Identifying the Relevant Information for Software Testing Technique Selection

S. Vegas
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo, 28660 Boadilla del Monte, Madrid, Spain
svegas@fi.upm.es

Abstract

One of the major problems within the software testing area is how to get a suitable set of test cases to test a software system. This set should assure maximum effectiveness with the least possible number of test cases. There are nowadays numerous testing techniques available for generating test cases. However, many of them are never used, while a few are used over and over again. Testers have little (if any) information about the available techniques, their usefulness and, generally, how suited they are to the project at hand. This lack of information means less tuned decisions on which testing techniques to use. This paper presents the results of developing an artefact (called a characterisation schema) to assist with testing technique selection. When instantiated for a variety of techniques, the schema provides developers with a catalogue containing enough information for them to select the best suited techniques for a given project. The schema, and its associated catalogue, assure that the decisions developers make are based on grounded knowledge of the techniques rather than on perceptions, suppositions and assumptions.

1. The Problem of Selecting Software Testing Techniques

According to Beizer [2], testing is considered as one of the most costly development processes, sometimes exceeding fifty per cent of total development costs. By one estimate [11], software consumers and organisations incur approximately US$50B in losses from defective software each year. This estimate suggests industry-wide deficiency in testing. One of the factors that influence the cost of testing is the number of test cases used. The more test cases are generated, the longer it will take to specify, execute and analyse these tests. This is why it is unworkable to run all possible combinations of input values, that is, rules out exhaustive testing [12]. Therefore, the tests are run on a relatively small set of cases, carefully chosen from the universe of system inputs. The choice of test cases is of utmost importance; as Harrold [5] claims, evaluation is a highly important process, as it is directed at assuring software quality. On the other hand, the selection needs to be tuned: the resulting set should be of minimum size, and also the set must reflect, on the basis of a small number of inputs, the behaviour of the system for the input universe.

Testing techniques are used to find a suitable set of test cases. There are nowadays some several tens of testing techniques, which raises the question of what difference there is between them. We can find distinctions in the literature as regards the mechanical (testing books) and theoretical and technical aspects (theoretical research articles, simulations and experiments) of the techniques. However, what the best suited techniques for evaluating a given system aspect are remains an open question [3]. In particular, several studies have been completed to establish differences and similarities between testing techniques –for an analysis of these studies, see [8]. However, these works do not deal with all the relevant aspects about testing techniques, neither are they exhaustive as regards the universe of testing techniques nor, in most cases, can the results of the studies be combined or compared because they use different metrics to measure one and the same aspect.

Nevertheless, testers face the question of which are the best-suited techniques every time they have to test a system. And, how is it answered at present? Neither systematically, nor following well-defined guidelines. Indeed, of the existing testing techniques, some are never considered for use at all and others are used over again in different projects without even examining, after use, whether or not they were really suited. The decisions made by developers are not so much haphazard as limited insofar as their knowledge of the techniques is. There are two main reasons why developers do not make tuned choices:

- The information available about the techniques is normally distributed across different sources of information (books, articles and even people). This

means that developers do not have an overall idea of what techniques are available and of all the information of interest about every testing technique.

- They have no access to pragmatic information concerning testing techniques unless they have used it before. Developers do not tend to share the knowledge they acquire by using testing techniques with others. This means that they miss out on the chance of learning about the experiences of others.

Therefore, the problem we face is how to identify relevant information for selecting testing techniques. Solving this problem would help testers to choose the best suited testing techniques for every project. We propose here a solution that allows developers to make the selection without being acquainted with the technique to the extent of having used it or knowing how it is applied.

The proposed solution is called a characterisation schema and is not confined to identifying useful selection criteria, but also provides an infrastructure for storing the information identified and specified for each technique. The schema fully describes the properties of the techniques according to the same pattern. The schema will be instantiated once for each technique to be included in the catalogue. Accordingly, it will be possible to build a repository containing all the techniques of interest to a given organisation.

The characterisation problem has not been specifically studied in the testing area, but there are several areas that deal with problems related to information characterisation and packaging. Even though these efforts do not fall within the testing area, they have been considered as related research, because they also aim to characterise relevant information.

Attempts have been made to characterise different software artefacts within the area of reuse. When software artefacts are reused, there is usually a reuse repository in which these artefacts are kept. Characterisations should be available for selecting the best-suited artefact from the repository. For example, Prieto-Díaz [14] or Kontio [9] have proposed characterisation schemas for reusable modules, but these proposals are not easily applicable to testing techniques. The proposal by Basili and Rombach [1] for characterising all software element types (processes, products, techniques, etc.) also falls within the area of reuse. Henninger [6] employs a similar approach, trying to capture the knowledge of software developers. However, both schemas are too generic for application to testing techniques, as they do not take into account specific testing technique characteristics.

