on scale point 2 and 50% on scale point 5 and there appears to be no underlying problem with the assessment process score the feature as scale point 2.

6.5.4 Summary of analysis methods

This section has discussed the problem of analysing and reporting the results of a feature analysis. The main theoretical problem is the issue of analysing ordinal scale measures which formally should not be used in certain types of calculations (i.e. sums and averages). We have suggested that the best way to analyse such data is to consider the percentage of feature scores that meet their acceptance threshold (i.e. the level required by the method/tool users).

However, if such information is not available and you need a single figure of merit, we suggest relaxing the dictates of measurement theory and summing the scores (weighted if necessary). This relaxation bears with it an obligation to treat the “figure of merit” with caution. You must be sure that the results are not misleading, in particular you should look out for high values resulting from very high scores on some features and very low scores on other features. In such a case you may decide that a lower scoring method/tool that covers all features to a reasonable level is a better choice than the highest scoring method/tool.

7. Quantitative Case Study Method

This section was prepared in collaboration with Lesley M. Pickard.

7.1 Introduction

This section describes how to undertake a quantitative case study based on work done as part of the DESMET project [9], [10]. In the context of methods and tool evaluations, case studies are a means of evaluating methods and tools as part of the normal software development activities undertaken by an organisation. The main benefit of such case studies is that they allow the effect of new methods and tools to be assessed in realistic situations. Thus, case studies provide a cost-effective means of ensuring that process changes provide the desired results. However, unlike formal experiments and surveys, case studies do not involve large number of projects so it is more difficult to ensure that ant evaluation will produce meaningful results.

7.2 Case study method

Case studies are a standard method of empirical study in various "soft" sciences such as sociology, medicine, and psychology, but there is little information available on how to perform a proper case study. However, Robert Yin's book provides a notable exception [11]. He takes the view that a case study should be used when "a how or why question is being asked about a contemporary set of events, over which the investigator has little or no control". For evaluations, we need case studies to evaluate "which is better," as well how or why. The "how" or "why" type of case study can provide valuable insights in to why a new technology results in better products, however, in this article we concentrate on discussing the "which is better" type of case study.

In the area of software engineering and information technology, there has been some interest in the method of case studies over the last few years, in addition to the DESMET. Lee [12] has considered a method for case studies of management information systems. His article provides a detailed discussion of how single-case case studies relate to the scientific method. Glass [13] takes a more pragmatic approach than Lee and discusses practical guidelines for planning and controlling pilot studies (another name for case studies). The DESMET guidelines fall between Lee's scientific theory and Glass's managerial practice and attempt to provide technical guidelines concerned with organising a trustworthy case study to evaluate different methods/tools. The critical issue of the DESMET approach is that case studies aimed at evaluating methods/tools are regarded as comparative and context dependent.

7.3 Terminology

One of the aims of the DESMET project was to use as much of the scientific method as possible and in particular to user the terminology of formal experiments. You will see a strong similarity with the terminology Shari Lawrence Pfleeger uses to introduce formal experiments [1]. Thus, as with formal experiments, the first decision when undertaking a case study is to determine what you want to know or investigate, that is, specify your requirements. We refer to a quantifiable specification of the requirements of the case study as an "**hypothesis**"; carrying out the evaluation will produce information that may allow you to test the hypothesis.

The methods/tools you want to evaluate are called the "**treatments**". In order to determine whether a treatment is beneficial if you need to compare it either with an alternative treatment or with the currently used method/tool. The method/tool you currently use is referred to as the **control** treatment. The "**experimental objects**" are *what* the treatment is being applied to (e.g. projects, specifications, or modules) and the "**experimental subjects**" are the individuals or teams *who* are applying the treatment. The control provides a **baseline** of information to enable comparisons to be made. In a case study, the "**control**" need not be fully defined. For example, if you are comprising the use of an object-oriented design method with your current design method, there is no obligation to specify the existing method. In this case you would only be interested in deciding whether the object-oriented method is better than your current method in the context of your own organisation. If you were performing a formal experiment, you would need to define the control fully, so that your experiment could be replicated.

"**Response variables**" are those factors, typically developer productivity and product quality, which are expected to change or be different as a result of applying the treatment. The difference between a formal experiment and a case study is in the use of what are known as "**state variables**". State variables are factors that characterise your case study project(s) and can influence your evaluation results (e.g. application area, system type). In a formal experiment, you sample *over* the state variables i.e. you select a random set of experimental objects which exhibit a variety of those characteristics which are typical in your organisation. In a case study you sample *from* the state variables i.e. select one or more experimental objects which are typical of your organisation as indicated by the distribution of the state variables.

Case studies are evaluation exercises that are undertaken on "real" projects. A "real" project has as its objective the delivery of a software (or software intensive) product (or product enhancement). The "real" projects being used in a case study are referred to as ***pilot projects*** or ***host projects***.

7.4 Guidelines for running case studies

There are eight steps to follow in designing and administering a case study:

1. Identify the case study context

2. Define and validate the hypothesis
3. Select the host projects
4. Identify the method of comparison
5. Minimise effect of confounding factors
6. Plan the case study
7. Monitor the case study against the plans
8. Analyse and report the results

The steps relate to the four criteria for research design quality discussed in [10] and help ensure that you can draw valid conclusions from your investigation:

1. ***construct validity***: establishing correct operational measures for the concepts being studied;
2. ***internal validity***: establishing a causal relationship and distinguishing spurious relationships;
3. ***external validity***: establishing the domain to which a study's findings can be generalised;
4. ***experimental reliability***: demonstrating that the study can be repeated with the same results.

The eight steps are discussed in the following sections.

7.4.1 Identify the case study context

The case study context sets the goals and constraints within which the case study must operate. The goals of the case study define what the case study sponsor wants to know. In the context of evaluations the goal will usually be related to whether or not investment in a particular technology is likely to be a cost effective investment. Sometimes the goal will be extended to consider under what conditions a new technology would be cost effective. The goals of the case study are the basis of the case study hypothesis.

The case study constraints provide the context for case study planning. They should identify:

- the case study sponsor;
- the resources available to organise and run the case study;
- the timescales within which the case study must be completed;
- the importance of the case study to the organisation in terms of the extent to which the requirements of the case study override the requirements of its host pilot project.

