Software Evaluation Problem Situations

Ioannis Stamelos(+), Alexis Tsoukiàs(*)
(+) Dept. of Computer Science, Aristotle University of Thessaloniki
54006, Thessaloniki, Greece
e-mail: stamelos@csd.auth.gr
(*) LAMSADE - CNRS, Université Paris Dauphine
75775 Paris Cedex 16, France
e-mail: tsoukias@lamsade.dauphine.fr

Abstract

Evaluating software is a complex decision aiding activity which requires the recognition of the problem situation in which the evaluation is requested, the establishment of a set of problem formulations which represent the client's "problems" and, for a specific problem formulation, the construction of an evaluation model which indicates how such an evaluation will be performed.

In this paper the first two aspects of such a process are discussed, providing a partial list of software evaluation problem situations, how they are characterised and what problem formulations they allow. Moreover two real case studies, concerning software evaluation, are briefly presented and discussed under this point of view (how the problem is perceived and formulated).

Keywords: Software Evaluation, Problem Formulation, Multiple Criteria Methodology.

1 Introduction

Software evaluation is an increasingly important problem in any sector of human activity. Industrial production, service provisioning and business administration heavily depend on software which is more and more complex, expensive and difficult to maintain.

In the present practice, evaluating software is seen as a simple process consisting of adopting a quality model, measure some attributes and then aggregate them following the quality model. Very little attention to the motivations of software evaluation, the purposes and the evaluation process itself is paid. The need for evaluating software may have different origins. Moreover various components of the software itself, its production process and its maintenance may be involved. The evaluation may be done for various purposes. Different actors may be involved in the evaluation process carrying different interests, problems and points of view. Evaluating software therefore is not a simple technical (engineering) activity aiming to define an "objectively good software", but a decision process where subjectivity, uncertainty and possibility for conflicts are present without any possibility of arbitrary reduction. Under this perspective, the recently proposed IUSWARE methodology (see [Morisio and Tsoukiàs 1997]) presents an outline of how software evaluation can be modelled under a decision aiding methodology.

However, from the point of view of a software engineer, it is not always clear what consequences the different "problem situations" may generate to his/her evaluations. The concept of "problem situation", although introduced by Roy ([Roy 1996]) and used by Morisio and Tsoukiàs ([Morisio and Tsoukiàs 1997]), remains a methodological, but not an operational concept.

Up to now, the research effort has focused on the solution of specific software evaluation problems ([Mosley 1992], [Poston and Sexton 1992], [Zahedi 1990], [Kontio 1996], [LeBlank and Jelassi 1994]). Moreover, in general, no particular attention is paid to the context of the problem and to the impact this context has on the evaluation approach, thus narrowing the scope of the examined problem. In the field of software cost estimation, certain criteria have been proposed for the evaluation of software cost models. Other work ([Kitchenham 1987, Meskens 1994, Park and Lim, 1999]) is related to the evaluation of a single software attribute (normally the quality or some quality sub-attribute for a software product) or the evaluation of the result of a software development activity ([Cardenas-Garcia and Zelkowitz 1991], [Giakoumakis and Xylomenos 1996]). The paper is part of a large project applying the Multiple Criteria Decision Aiding methodology in software eval-

uation (see [Morisio and Tsoukiàs 1997, Vlahavas et al. 1999, Stamelos et al. 2000, Paschetta and Tsoukiàs, 2000, Blin and Tsoukiàs 2001]

The aim of this paper is to go deeper in analysing the contents of different "problem situations" suggesting a basic classification of "software evaluation problem situations", mainly based on the experience of the first author in the telecommunication industry. The reasons for such an analysis is twofold:
- to provide some hints to analysts involved in a software evaluation about how their analysis would change assuming a specific problem situation;
- to provide a knowledge base for the construction of intelligent front end tools in software evaluation support systems (of the type described for instance in [Boloix and Robillard 1995] or in [Vlahavas et al. 1999, Stamelos et al. 2000]).

The main idea is that the same problem situation may contain different possible problem formulations (for each of which different evaluation models can be conceived). An appropriate perception of the problem situation may help in defining a suitable set of problem formulations among which the client and/or the evaluator may choose the one better fitting his/her necessities.