The area of technologies selection covers work on characterising techniques, methods and tools related to software development for later use. Birk [4] proposes a generic characterisation schema for software technology selection. He also provides a series of guidelines for generating the relevant characteristics of a given technology. This schema is again too generic, which means that it is not useful for selecting testing techniques because it does not take into account their specific characteristics. Maiden suggests an approach [10] for selecting requirements elicitation techniques, which means that it cannot be used for testing techniques.

The article is organised as follows. Section 2 describes how the schema was generated. Section 3 discusses how the resulting schema is used. Section 4 shows how the schema was validated. Finally, Section 5 discusses our conclusions.

2. Building a Characterisation Schema for Testing Techniques

When deciding which information is of interest for selecting testing techniques, we could form an opinion about what information appears to be most relevant. Because of the immature theory on testing, on the basis of a subjective opinion alone, however, we cannot be sure that this information will be of interest to the people who are going to use the repository in the future or that it is really relevant for the selection process. Therefore, the process we followed to generate a schema that would reflect the relevant information for selection as accurately as possible is as follows. Firstly, we build a schema reflecting our view of the selection problem. Then, this schema is complemented with the point of view of the developers and other researchers. Finally, experts in the testing area inspect the generated schema. At this point, the schema is ready for validation. In the following, we discuss the generation of the characterisation schema stage by stage.

2.1. First iteration: theoretical schema

As mentioned above, the first step to building the characterisation schema is to generate the schema according to our own view. For this purpose, we compiled and analysed testing techniques to study the differences and similarities between them. With a view to improving information organisation within the schema, we classify the information, first, according to the different levels of which the testing process is composed and then around the elements involved in this process.

There are two separate moments during the software system testing process:

- Identification of the software system quality attributes that are to be assessed, as well as when they are to be tested, the metrics to be used for evaluation, and the parts of the system that will be affected by each test.
- Performance of each of the tests identified in the previous step, which involves generating, executing and analysing the results of the test cases, as well as

planning and specifying the environment in which the tests are to take place.

The first moment occurs when planning the control and quality assurance activities to be carried out during the project, that is, it takes place at the time when the testing tactics are defined. The second moment is spread across the entire project and corresponds to testing performance.

Based on this view of the testing process we have arranged the information the characterisation schema contains at two levels:

- *Tactical level.* The selection of the testing techniques for a project will be influenced by the goal of the test cases. The test cases needed to evaluate software system security are not the same as the tests for assessing system reliability or algorithm efficiency. The information at this level is related to the use to which the test cases are put. Test case use is determined by:
  - The *purpose* or objective of the test, which states the test aim. Thus, for example, one technique or another will be used depending on the quality attribute under evaluation and the rigour with which it is to tested.
  - The *scope* of the test, which specifies the part or parts of the system affected by the test.
- *Operational level.* This level is related to the conditions of testing technique operativeness. This means that it may or may not be appropriate to apply a given technique depending on the knowledge and experience of the personnel and whether or not the available tools are suitable. More precisely, the operational information about a testing technique is:
  - *Agents.* The people who will use the technique. It may be more advisable to use one technique than another depending on their characteristics.
  - *Technique.* Some testing technique characteristics, such as application cost or how easy it is to understand, can be relevant for decision making.
  - The *results* of applying the technique, that is, the test cases, will also have characteristics of interest for selecting a testing technique. How many test cases the technique generates or the number of repeated test cases are two such features.
  - Some characteristics of the software or *object* on which the technique is to be applied can determine the use of one technique or another, for example, the programming language used, software size, etc.

The theoretical schema generated in this first step is shown in Table 1, column labelled T.

2.2. Second iteration: empirical schema

Our view of software testing has been complemented with the opinion of producers (researchers in the testing area) and consumers (testers) who are, ultimately, the ones who will use the schema. For this purpose, we surveyed a series of producers and consumers, who were asked what information they believed to be relevant for fully describing the properties of (producers) or for selecting (consumers) a testing technique.

One of the key questions at this stage is how to decide when to stop gathering information, that is, when the set of information gathered can be considered representative of the opinion of producers and consumers. Therefore, at the same time as information was gathered, we ran a schema stability analysis. Figure 1 shows the results of this analysis, illustrating the accumulated growth speed of the empirical schema.

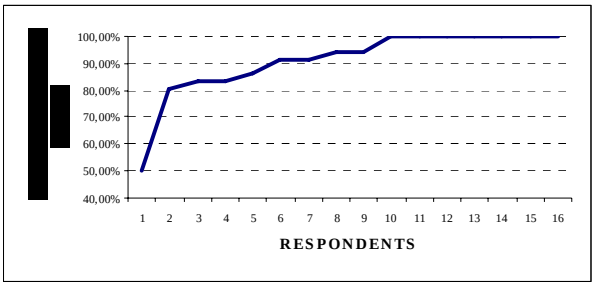

Figure 1. Schema growth speed

The x-axis of Figure 1 shows the surveyed subjects. The y-axis shows the empirical schema size as a percentage of its final size at each point. Note that the schema reaches 50% of its final size with the first respondent, rising to 80% for the second. Furthermore, we find that the schema reaches its final size with the tenth respondent. This means that the last six respondents (37.5% of the total) added no new information to the schema. Therefore, the schema could be considered stable and surveying could stop.