7.4.2 Define and validate your hypothesis

7.4.2.1 Defining an hypothesis

Defining an hypothesis involves restating your goals in a formal testable manner. Defining the case study hypothesis correctly is both the most difficult and the most important part of designing a case study. If the hypothesis is wrong the case study will not produce useful results and will be a waste of time and money. Time taken in specifying the hypothesis and getting it agreed increases your chance of getting useful results. It is comparable to spending time getting software requirements right before the software development is started. If the requirements are wrong, the development will be worthless.

When specifying a hypothesis you need to know:

- the goals of the case study;
- the method/tool you wish to evaluate (i.e. your treatment);
- what aspect of the method/tool are you interested in investigating;
- what effect in the response variable you are looking for.

There are various sources of information you can use to help you specify your hypothesis:

1. The identification of a particular problem in your development process which the method/tool is intended to solve;
2. Claims made by developers/vendors of the method/tool you are interested in;
3. Results of any previous in-house evaluations;
4. Reports of evaluations or descriptions of the method/tool from external sources.

When you formulate your hypothesis you should begin by defining the effect you expect the method to have. This definition must be detailed enough to make clear what measurements are needed to demonstrate the effect. For example, if you expect a new method to improve productivity, you must state whether effort and duration will be affected and in what way. Without this information, you cannot identify the data you need, and you will not be able to measure and collect information to draw valid conclusions.

It is necessary to determine that you can measure the effect of the method/tool directly or, if this is not possible, to ensure that you measure a valid ***surrogate***. A surrogate is a measure we use when we cannot measure what we really want to. For example, we might count the number of defects found during acceptance testing as a surrogate quality measure but what we really want to know is how many defects the users of the system will find once it is deployed.

It is also important to define what is not expected to happen. If you expect a method/tool to increase productivity without affecting quality or lead time you need to make the "no change" factors explicit in your hypothesis. If you do not check whether or not the new technology has unwanted side-effects, your evaluation may lead to incorrect results.

In addition you need to remember that formally, we never prove hypotheses, we can only disprove them, so we state our evaluation hypothesis in terms of the ***null*** hypothesis that there is no difference between treatments. However, research is usually proposed and funded on the basis that the alternative hypothesis is plausible i.e. that the new method is significantly better than the current method. Thus although the formal case study data analysis and evaluation should address the null hypothesis, you should be ready to present your findings to managers and staff in terms of the alternative hypothesis.

The more clearly you define your hypotheses, the more likely you are to collect the right measures, to test them properly, and to achieve construct validity. You must specify carefully what really interests you. For example, process improvement programs often define quality as the reduction of rework and waste, presenting quality in terms of defect rates from the perspective of a software developer. However, this definition differs from the user's point of view, where operational reliability, efficiency, and usability reflect how the user sees the software. It is vital that you consider all the effects you expect the new technology to have, including the neutral "no change" effects, to construct a complete hypothesis. It is also important that you can identify the response variables you will use to test your hypothesis. If you are forced to use surrogate measures, you need to be certain that they are valid surrogates.

7.4.2.2 Validating your hypothesis

You need to consider the following issues to determine whether your hypothesis is valid and you can correctly interpret the evaluation results:

1. Can you collect the data to allow the selected measure(s) to be calculated?
2. Are the response variables valid (i.e. direct measures or valid surrogates?
3. Can the effects of the treatment you wish to investigate be clearly identified and isolated from the influences of the rest of the software development activities?
4. If you intend to integrate the method/tool into your development process is the treatment likely to have an effect, possibly detrimental, on your environment other than the one which you wish to investigate?

7.4.3 Selecting host projects

The host projects you choose must be representative of the type of projects your organisation or company usually performs to ensure that the results from the evaluation are applicable to more than just the trial project.

One way of identifying typical projects is to define your ***organisation profile***. You should try to describe your projects in terms of significant state variables, and then use this state variable information to select host projects that are most typical of your organisation. Your selection should be based on the frequency with which each type of project is developed.

In practice, it may be difficult to control the choice of host project. However, the extent to which the host project is typical of the organisation is central to the issue of external validity. If your host project is atypical of the projects you usually undertake, you will not get very useful results.

Some examples of state variables to help identify characteristic projects are:

- application area
- methods and tools used
- development process
- business type (e.g. finance)
- scale of project
- quality and experience of staff

As well as organisational state variables, you may need to identify state variables which are particularly relevant to the correct application of the method or tool under investigation. Some examples are:

- quality and experience of staff needed to use a particular method or tool may be different from your "norm" for projects. For example, a method may only show an improvement in quality or productivity if used by experienced staff and may show a deterioration if used by inexperienced staff
- physical environment
- complexity level of problem
- product size range

The values of the state variables form your organisation profile which allow you to identify the characteristics that typical project has in your organisation. For example if your organisation deals with DP applications with a typical product size of 30-50K lines of code you should select host projects that match these characteristics.

Profiles express relative frequency over a set of values of a state variable. However, different state variables may interact. For example, if 80% your organisation products are DP applications and 20% scientific, they may have different typical product sizes, for example 30-50K lines of code for DP processing and less than 20K lines of code for scientific.

The selection of host projects is an activity where case study design diverges from the design of formal experiments. Instead of taking a random sample of experimental objects so that you can be sure that your results are widely applicable, you must select an experimental object that is typical of the objects handled by our own organisation, in the belief that the results will apply to other similar objects in your organisation. It must be emphasised that this approach is not a rigorous scientific approach, it is purely pragmatic approach and as such is subject to the risk that the results observed on the host project may be atypical (i.e. a fluke). Nonetheless, the DESMET project takes the view that decisions made on incomplete information are likely to be far better than decisions made on guesswork or wishful thinking.

7.4.4 Identifying the method of comparison

Identifying the method of comparison in a quantitative case study is equivalent to defining the experimental design in a formal experiment. There are three ways to organise your study to facilitate this comparison:

1. By comparing the results of using the new method with a company baseline.
2. If the method applies to individual components, by applying the method at random to some product components and not to others.
3. By comparing similar projects.

7.4.4.1 Company baseline case studies

This method can be used if your company gathers data from projects as standard practice, and makes data available to determine information such as average productivity or defect rate. This type of case study involves a single host project using the new method. The response variable values from the host project are compared with the corresponding variables from comparable previous projects or a subset of comparable projects (e.g. projects undertaken in the same timeframe as the case study).