Table 1, column labelled E, shows the contents of the empirical schema. Note that the empirical schema provides some information that did not appear in the theoretical schema, since practitioners care about practical issues that theoreticians very often overlook. As compared with the theoretical schema, it should be mentioned that:

- A new level, called *use level*, appears, specifying subjects' earlier experiences of technique use. This level contains two elements:
  - *Project*, which specifies information regarding earlier projects in which the technique was used.
  - *Satisfaction*, which specifies what opinion the technique merits among people who have used it before.
- The *tools* attribute of the *tactical level* appears here as an element, containing information regarding the tools available for a technique.

2.3. Synthesis of perspectives: proposal of a preliminary schema

The goal of this stage is to synthesise viewpoints, creating a single schema that integrates the theoretical and empirical schemas.

A series of rules were followed to synthesise the two schemas in an orderly fashion:

1. The levels and elements of the synthesised schema will be the union of the levels and elements of the original two schemas.
2. Any attributes that appear in just one of the characterisation schemas will appear unchanged in the synthesised schema.
3. Any attributes that appear in both schemas and are equal[1] will appear unchanged in the synthesised schema.
4. Any attributes that appear in the two schemas and are similar[2] will be studied to decide whether they are used to generate one or several attributes.
5. In no case will information be deleted from the characterisation schema.

Table 1 shows the results of the synthesis, specifying the source of each schema attribute. The column labelled T indicates the attributes from the theoretical schema and column E designates the attributes from the empirical schema.

It is interesting to note that 14 of the attributes present in the preliminary schema did not appear in the theoretical schema. On the other hand, there are only two attributes that are present in the preliminary schema and not in the empirical schema. In other words, 58% of the attributes of the preliminary schema are common to the original two schemas; of the other 42%, the theoretical schema supplies 5% and the empirical schema 37%. Note that this is understandable bearing in mind that the empirical schema is the fruit of the opinions of 16 individuals (both practitioners and researchers), whereas the theoretical schema is generated exclusively on the basis of our own information. It is no less true, however, that the information used to generate the theoretical schema was gathered by examining the differences and similarities between techniques, as well as data found in textbooks regarding the characteristics on which testing technique selection is based, which would appear to be a fairly comprehensive set of sources.

The major omissions of the theoretical schema are pragmatic aspects: the *use* level and the *tools* element. Minor omissions are some attributes of the *technique* element (*maturity level*, *inputs* and *data cost*) and an attribute of the *object* element (*size*). The empirical schema, on the other hand, has theoretical omissions, namely, two attributes of the *results* element (*completeness* and *adequacy degree*).

Table 1. Synthesised schema

LEVEL	ELEMENT	ATTRIBUTE	T	E
Tactical	Objective	Quality attribute	■	■
		Rigour	■	■
	Scope	Phase	■	■
		Element	■	■
		Aspect	■	■
Operational	Agents	Experience	■	■
		Knowledge	■	■
	Tools	Identifier	■	■
		Automation		■
		Cost		■
		Environment		■
		Support		■
	Technique	Comprehensibility	■	■
		Maturity level		■
		Cost of application	■	■
		Inputs		■
		Adequacy criterion	■	■
		Test data cost		■
		Dependencies	■	■
		Repeatability	■	■
		Sources of information	■	■
	Results	Completeness	■	■
		Correctness	■	
		Effectiveness	■	■
		Type of defects	■	■
		# of generated cases	■	■
		Adequacy degree	■	
	Object	Software type	■	■
		Software architecture	■	■
		Programming language	■	■
		Development method	■	■
		Size		■
Use	Project	Reference projects		■
		Tools used		■
		Personnel		■
	Satisfaction	Opinion		■
		Benefits		■
		Problems		■

2.4. Schema improvement: expert peer review

After synthesising the schemas, we thought it was advisable to send the resulting schema, along with a questionnaire, to a series of experts (of whom four replied) in the testing area for them to express their opinions on form and content. The opinions on form include issuing judgements about the suitability of schema organisation or the names that appear in the schema. The opinions on content include issuing judgements about the existence of possible redundancies, missing information, etc., in the schema.

Additionally, the expert responses were analysed by

[1] Two attributes are considered equal if they bear the same name and belong to the same element and level.