Note. If your organisation profile identifies several major project types, you can organise a multi-project case study with a host project from each of the major project types.

Pros
Since a company baseline is derived from values from a set of projects, it can be used to assess the variability of the response variable values. This improves the chance of identifying a significant change in the response variable value obtained from the host project.

Cons
The design relies on an organisation having a set of reliable, comparable productivity and quality measures from its projects.

7.4.5 Within project component comparison case studies

In this design, components (e.g. modules or subsystems) in a single host project are assigned at random to each of the treatments. In this case, the case study resembles a formal experiment, since the response variable values are replicated (i.e. you obtain values from each component) and standard statistical methods can be used to analyse the results. However, since the host project is not replicated nor drawn at random from the population of all projects, this type of case study is not a true formal experiment.

This kind of study is useful for methods that may be applied to different degrees. For example, if you want to know what level of structural testing is most cost effective, you can measure the level of structural testing achieved for different modules and compare testing effort and subsequent defect rates (or defect detection efficiency if you have seeded errors).

Pros
1. Since the case study applies within a single host project the influence of some confounding factors is reduced. For example, the project staff are the same, the application is the same.
2. This design gives the highest level of statistical confidence in the results because it allows the case study to be organised and analysed as if it were a formal experiment

Cons
1. It practice it is difficult to ensure that components are assigned at random to the treatments.
2. If staff perceive one treatment to be superior, it is difficult to ensure that they use the assigned treatment and apply it without bias.

7.4.5.1 Cross-project comparisons - Sister project case studies

A cross-project comparison or sister project design is appropriate when you cannot use a company baseline or a within project comparison. The term "sister project" is used for this type of design by analogy with the term "sister ship". Sister ships are ships built to the same specification in the same ship-building years by the same workers using the same production methods. In the case of software projects, sister projects are projects that create similar applications in the same organisation developed by similar software development staff, using development methods.

To perform a cross-project comparison, the case study must comprise (at least) two host projects: one using the new method/tool, and the other using the current method/tool, all other development methods and procedures should be the same. Each project should be typical of your organisation, and both should have similar characteristics according to the state variables you have chosen. The host projects are run in parallel and the response variables from each are compared.

Pros
The design can be used when there is no company baseline available.

Cons
1. It is the most expensive method of running a case study because it involves more than one host project.
2. It gives the least reliable result because the "norm" is only derived from one project(or experimental object) whereas the other two types of design comparisons are based on many experimental objects (projects for organisation baseline and components for within-project design).

7.4.5.1.1 An alternative sister project design

In some cases it is possible to development a product a second time using a different development method. This is called a ***replicated product design***.

The advantage of a replicated product design is that some of the difference between the sister projects is eliminated because they both produce the same product. However, it is often difficult to ensure that both products are produced to the same commercial standards. For example, the method is often used when a "research" group want to demonstrate the superiority of a new method compared with current development methods. However, if the research group staff undertake the replication project, the results may be biased because the research group will usually have more experience with the new method than would the development staff, and would be more motivated to see it succeed. In addition, a research group may not develop the replicate product to the same rigour as the development group in terms of quality control activities such as reviews, documentation etc.

These problems can be overcome if the case study sponsor insists that the development group undertake both projects to commercial standards, and the product which performs best in final system test (or acceptance test) not only determines which method is best but is also the one released to customers.

7.4.6 Minimising the effect of confounding factors

When the effect of one factor cannot be properly distinguished from the effect of another factor, we say that the two factors are confounded. Confounding affects the internal validity of the study. For example, if expert software

engineers tested tool A while novice software engineers tested tool B, we could not tell if the higher-quality software produced by the experts was the result of their experience or of using tool A.

Software-related case studies often have confounding factors. The most significant are likely to be:

- using a case study to learn how to use a method or tool while trying to assess benefits at the same time. In this case, the effects of learning to use the method or tool might interfere with the benefits of using the method or tool. For instance, long-term productivity improvements might be hidden because of initial decreases in productivity due to the learning curve. To avoid this effect you need to separate activities aimed at learning how to use a new technology from those aimed at evaluating it. In particular you need to ensure your staff receive appropriate training and are also given the opportunity to use the method/tool in anger before taking part in the case study. In addition, you should monitor the host project to ensure that the development staff are using the method/tool correctly
- using staff who are very enthusiastic or very sceptical about the method or tool. Staff morale can have a large impact on productivity and quality. Differences in the response variable may be due to staff enthusiasm, or to differences in enthusiasm from one developer to another. To minimise this effect you need to staff a case study project using your normal methods of staff allocation;
- comparing different application types. For example, productivity of real-time system developers is usually lower than for data processing systems, so case studies should not compare across application domains. Appropriate selection of case study projects will avoid this problem.

Sometimes it is possible to control a confounding effect rather than eliminate it. This usually means that we can design a multi-project case study where the different host projects are selected to represent projects developed under different conditions. For example, to investigate whether the benefits of some method or tool are influenced by application type, we can identify a pair of case study projects for each application type: one to use the new method and one to use the current method.

Partial confounding is sometimes controlled by measuring the confounding factor and adjusting the results accordingly. For example, in studying how different levels of reuse affect quality and productivity, you may select a host project where components (such as specifications, designs or code) are being reused, measure the amount of each component that is reused, the component development productivity and its defect rate. If you suspect that component complexity affects productivity and defect rates as well as reuse, you can record component complexity and use partial correlations to assess the relationship between percentage reuse, productivity and defect rates adjusted for complexity.

7.4.7 Planning the case study

Basili, Selby and Hutchens [14] emphasise that organisations undertaking experiments should prepare an evaluation plan. The plan must identify all the issues to be addressed so that the evaluation runs smoothly. This includes identifying the training requirements, the measures needed for analysis, the data collection procedures that will be used, and the people responsible for data collection and analysis. The attention to detail contributes to experimental reliability.

The evaluation should also have a budget, schedule and staffing plan separate from those of the host project. A separate plan and budget is need to ensure that the budget for the evaluation does not become a contingency fund for the host project itself! Clear lines of authority are needed for resolving the inevitable conflicts of interest that occur when a development project is used to host an evaluation exercise. For a more detailed checklist of factors to consider in a case study plan see Glass [13].

7.4.8 Executing the case study

The case study's progress and results should be compared with the plan. In particular, you need to ensure that the methods or tools under investigation are used correctly, and that any factors that would bias the results are recorded (such as change of staff, or a change the priority or importance of the case study projects). Auditing conformance to experimental plans and recording any changes are essential elements of experimental reliability. At the end of the study, you should write include an assessment of the evaluation process in your evaluation report including any recommendations for evaluation process changes.

7.4.9 Analysing and reporting results

The method you use to analyse your results depends on the case study design you have used. If you have used a company-baseline design, you need to derive an appropriate baseline for the response variable(s) of interest. If you have used a within-project component-based design you can use standard statistical techniques such as analysis of variance (ANOVA) to analyse your results. In the case of a sister project design, there is little formal analysis that can be done, you can only assess ***subjectively*** whether the differences in the response variable values from the host projects are large enough to indicate a difference between the treatments.

In addition, in the context of a case study, you will need to perform some preliminary analysis prior to starting the case study. This relates to the definition of the organisation profile that you need to select the host projects. As part of the report, you should compare the host project(s) state variables with the organisation profile. Note. If you were unable to select the host project(s) on the basis of their fit to the organisation profile, you will need to assess their "typicality" when you analyse the results to confirm whether or not the results are likely to generalise.

These issues are discussed in section 0.

7.4.10 Case study checklist

To complete this discussion of the procedures for performing a quantitative case study, this section presents a checklist of questions you need to be able to answer if you are organising a case study.

Setting the Context
What are the objectives of your case study?
What are your external project constraints?

Setting the Hypothesis
What is your evaluation hypothesis?
How do you define in measurable terms what you want to evaluate (that is, what are your response variables and how will you measure them)?

Validating the Hypothesis
Can you collect the data needed for calculating the selected measures?
Can the effects of the treatment you wish to evaluate be clearly identified and isolated from the influences of the rest of the development?
If you intend to integrate the method or tool into your development process, is the method or tool likely to have an effect other than the one you want to investigate?

Selecting the host projects
What type of applications does your organisation undertakes?
What proportion of projects are undertaken in each application type?
Which state variables or project characteristics are most import to your case study?
Do you need to generalise the result to other projects? If so, is your proposed case study project typical of those projects?
Do you need a high level of confidence in your evaluation result? If so, do you need to do a multi-project study?

Selecting the Case study design
At what level does the treatment apply, project/product or component level?
Do you have productivity and quality records on past projects?

Minimising Confounding
Have the staff been assigned to the host project in the normal way?
Has adequate training been given in the new technology prior to the case study?
Have the experimental subject had sufficient experience in the new technology outside the classroom?

Planning the Case study
What are the experimental subjects and objects of the case study?
When in the development process or lifecycle is the method to be used?
When in the development or lifecycle are measurements of the response variables to be made?
Are the evaluators and development staff aware of their roles and responsibilities in the case study?

Monitoring the case study
Do you have a mechanism for ensuring that the experimental subjects are using the new method/tool correctly?
Is there a mechanism for reporting and investigating any deviations from the case study plan?

Analysing the results
How are you going to analyse the case study results?
Is the type of case study going to provide the level of confidence required?

7.5 Data analysis for case studies

7.5.1 Establishing a organisation profile

The main problem with establishing an organisation profile is deciding which state variables are important in your organisation. Once you have identified suitable state variables, setting up the profile simply involves defining the distribution of projects over those variables. How you set up your profile for a particular state variable depends on its scale type.

For a nominal scale state variable (e.g. application type) you simply need to calculate the proportion of your projects in each category (e.g. 20% scientific against 80% information systems). You can use a similar technique for an ordinal scale variable (e.g. product complexity measured on a subjective scale such as very high, high, average, low, very low). For either type of variable, a "typical" project would be one that exhibited the characteristics that were most common in your organisation.

If your state variables are interval or ratio scale, you can construct a *histogram* or *frequency diagram* to assess the distribution of values or use a *box plot* to give a graphical representation of the distribution. Figure 3 shows a frequency diagram for a team experience. In this case, there is no "typical" value for team experience. However, you should avoid selecting a host project with a team experience of more than 4 years or less than one year.