[2] Two attributes are considered similar if they do not bear the same name or do not belong to the same element or same level, although they represent the same or similar concepts.

checking whether there were contradictory or coincident before finally making a decision on whether or not to accept the suggestion, and if so, how to add the suggestion to the schema. Indeed, a series of rules were followed to decide whether or not to accept the expert suggestions:

1. If the experts disagree, the majority view will be respected.
2. If more than one expert recommends a given change, the recommendation will be taken into account.
3. If only one expert recommends a change, this change will be accepted provided the proposed change is not due to a misinterpretation of the schema, its logic or its content. In other cases, however, a change is not always as evident when it is recommended by one rather than several experts. Then, it is the expert's versus our opinion. It is sometimes impossible to reconcile the two viewpoints, and it was decided that our opinion should take precedence, as it is contradictory to make modifications in which we do not believe or about which we were not sure.

Table 2 shows the results of the expert peer review, leading to the final schema.

The changes made owing to the experts' suggestions were:

- Five attributes have been deleted: three from the *tactical* level (*quality attribute*, *rigour* and *phase*) and two from the operational level (*maturity level* and *adequacy degree*), since the experts found they were redundant.
- The *correctness* attribute of the *operational* level was replaced by another named *precision*, which was more meaningful according to the experts.
- Two attributes were moved from the *operational* level to the *tactical* level (*effectiveness* and *defect type*), since, according to the experts, these attributes refer to information that should be known as soon as possible in the selection process.
- A new attribute, termed *purpose*, was created and placed in the *tactical* level, since the experts agreed that this information was necessary, and missing from the schema.
- The *results* element was renamed as *test cases*, which, according to the experts was more meaningful.
- The *use* level was renamed as *historical* level, again, with the aim of making it more meaningful.

3. Selecting Testing Techniques through the Proposed Schema

Having presented the schema, we now prescribe the procedures associated with the use and evolution of the repository. The repository will be used directly by pro-

Table 2. Final schema

LEVEL	ELEMENT	ATTRIBUTE	DESCRIPTION
Tactical	Objective	Purpose	Type of evaluation and quality attribute to be tested in the system
		Defect type	Defect types detected in the system
		Effectiveness	What capability the set of cases should have to detect defects
	Scope	Element	Elements of the system on which the test acts
		Aspect	Functionality of the system to be tested
Operational	Agents	Knowledge	Knowledge required to be able to apply the technique
		Experience	Experience required to be able to apply the technique
	Tools	Identifier	Name of the tool and the manufacturer
		Automation	Part of the technique automated by the tool
		Cost	Cost of tool purchase and maintenance
		Environment	Platform (sw. and hw.) and programming language with which the tool operates
		Support	Support provided by the tool manufacturer
	Technique	Comprehensibility	Whether or not the technique is easy to understand
		Cost of application	How much effort it takes to apply the technique
		Inputs	Inputs required to apply the technique
		Adequacy criterion	Test case generation and stopping rule
		Test data cost	Cost of identifying the test data
		Dependencies	Relationships of one technique with another
		Repeatability	Whether two people generate the same test cases
		Sources of information	Where to find information about the technique
	Test cases	Completeness	Coverage provided by the set of cases
		Precision	How many repeated test cases the technique generates
		Number of generated cases	Number of cases generated per software size unit

	Object	Software type	Type of software that can be tested using the technique
		Software architecture	Development paradigm to which it is linked
		Programming language	Programming language with which it can be used
		Development method	Development method or life cycle to which it is linked
		Size	Size that the software should have to be able to use the technique
Historical	Project	Reference projects	Earlier projects in which the technique has been used
		Tools used	Tools used in earlier projects
		Personnel	Personnel who worked on earlier projects
	Satisfaction	Opinion	General opinion about the technique after having used it
		Benefits	Benefits of using the technique
		Problems	Problems with using the technique

ducers and consumers and indirectly by the librarian. Producers will be able to provide new information for the repository, extracted from the results of their research on developing new and studying the applicability conditions of existing techniques. Producers will also use the repository contents to find new research goals or lines. Consumers, likewise, will be able to select testing techniques for the projects on which they are working, making the selection based not on their knowledge but on the information contained in the repository (which has been generated by successive instantiations of the characterisation schema proposed here for different techniques). The repository will provide both a complete set of techniques and technique information that is relevant for selection purposes. Consumers will also be able to provide feedback on the contents and the actual structure of the schema from the results of repository use and the techniques selected, extracted from their experiences using the techniques. Finally, the librarian will update the repository on the basis of the information supplied by producers and consumers, taking care to maintain the coherence of the information it contains. There are, then, five procedures, each associated with repository use, which will also permit its evolution. Figure 2 illustrates these uses.

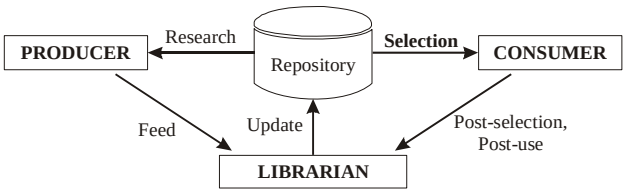

Figure 2. Schema use

Below, we go deeper on what we consider to be the primary schema use: selection. Two concepts need to be introduced to be able to explain this process:

- *Bounded variables.* These are schema attributes whose value is imposed by the project and cannot be changed during selection. For example, the project development method, the type of software under development, etc.
- *Free variables.* These are schema attributes whose value can be changed depending on the current selection needs and/or preferences. For example, the characteristics of the people who are to apply the testing techniques or the tools to be used are not necessarily pre-established by the project in question.