Figure 3 Frequency Diagram for Team Experience

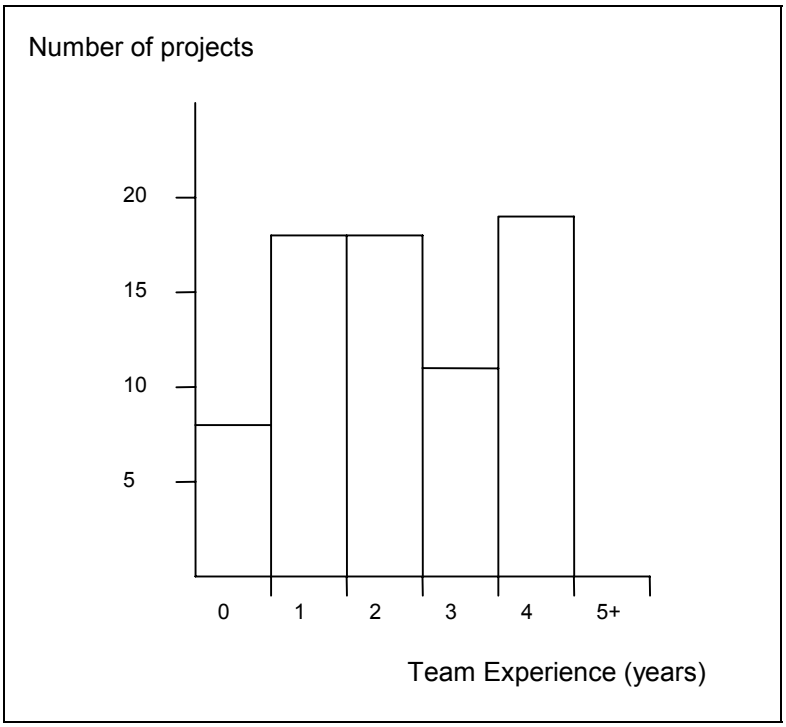


Figure 4 Boxplot of product size for a single company

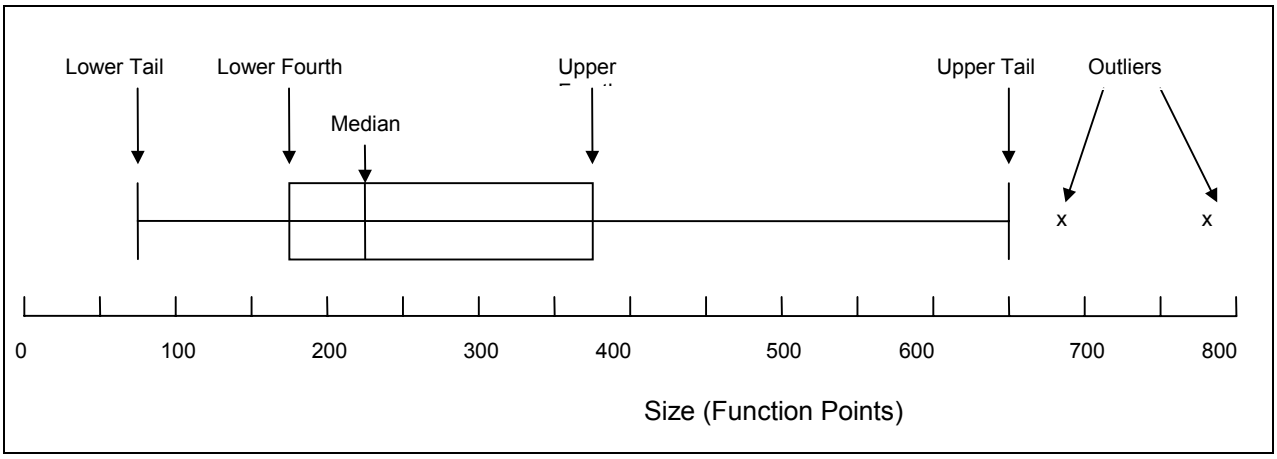


Figure 4 shows a boxplot of product size data. It is constructed from five statistics: the median, the upper fourth (or upper quartile), the lower fourth, the upper tail and the lower tail. The upper and lower fourths are the 75- and 25-percentile points. The upper tail is constructed by multiplying the box length by 1.5, adding the value to the upper fourth and truncating to the nearest actual value. The lower tail is constructed by a similar process. Values that are larger than the upper tail or smaller than the lower tail are called *outliers* and are marked explicitly on the boxplot.

In this case projects of "typical" size would be between 160 (i.e. the lower fourth) and 450 (i.e. the upper fourth) function points. You should select host projects that are expected to be as close to the median value (i.e. 240 function points) as possible.

7.5.2 Company baseline design

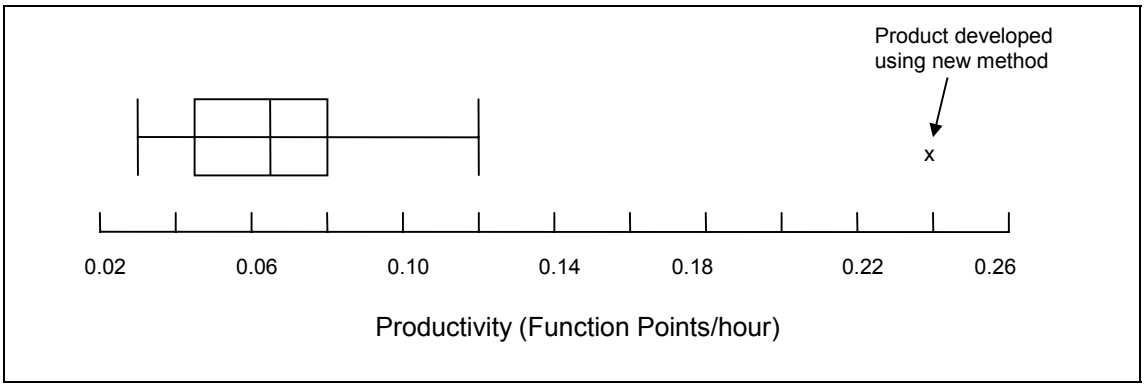
If you use a company baseline design, your will obtain one response variable value from your host project and a set of values from other projects produced by your organisation. A company baseline identifies the central value and spread for a particular response variable (e.g. a measure of productivity, or quality) from the values collected from a set of projects. Clearly there are two issues:

1. Which statistics should be used to measure the central value and spread.
2. Which projects should be included in the dataset used to calculate the values.

In the natural sciences, investigators usually use the mean and variance to define the central value and spread of a dataset. These statistics are fine if you have interval or ratio scale measures that are distributed Normally (i.e. show a bell-shaped distribution). If your response variables do not exhibit a Normal distribution, we recommend using boxplots to establish your company baselines. Incidentally, a boxplot is a useful means of deciding whether or not your data is Normally distributed. If a dataset is distributed Normally, a boxplot would show the mean in the centre of the box, the tail lengths would be approximately equal, and the distance from the median to the upper (or lower) tail would be approximately three standard deviations. Looking at the boxplot in Figure 4, it seems that product size is non-Normal for this dataset since the median is skewed to the left of the box.

As an example of a company baseline, suppose you are comparing the productivity of a new method compared with the productivity of your current method. Figure 5 shows productivity boxplot derived from 46 projects that used the current development method. The baseline for average projects is a productivity value between the upper and lower fourths (0.044 and 0.076) respectively. When the productivity of the host project is displayed on the boxplot, it is clear that the host project has unusually high productivity compared to the company baseline.

Figure 5 A company productivity baseline



A baseline can be refined by using projects that have similar state variable characteristics to the host project or by using project completed in the same timeframe as the host project.

If you have two response variables (e.g. productivity and quality), you can use a scatterplot to view the relationship between both variables. You can then plot the values obtained from a host project to see how the results compare with "normal" relationship.

7.5.3 Within-project comparison

When you use a within-project design, you will have replicated response variables from the within project experimental objects (e.g. modules or subsystems). The most appropriate technique to analyse your data depends on:

1. The scale type of your response variable(s).
2. Whether your data is Normally distributed or not.

These issues are discussed briefly below. A more detailed discussion of statistical analysis techniques from a software engineering viewpoint can be found in Kitchenham [16], alternatively refer to other standard statistical texts for parametric [17] and non-parametric [18] methods.

Normally distributed Data
If the data is Normally distributed (or reasonably close to Normal), it must be ratio or interval scale and you can use Analysis of Variance(ANOVA) to analyse your results, see for example [17]. ANOVA is based on deciding whether the difference between the mean values of the response variables in each treatment group is very large compared with the variance of the values within each group.

There are statistical tests you can use to determine whether or not the difference between group means is significant. If there are two treatment groups you can use the Student's "t" test. If there are more than two treatment groups you use the ANOVA F test to confirm that there are differences between the group means, and the "t" test to test pairwise comparisons.

Non-Normal Data
If your data is not Normally distributed then the choice of the analysis technique depends on the scale type of the response variable(s).

If the response variable is nominal or ordinal, the most appropriate technique is usually a Chi-squared test [18]. For example, you might be interested in whether a new module design method reduces the proportion of modules that exhibit system test errors. The response variable would be a nominal variable identifying whether or not a module revealed defects during system test. You can use the Chi-squared test to assess whether there is a greater probability of producing defect-free modules using the new design method than using the current design method.

If the response variable is interval or ratio scale, you should consider a non-parametric analysis of variance such as the Kruskal-Wallis method which uses ***ranks*** rather than the raw data values [18].

7.5.4 Sister project design

If you have used a sister project design, you will have only one value of productivity or quality per treatment, i.e. no replication. In this case you cannot use statistical methods to assess whether the difference between the response values is significant. You can only compare the value obtained from the control project informally with

the value obtained from the treatment project. A useful way of presenting the results is to calculate the percentage difference between the treatment and the control.

There is one special cases when statistical significance can be tested. This is when the treatment can only be applied at project level but the effect is measured at the component level. For example, if your evaluation the use of a system design method that is expected to produce to better formulated modules. In this case, you can use the same statistical techniques as you would use for a within project case study.

If you are analysing a sister project case study, it is important to confirm the typicality of the host projects using state variables For example, suppose you are investigating the effect of a new development method on productivity. For the two host projects participating in the study, you measure productivity in function points per staff hour using method A (the current method) and using method B (the new method).

Table 8, using real data from [15], shows what you might find. The data indicate that the projects are quite similar with respect to the state variables: size, team experience, project manager experience, duration and function point adjustment factor. Thus, the results in Table 8 suggest that using method B would improve productivity. However, to draw that conclusion, you must be sure that both projects are typical of those undertaken by the organisation. You must also understand the factors that are important for software development in the organisation which might affect successful use of methods A and B.

Table 8 Productivity and State variable information from two projects using different development methods

Variable	Method A	Method B
Productivity (function points/hour)	0.054	0.237
Size (function points)	118	168
Team experience (years)	1	1
PM experience (years)	1	1
Duration (months)	10	9
Function point adjustment value	25	27

Figure 3 and Figure 4 show the distribution of team experience and size for projects in the organisation from which the data in Table 8 was obtained. From Figure 3 you can conclude that team experience of 1 year is not unrepresentative of your organisation. However, it is clear from Figure 4 that the two host projects were relatively small ones (in the lower 25% range). Thus, there must be some doubt about whether the host projects are truly representative of the organisation's projects. There is a risk that any productivity improvements resulting from method B might occur only on smaller projects.

8. Evaluating DESMET

8.1 Introduction

This report has described the DESMET method, but it has not justified why you should accept DESMET more readily than any other software engineering method/tool. DESMET project believed that DESMET needed to be evaluated like any other method. In this section I will describe the evaluation exercises that the DESMET project undertook on the DESMET method itself.

The DESMET project consortium included two organisations tasked with developing the method (University of North London and the National Computing Centre) and two commercial organisations tasked with evaluating the method (GEC-Marconi and BNR- Europe). In terms of project resources, more effort was allocated to evaluating the method than was allocated to developing it. In this final section, I will outline the evaluation process we used and the results of the evaluations of the three DESMET components I have described in this report (i.e. Evaluation Method Selection, Feature Analysis Guidelines and Quantitative Case study Guidelines). The article is based on [19].

8.2 Basis of DESMET evaluation

When we started the DESMET project, we assumed that we could evaluate our method simply by using it to conduct some evaluation exercises. However, when we identified the different types of evaluation and their different advantages and limitation we realised evaluating DESMET would be more difficult than we had first thought.

Thus, in order to identify an appropriate method of evaluating the DESMET method, we attempted to apply our guidelines for evaluation method selection to DESMET itself. The major problem was to define the nature of the impact of using the method. In the view of the DESMET project, the aim of an evaluation method is to ***reduce the risk*** of an invalid or incorrect evaluation exercise. Thus, DESMET fell into the category of technologies whose impact is not directly measurable on a single project (see Section 0).

Such technologies cannot be evaluated by quantitative case studies, because a single outcome cannot be used to give a quantitative measure of probability. Surveys can be used to assess technologies where the measurable impact is indirect, if the technology is in widespread use. However, DESMET was a new technology, so a survey was ruled out. Formal experiments were ruled out for three reasons:

- our resources (in terms of effort and timescales) were insufficient to allow for replicating evaluation exercises
- an evaluation method has too many elements to fit well into a formal experiment;
- it was difficult to identify a control treatment to act as the basis for comparisons.

Thus, we were left with evaluating DESMET by means of a Feature Analysis. We wanted to demonstrate DESMET in action, so our evaluation method for DESMET was case study-based feature analysis.

DESMET is a multi-component method comprising guidelines for performing a number of different evaluation activities (Evaluation method selection, Quantitative Case Studies, Quantitative Experiments, and Feature Analysis), in addition there were several other supporting components to the method (a procedure for assessing evaluation maturity of an organisation; guidelines for coping with human factor issues; a measurement handbook; and a data collection tool). The DESMET partners responsible for evaluating DESMET, therefore, developed a common strategy for evaluating any DESMET component and a standard feature-based questionnaire (called a Generic Evaluation Plan) for each different component.