The steps for **schema use** by consumers for selection purposes are:

1. *Determination of bounded variables.* Identify the desired technique values for the attributes that belong to the tactical level and the object element of the schema. Also, identify whether there are other attributes that could impose any sort of constraint on the technique to be chosen (for example, whether given personnel should be used).
2. *Preselection of an initial set of techniques* .Compare the desired values of the attributes with the values of each technique the repository contains. Preselect the techniques whose values match the specified values.
3. If the set of preselected techniques is empty, relax one of the constraints and return to step 3.
4. Examine the remaining attribute values of the preselected techniques, paying special attention to the *dependencies* attribute.

An example of selection is shown later in Section 5.1.

4. Validation of the Proposed Solution

To validate the characterisation schema we ran two evaluations: an empirical evaluation, which assessed the static aspects of the schema, and an experimental evaluation, which assessed the dynamic aspects of the schema.

4.1. Empirical evaluation

The objective of this evaluation is to check schema feasibility from the producer and consumer viewpoints, as well as to evaluate schema flexibility. In other words, the primary goal of this evaluation is to find out whether it is possible to locate the information the schema contains, how this information can be used during selection and whether it is possible to instantiate the schema for any existing testing technique. Each of these points is discussed below.

One testing technique was instantiated to check **feasibility from the producer viewpoint**. The chosen technique was *decision coverage*, from the control flow technique family. The schema instantiation for this

technique is shown in Table 3.

The main finding from this instantiation is that the schema is feasible from the producer viewpoint. Additionally, there is information that is difficult to find, especially information related to reference projects. This is because organisations do not like to see their private data published, if they actually keep a record of these data. Also, there were two schema attributes (precision and completeness), whose value was not found. However, they are found in both the theoretical and empirical schemas and the experts did not consider them unsuitable. This appears to be relevant information that is not available in the literature on testing techniques. So, it is information considered relevant from all viewpoints (note that there are not many attributes in the schema of which this can be said), but is, however, not easy to locate. In addition, information about the testing techniques is sometimes contradictory. This may be due to the fact that the knowledge on testing techniques is not yet mature enough and assertions are sometimes made overlooking some key parameters

Table 3. Decision coverage technique

L	ELEM.	ATTRIBUTE	VALUE
Tactical	Objective	Purpose	Find defects
		Defect type	Control
		Effectiveness	48%
	Scope	Element	Units
		Aspect	Any
Operational	Agents	Knowledge	Flow graphs
		Experience	None
	Tools	Identifier	LOGISCOPE
		Automation	Obtain paths
		Cost	€3,000-6,000
		Environment	Windows; C/C++
		Support	24 Hot-line
	Technique	Comprehensibility	High
		Cost of application	Low
		Inputs	Source code
		Adequacy criterion	Control Flow
		Test data cost	Medium
		Dependencies	Supplemented with techniques that find processing errors.
		Repeatability	No
		Sources of information	Sommerville
	Test cases	Completeness	--
		Precision	--
		# of generated cases	Exponential # decisions
	Object	Software type	Any
		Software architecture	Any
		Programming language	Any
		Development method	Any
		Size	Medium
Historical	Project	Reference projects	--
		Tools used	--
		Personnel	--
	Satisfaction	Opinion	OK, but should be complemented with others
		Benefits	It is easy to apply
		Problems	Dynamic analyser should be avoided when used with real time and concurrent systems due to code instrumentation

that limit the applicability of the claim in question. Finally, the metrics used to fill in some attributes are not easy to interpret, such as the metric associated with effectiveness, which is *probability of detecting a fault*. Can this metric be considered valid for specifying how many faults a technique can detect? Would not the *percentage of faults that the technique can detect* be a better choice?

To check **schema flexibility,** we decided to select a number of technique families, which covers the variety of techniques between families, and a number of techniques within each family, which covers the variety of techniques within each family. Additionally, we opted for well-known techniques, as this gives a better understanding of how the schema is instantiated. The chosen techniques were:

- *Functional techniques.* Boundary value analysis and random testing.
- *Control-flow techniques.* Sentence coverage, decision coverage, path coverage and threads coverage
- *Data-flow techniques.* All-c-uses, all-p-uses, all-uses, all-du-paths and all-possible-rendezvous.
- *Mutation:* Standard mutation and selective mutation.