8.3 Evaluation strategy

Generic Evaluation Plans were prepared for each of the individual components within the method. Evaluation criteria were identified for three successive **levels of validation** defined as follows:

1. **Basic:** Is the component description complete, understandable, usable, internally consistent etc.? Would potential users have confidence that they could use it "for real" or carry out a trial?
2. **Use:** Is the component "helpful"? That is, when it was used, did it achieve its objective, produce the specified results, produce usable and relevant results, behave as expected, and not require expert assistance?
3. **Gain:** Is it better than what was available previously (the "control" situation)? Did it provide added value (e.g. were better decisions made), has it supported a hypothesis, or has it satisfied specific gain criteria? When there was no control situation against which to test a component, we agreed that the gain validation of the procedure would have to be in terms of its usability and effects on people's thinking about how to do evaluations.

The Generic Evaluation Plans attempted to provide a context for planning an evaluation in which the methods and procedures described in the component were related to the evaluation criteria. The plans also provided a format for gathering data (via a series of questionnaires) at significant points during the trial (called evaluation milestones). Anyone planning a trial could use the Generic Evaluation Plan to produce a specific evaluation plan appropriate both to the DESMET component and the host project environment, whilst at the same time ensuring that the trial results would be comparable with any other trial of that component. Having prepared Generic Evaluation Plans for all the components, we identified three methods of evaluation:

1. *A Review*

This consists of a detailed inspection of and critical commentary on the component documentation by an expert reviewer. An external review is one performed by a reviewer outside of the DESMET project team. An internal review involves a DESMET team member who played no part in the design or production of the original component documentation. It is an example of a "screening" mode feature analysis.

This is the least satisfactory means of evaluating the benefit a component and is given least weight in overall component assessment. However, it is very useful for identifying usability problems with the component (e.g. poor presentation, readability, unjustified assumptions etc.). The review process

contributed significantly to the identification of areas needing improvement prior to producing a commercial version of the method.

2. *A Simulated Case Study*
This involves an expert reviewer giving consideration to performing a contrived case study by working through the Generic Validation Plan in detail and responding to the questionnaires at the appropriate evaluation milestones. For the initial milestones covering preparatory work - digesting the documentation and performing preliminary planning - this approach offers a reasonably satisfactory validation technique.

A variation of the simulated case study concerns a "retrospective" validation where the reviewer answers the questionnaires after a trial has been completed. Here, the actual evaluation will not have benefited from the DESMET planning documents, but the validation may help to show how the DESMET component could have led to different results or conclusions. This is a more satisfactory means of evaluating the benefit a component than a simple review. It is also very useful for identifying practical problems with the component (e.g. ambiguities, missing conditions etc.). Like simple reviews, simulated case studies contributed significantly to the identification of areas needing improvement prior to producing a commercial version of the method.

3. *Case Study*
Applying a DESMET component to a real evaluation exercise, using the Generic Validation Plan to create a features analysis questionnaire, is the most satisfactory means of evaluating the benefit of a component. Case studies were, therefore, given highest priority in component assessment. They were useful in identifying areas where the methodology was particularly beneficial as well as areas for improvement.

The extent of the evaluation activity that took place for each main DESMET component is summarised in Table 9. Note that the counts of reviews and paper trials relate to those performed by external experts and exclude those undertaken by the DESMET project itself. BNR-Europe and GEC-Marconi performed most of the live trials, but the University of North London undertook the two trails of the guidelines for formal experiments.

Table 9 Evaluation exercises performed on main DESMET components

Component Name	Activity	Occurrence
Case Study Guidelines	Case study	5
	Simulated case study	1
	Review	1
Formal Experiment Guidelines	Case study	2
	Simulated case study	1
	Review	2
Feature Analysis	Case study	1
	Simulated case study	1
	Review	1
Evaluation Method Selection	Case study	4
	Limited simulated case study in support of review	4
	Review	2

8.3.1 Feature refinement

A flaw in the DESMET evaluation was that the subfeatures were not fully identified when the questionnaires were designed, they were identified when the questionnaires were analysed. The evaluation criteria were refined into the following subfeatures, as follows:

Basic Validation - concerned with the quality of the component documentation. Subfeatures identified for the evaluation exercise were:

Completeness
Understandability
Internal Consistency
Organisation
Appropriateness for audience
Readability

Use Validation - concerned with the quality of the component. Subfeatures identified for the evaluation exercise were:

Understandability
Ease of Implementation
Completeness (are they self-contained?)
Ability to produce expected results
Ability to produce relevant results
Ability to produce usable results

Gain Validation - concerned with the benefits delivered by the component. Subfeatures identified for evaluation exercise were:

Appropriateness for task
Comparison with alternative approaches
Support for decision-making
Cost effectiveness
Clarity (does it illuminate the process?)

Every question in every questionnaire was matched against one or more of these criteria so that the answers obtained from the trials could be weighed to produce a conclusion.

8.4 Case study evaluation results

8.4.1 Scope of live trials

GEC-Marconi evaluated the quantitative case study guidelines by using it on three evaluation exercises. The first case study investigated the effects of re-engineering on software productivity and quality. The second case study investigated the effects on productivity and application performance of using a user interface. The third case study compared two case tools and their effects on quality and productivity. BNR used the case studies guidelines on two case studies one concerned with automatic test generation from a formal specification, the other investigating a tool for object oriented design.

8.4.2 Evaluation Results

The evaluation results are summarised in Table 10.

Table 10 Results of evaluation of Case Study procedures

Basic	Complete:	Some help needed with problem complexity and other details of the investigation.
	Understandable:	More detail on state and response variables. Difficult to select appropriate statistical test.
	Internally consistent:	Yes
	Well-organised:	Yes, except for links to Metrics Handbook.
	Appropriate for audience:	Descriptions not written in enough detail.
	Well-written (readable):	Yes.
Use	Produced expected results:	Yes.
	Produced relevant results:	Yes.
	Produced usable results:	Not clear, as none of the recommendations have yet been implemented.
	Self-contained:	Some consultancy needed to help formulate hypothesis.
	Procedures understandable:	Yes

	Procedures easily implementable:	Effort required to implement the procedures was not always available, as project needs diverted attention from the evaluation. Not clear how to avoid confounding factors.
Gain	Appropriate for task:	Yes.
	Better than other available guidance:	Not answered.
	Good support for decision-making:	Not answered.
	Cost-effective:	Not clear.
	Illuminated the process:	In most cases.