We were able to instantiate all these techniques (see [16]). Of course, this does not mean that the schema is totally flexible, as it would be necessary to instantiate the schema for *every* existing testing technique for this purpose. However, the fact that we were able to instantiate a series of techniques that are representative of existing techniques without difficulty indicates that the schema is flexible enough to be able to instantiate the majority of, if not all, testing techniques.

We performed a selection for a particular project to check **feasibility from the consumer viewpoint**. For this purpose, we followed the steps defined in Section 4. The problem posed was:

> A car park management system (concurrent system) is to be built. At this stage of the project, the QA team has identified the key quality attributes of this software system. These were obtained by examining the characteristics of the software under development, as well as its application domain. In this particular case, the essential attributes are: correctness, security and timing.
>
> The project situation is as follows. The system is to be coded in Ada, the development team is fairly experienced in developing similar systems, and almost all the errors they are found to make are typical of concurrent programming. The testing team is also experienced in testing this type of systems.

It was solved as follows.

1. *Determination of bounded variables*, shown in Table 4.

Table 4. Bounded variables

LEVEL	ELEMENT	ATTRIBUTE	VALUE
Tactical	Objective	Purpose	Find faults
		Defect type	ANY
		Effectiveness	>50%
	Scope	Element	ANY
		Aspect	ANY
Operational	Object	Software type	Real time
		SW architecture	Concurrent
		Program. language	Ada
		Develop. Method	ANY
		Size	Medium

2. *Pre-selection of an initial set of techniques.* The values of the bounded variables identified in the previous step were compared with the technique values contained in the repository. The techniques selected
   after situation/technique matching are: *boundary value analysis*, *random*, *path coverage*, *all-possible-rendezvous*, *all-c-uses*, *all-p-uses*, *all-uses*, *all-du-paths*, *standard mutation* and *selective mutation.* The *sentence coverage* and *decision coverage* techniques are rejected because their effectiveness is low, and the technique *threads coverage* is discarded because it is for object-oriented software.
3. *Identification of the best-suited techniques for selection.* Of the pre-selected techniques, there is one that is specific for Ada-style programming languages. Although there are general-purpose techniques that are more effective, the technique that is specific for concurrent software appears to detect the faults proper to concurrency better than the other techniques. Furthermore, the *path coverage* technique states that when used with concurrent and real-time systems, a dynamic analyser cannot be used as a tool. Additionally, the techniques *all-c-uses*, *all-p-uses*, *all-uses*, *all-du-paths*, *standard mutation* and *selective mutation* cannot be used without a tool (which is not available in the situation under consideration). Therefore, the *all-possible-rendezvous* technique is selected. However, the dependency attribute states that the technique should be supplemented with a black-box technique. Observing the black-box techniques in the pre-selected set (*boundary value analysis* and *random*), it is found that the *random testing* technique is useful for people with experience in the type of tests to be run and is, therefore, also selected.

From this, we conclude that the schema is feasible from the consumer viewpoint, as at least one selection has been able to be made.

4.2. Experimental evaluation

The objective of this evaluation is to check schema completeness, effectiveness, efficiency, usability and user satisfaction from the viewpoint of the consumers in all cases. The primary aim of this evaluation is to try to understand how schema use affects the testing technique selection process. That is, whether the schema is really an improvement on selection using other resources (basically books) or, contrariwise, it is preferable to carry on using the traditional selection process, because the schema increases the workload; and whether the schema is of assistance to consumers in the sense that it improves the work they do. To assure that the comparison between the schema and books was as balanced as possible, the subjects who participated in the experiment have worked with a catalogue on paper rather than the automated repository in the shape of a tool.

The null **hypotheses** of this experiment are:

H_{01}: The efficiency of the selection process is independent of the method used for the purpose and the project in question.

H_{02}: The usability of the method of selection is independent of the method used for the purpose and the project in question.

H_{03}: The completeness of the original set of information for making the selection is independent of the method used for the purpose and the project in question.

H_{04}: The effectiveness of the selection process is independent of the method used for the purpose and the project in question.

The fifth aspect, user satisfaction, will not be considered for establishing a hypothesis that can be refuted by means of statistical evidence. This aspect will be assessed informally, examining the opinions of each subject.

The experiment **parameters** are the characteristics that either do not influence or are not intended to influence the result of the experiment [7]. Table 5 reflects the parameters and the assigned values for this experiment.

Table 5. Parameters for the experiment

PARAMETER	VALUE
Subject experience	Novice
Task to be performed	Testing technique selection
Documentation	Requirements Project context
Test	Unit
Attribute	Correctness

Below, we will describe the variables whose value varies, because it is of interest to observe how they affect the response variable [7]. There are two **factors** in this experiment: the selection method and the software project:

- *Method of selection.* This factor has two possible alternatives: books and schema. For selection using books, the subjects will be given three books, which will be the only ones they are allowed to use. The books used are the highly reputed [2], [13] and [15].

For selection using the schema, the subjects will be given the repository discussed above and will follow the process defined in Section 3.