8.5 Feature analysis evaluation results

8.5.1 Scope of live trials

GEC-Marconi performed a trial of the Feature Analysis guidelines by using feature analysis to screen a set of 13 risk management tools for two purposes: in-house use and third party marketing.

8.5.2 Evaluation Results

The results of the evaluation of feature analysis is shown in Table 11.

Table 11 Feature Analysis Evaluation Results

Basic	Complete:	Yes but more features should be included in the sample lists.
	Understandable:	Yes
	Internally consistent:	Yes.
	Well-organised:	Too much DESMET introduction.
	Appropriate for audience:	Comments indicate that the paper may be too academic for some practitioners.
	Well-written (readable):	Somewhat. Needs less introductory material.
Use	Produced expected results:	Yes.
	Produced relevant results:	Inconclusive:.
	Produced usable results:	Approach very formal and needed independent features to work.
	Self-contained:	No guidance on how to score dependent attributes.
	Procedures understandable:	Mostly, but some parts not clear).
	Procedures easily implementable:	Needed tailoring
Gain	Appropriate for task:	Feature sets sometimes); some adjustment and/or compromise required
	Better than other available guidance:	Not mentioned in reviews.
	Good support for decision-making:	Yes.
	Cost-effective:	Took much longer than planned.
	Illuminated the process:	Yes

8.6 Evaluation method selection evaluation

8.6.1 Scope of live trials

BNR used the Evaluation Method Selection guidelines to assess the best method of evaluation for four different empirical studies. The four studies were: to evaluate automatic test generation; to evaluate the adoption of Schlaer/Mellor ObjecTime; to evaluate several different architectures for telecoms switching; to evaluate protocol specification and verification using a formal specification language.

8.6.2 Evaluation Results

The results of evaluating the evaluation method selection procedures are given in Table 12.

Table 12 Evaluation Method Selection Evaluation Results

Basic	Complete:	No.
	Understandable:	Somewhat; there seem to be cases not covered.
	Internally consistent:	Yes.
	Well-organised:	No
	Appropriate for audience:	Yes.
	Well-written (readable):	No
Use	Produced expected results:	No.
	Produced relevant results:	Yes.
	Produced usable results:	No.
	Self-contained:	No.
	Procedures understandable:	Mostly.
	Procedures easily implementable:	Mostly.
Gain	Appropriate for task:	For the most part, the component asks sensible questions and gives sensible answers.
	Better than other available guidance:	Not applicable.
	Good support for decision-making:	Out of four trials, two answers were judged correct, one was judged incorrect, and the fourth case lead to no clear result..
	Cost-effective:	Not applicable.
	Illuminated the process:	Not applicable.

8.7 Evaluation conclusions

This section has briefly described the evaluation exercise the DESMET project undertook to evaluate the DESMET method. The most important aspects of the DESMET evaluation were live trial of the different components of the method, although we also initiated reviews and paper trials. Apart from the results of the individual live trials, it is worth noting that the range of technologies used in the live trials was large. The trials covered specification methods; GUI evaluation; reuse; risk tools; testing methods. This wide range gave us some confidence that we had succeeded in our aim to develop a general evaluation method.

However, we are not complacent about the results of the evaluation exercises since there were fairly serious criticisms of the guidelines we produced. As a result we have been upgrading and improving the method since the end of the project. However, we can confirm that the commercial companies that trialled the method have continued to use it. In the words of Niall Ross of BNR

> "*DESMET will give you useful information, show you some interesting examples and ask some interesting questions you might not have thought of yourself, but whether they stimulate you to produce valuable questions depends on your native wit.*"

9. REFERENCES

1. Pfleeger, S.L. Experimental Design and Analysis in Software Engineering. Parts 1 to 5. SIGSOFT Notes, 1994 and 1995.
2. Kitchenham, B.A. Pickard, L.M., and Pfleeger, S.L. Case studies for method and tool evaluation. IEEE Software vol. 12, no. 4, July 1995, pp52-62.
3. Card, D.N, McGarry, F.M. and Page, G.T. Evaluating software engineering technologies, IEEE Transactions on Software Engineering, 13(7), 1987.
4. Gilb, T. Principals of software engineering management. Addison-Wesley, 1987.
5. Walker, J.G. and Kitchenham, B.A. Quality Requirements Specification and Evaluation. in Measurement for Software Control and Assurance (B.A. Kitchenham and B. Littlewood eds.), Elsevier Applied Science, 1989.
6. Linkman S.G. DESMET Maturity Assessment, DESMET Report DES/2.2/5, 1993.
7. H.M. Parsons. What happened at Hawthorne? Science 183, pp922-932, March 1994.
8. Shari Lawrence Pfleeger, Joseph Fitzgerald Jnr. and Dale Rippy, "Using Multiple Metrics for Analysis of Improvement", *Software Quality Journal,* 1(1), March 1992.

9. Kitchenham, B.A. Pickard, L.M., and Pfleeger, S.L. Case studies for method and tool evaluation. IEEE Software vol. 12, no. 4, July 1995, pp52-62.
10. Pickard, L.M. Case Study Design and Analysis Procedures (CSDA). DESMET report WP2.2/4, 1992.
11. Yin, R.K. Case Study Research Design and methods. 2nd ed. . SAGE Publications, 1994.
12. Lee, A.S. A scientific methodology for MIS Case Studies. MIS Quarterly, March 1989, pp 33-50.
13. Glass, R.L. Pilot studies: What, why and how. The Software Practitioner, January 1995, pp 4-11.
14. Basili, V.R., Selby, R.W. and Hutchens, D.H. Experimentation in software engineering. IEEE Transactions on Software Engineering, SE-12(7),1986, pp758-773.
15. Desharnais, J-M. Analyse statistique de la productivitie des project de developpment en informatique a partir de la technique des point des fonction. Université du Quebec a Montreal, Masters thesis, 1989.
16. Kitchenham, B.A. Software Metrics for Process Improvement. NCC Publishers, 1996.
17. Cooper, B.E. Statistics for Experimentalists. Pergamon Press, 1969.
18. Siegel, S and Castellan N.J. Jr. Nonparametric statistics for the behavioral sciences. McGraw-Hill, 2nd Ed, 1988.
19. Chris Sadler, Shari Lawrence Pfleeger and Barbara Kitchenham. Trials results and validation. DESMET Report D4.1, September 1994