- *Software project.* This factor will encompass all the project characteristics, from the software system, through time and financial project constraints, to personnel, etc. The four software projects chosen are: a video club management system (M), a bank loan approval system (B), a car park control system (S) and a system for monitoring pump water level (RT).

The **response variables** are the experiment outputs and reflect the relationships between the different factor levels [7]. The response variables of this experiment, shown in Table 6, will be gathered by means of nine forms designed for the purpose.

The experiment was run with 87 subjects, who were divided into four groups. The **experimental design** is discussed below. Table 7 shows how the methods of selection were assigned to groups.

From Table 7, we can see that the subjects of group 1 and 3 will make both selections with the schema and books, respectively. Groups 2 and 4 will make the selection first with books (schema) and then with the schema (books). According to this distribution, groups 2 and 4 can be used to examine the effect of the method of selection (preference was given to group 2 because it represents reality), and the purpose of groups 1 and 3 is to assess the learning effect on these inexperienced subjects.

Table 6. Experiment response variables

ASPECT	RESPONSE VARIABLE
Completeness	Amount of info used during selection Sort of the info used during selection Amount of info missed during selection Sort of the info missed during selection
Usability	# problems found during selection Description of the problems found # times schema help consulted Attributes consulted in help
Effectiveness	# techniques considered during selection # techniques selected Techniques selected
Efficiency	Time spent studying the techniques Selection time Time spent consulting doubts about schema
User Satisfaction	Advantages and disadvantages of schema use Would you be willing to use it? What improvements you would make? What did you like or not like? Has your view of selection changed? What have you learnt? Would you do things differently next time? Suitability of attribute names and distribution

Table 7. Assignation of selection methods

	Group 1 (18 pers.)	Group 2 (33 pers.)	Group 3 (18 pers.)	Group 4 (18 pers.)
Selection 1	Schema	Books	Books	Schema
Selection 2	Schema	Schema	Books	Books

Continuing with the experimental design, Table 8 shows how projects were assigned to subgroups. Eight subgroups, from A to H, were set up in each group.

Table 8. Assignation of projects

	Selection 1				Selection 2			
REP.	**M**	**B**	**S**	**RT**	**M**	**B**	**S**	**RT**
A	X	-	-	-	-	-	X	-
B	X	-	-	-	-	-	-	X
C	-	X	-	-	-	-	X	-
D	-	X	-	-	-	-	-	X
E	-	-	X	-	X	-	-	-
F	-	-	X	-	-	X	-	-
G	-	-	-	X	X	-	-	-
H	-	-	-	X	-	X	-	-

Two different statistical methods were used to **analyse the data** collected during the experiment: analysis of variance (ANOVA) and analysis of simple correspondences, used for quantitative and qualitative variables, respectively. ANOVA is used to study the relationships between a quantitative response variable and one or more qualitative factors [7]. Its objective is to determine whether the differences between the means of the response variable in the groups established by the combinations of factor levels are statistically significant. The complete data collected from the experiment can be found in [16]. Next, the obtained results are summarised.

Schema efficiency has been studied with the three response variables that appear in Table 6. The experiment showed that the schema helps to reduce both the learning and the selection time as compared with books and that the time spent consulting the schema can be considered negligible with respect to the other two. Moreover, the total time required to solve the selection problem is the sum of the learning time, plus the selection and consultation time (which is zero if books were used for selection). Accordingly, it can be concluded that the characterisation schema makes selection more efficient.

Schema usability has been examined on the basis of the four response variables shown in Table 6. These variables can be used to compare the schema and books and to assess the schema. From the comparison, it was possible to deduce that the subjects have fewer problems using the schema than books and the frequency of appearance of each problem is lower with the schema. The main problem encountered by the subjects using the schema is the presence of uninstantiated attributes. The main problem of the subjects using books is the poor organisation of the available information, as well as missing information of interest and the existence of information that is unnecessary for selection purposes. From the schema assessment, it was possible to deduce that consultations of the meaning of attributes are generally low, and the attributes that represent concepts that are not intuitive or are difficult for the subjects to interpret are the ones most often consulted. From this, it

can be deduced that characterisation schema usability is acceptable, although it could be improved by building a tool to assure that all the information is entered.

The response variables used to examine **schema completeness** appear in Table 6 and study both the information used by the subjects during selection and missing information. The main finding here is that it is important for the characterisation schema to be completely instantiated for it to be useful to consumers. Another interesting point observed is that subjects are not always able to ascertain information that does not appear in, but can be easily deduced from, the schema, such as the time it will take to apply the technique. We can get an idea of this value, if we know the complexity of the technique, the people, the tool and software size.

The response variables used to examine **schema effectiveness** appear in Table 6. It was found that the original number of techniques used to make the selection is lower for books than for the schema and varies from subject to subject; the number of selected techniques is lower for the schema than for books; and the subjects using books select families of techniques, opting for a technique of the family if they are very well acquainted with it. This means that the subjects are unable to distinguish a technique from a family. This means that books are confusing as regards the information they provide. This could be the reason why the subjects tend to select more techniques and techniques with which they are very familiar. Finally, it should be stressed that the schema leads to more precise selections.

With respect to **user satisfaction**, the subjects can be said to like the schema. They would be prepared to use (but not instantiate) it if given the opportunity, although they consider it contains too much information. They view the fact that there are uninstantiated attributes or that they have to handle so much information as a drawback.

As mentioned above, we have studied whether there is **learning** in groups 1 and 3 and the **influence of the method** in groups 2 and 4.

- *Learning effect*. We found that it affects schema efficiency in these two groups. This means that as subjects become better acquainted with the schema, their selections are faster. We also found that group 3 was delighted with the prospect of using the schema, whereas group 1 failed to appreciate the schema, as they had not had to undertake a selection without it.
- *Influence of the method*. Using one method before another affects usability. This means that usability was better for books in the group that used first the schema and then books. In relation to what we discussed above, we also found that group 2 were delighted with the schema and group 4 made a more guided selection, as they used the schema for the first selection and books for the second.

5. Conclusions

The main problem met by software developers when choosing the best suited testing techniques for a software project is information. The information on testing techniques is, at best, distributed across many sources of information and, at worst, non-existent. The approach we have taken in this research to the problem of gathering relevant information about software testing techniques is called a characterisation schema. Using this schema, we can build a repository that contains the description of each technique of interest and describes all techniques according to the same pattern so that a decision can be made on whether or not to use a technique without having procedural knowledge of the technique.

An empirical and iterative process has been followed to search for the information that such a characterisation schema should contain. This process is empirical because it takes into account the opinions of a variety of people involved in software testing and iterative because it is gradually refined as new opinions are added to the schema.

Finally, the generated schema has been validated in two ways. First, it has been instantiated for several testing techniques, by means of which we were able to test the schema for several techniques and check that it is possible to find the required information. Secondly, an experiment was run to check its behaviour against the use of other methods of selection, such as books. This experiment found that schema use is more efficient and complete than the use of books for selection purposes and that selection is less problematic. However, we were not able to demonstrate schema effectiveness, as the current state of testing technique selection is less stable than it was thought to be. We were, however, able to deduce that selections made using the schema are finer tuned than when using books.

References

[1] V.R. Basili and H.D. Rombach. Support for comprehensive reuse. *Software Engineering Journal*, 303-316, September 1991

[2] B. Beizer. *Software Testing Techniques*. International Thomson Computer Press, Second Edition, 1990

[3] A. Bertolino. Guide to the knowledge area of software testing. Software Engineering Body of Knowledge, February 2001.

[4] A. Birk. Modelling the application domains of software engineering technologies. *Proceedings of the Twelfth International Conference on Automated Software Engineering (ASE).* Lake Tahoe, California, November 1997.

[5] M.J. Harrold. Testing: A roadmap. *Proceedings of the 22nd International Conference on the Future of Software Engineering*, 63-72, Limerick, Ireland, May 2000

[6] S. Henninger. Accelerating the successful reuse of problem solving knowledge through the domain lifecycle. *Proceedings of the Fourth International Conference on Software Reuse*, 124-133, Orlando, Florida, April 1996.

[7] N. Juristo, A.M. Moreno *Basics of Software Engineering Experimentation.* Kluwer Academic Publishers, 2000

[8] N. Juristo, A.M. Moreno, S. Vegas. A Survey on Testing Technique Empirical Studies: How Limited is our Knowledge. *Proceedings of the International Symposium on Empirical Software Engineering*, Nara, Japan, October 2002.

[9] J. Kontio, G. Caldiera, V.R. Basili. Defining Factors, Goals and Criteria for Reusable Component Evaluation. *Proceedings of the CASCON'96 Conference*. Toronto, Canada, November 12-14, 1996.

[10] NAM Maiden and G. Rugg. ACRE: Selecting methods for requirements acquisition. *Software Engineering Journal.* 11(3):183-192,1996.

[11] RTI. The Economic Impact of Inadequate Infrastructure for Software Testing. Planning Report 02-3, National Institute of Standards and Technology. May 2002.

[12] G.J. Myers. *The Art of Software Testing.* Wiley-Interscience, 1970

[13] S.L. Pfleeger. *Software Engineering: Theory and Practice*, Mc-Graw Hill. 1999

[14] R. Prieto-Díaz. *Software Reusability*, Vol 1, Chapter 4. Classification of Reusable Modules, 99-123. Addison-Wesley, 1989.

[15] I. Sommerville. *Software Engineering, 5th edition.* Pearson Education. 1998.

[16] S. Vegas. *A Characterisation Schema for Selecting Software Testing Techniques.* PhD Thesis. Facultad de Informática, Universidad Politécnica de Madrid. February 2002.