Similar concerns as ours are presented in [Basili 1995]. In that paper, the Goal-Question-Metric (GQM) paradigm for software evaluation through metrics is discussed. This approach focuses on the establishment of links between the managerial and technical goals of a specific project and the software measurements needed to provide the necessary information for the project. These links are modelled by a directed graph, where the entry points are goals, the intermediate nodes are questions and the exit points are metrics. The GQM approach is well-suited for the generation of problem descriptions and the respective constructive evaluation models. However, evaluation problem descriptions are non-formal (templates are used) and the focus is mainly on the necessary measurements rather than on the analysis of the problem situation and the underlying problem element associations.

The paper is organised as follows. Section 2 presents a brief description of the "problem situation" and "problem formulation" concepts as introduced in the IUSWARE approach (for more details on the general decision support framework to which IUSWARE belongs the reader might see [Tsoukiàs, 1997, Bouyssou et al., 2000], but also [Beroggi, 1999]). Section 3 presents a set of different "software evaluation problem situations" characterised by the two concepts introduced in the previous section. Section 4 presents two real case studies in which different problem formulations have been simulated during the decision process in order to clarify and better understand the evaluation model under construction. Section 5 presents a discussion of how the concepts introduced in the paper could be used oper-

ationally. Some future research directions are included in the conclusions.

2 Problem Situation and Problem Formulation

Following the IUSWARE (see [Morisio and Tsoukiàs 1997]) terminology, a problem situation PS is a triplet $\langle A_{PS}, O_{PS}, RS \rangle$ where:
A_{PS}: are the actors involved;
O_{PS}: are the objects (problems, interests, opportunities, stakes) introduced by each actor (for instance in buying a software an actor may be concerned with the software maintenance, while another actor may be concerned by the organisation changes the software may induce);
RS: are the resources allocated by each actor to each object concerning him/her (as for instance knowledge, money, decision power etc.).

For a problem situation and for a given time and a given client (decision maker), one or more problem formulations Γ can be defined, which are triplets $\langle A_\Gamma, V_\Gamma, \Pi_\Gamma \rangle$, where:
A_Γ is the set of potential alternatives (actions) to consider;
V_Γ is the set of points of view or dimensions under which the evaluation has to be done;
Π_Γ represents the scope of the evaluation expressed in a more or less formal way (a problem statement). Typical problem statements include, but are not limited to, choice, ranking, absolute evaluation, rejection, description etc..

A problem formulation therefore translates the concerns expressed in the problem situation model in a "formal" problem on which it is possible to apply some techniques such as statistics, measurement, operational research, simulation, etc.

Usually A_Γ results from specific suggestions done by the client and/or other actors. It may also result as a consequence of the actors' concerns, but in such a case the alternatives thus conceived are rather hypothetical than real actions to be undertaken.

The actors' concerns are usually transformed in V_Γ. Such a process (normally supported by an analyst) combines the intuitive knowledge of the client (and/or the other actors) about the problem situation, the domain knowledge of the actors involved in the decision process and the methodological knowledge of the analyst. Just to give an example in a software acquisition situation, a concern of the client might be "cost reduction for

the company"; the analyst might derive and define with the client two separable points of view: real costs and software productivity (estimated savings due to the new software). Such points of view might be further analysed in precise evaluation dimensions and criteria, but this is relevant to the construction of an evaluation model.

Finally the problem statement belongs to the client. As soon as the set A_Γ is established the analyst should define with the client the use of such a set and what it is expected as a result. The evaluation model will become a resource for the client in the problem situation where such a model is going to be used. It is "something" the client "needs" and is consequently established.

Once a problem formulation is established an evaluation model $\mathcal{M}$ can be built, consisting of a n-uple $\langle A^*, D, M, E, G, U, R\rangle$ where:
- A^* are the alternatives to be evaluated under this specific model;
- D are the evaluation dimensions considered (usually an hierarchy);
- M and E are the available measures at the leaves of the dimensions' hierarchy (possibly at the intermediate nodes as well) and the associate scales (metrics);
- G are the criteria defined in order to evaluate a decision about the set A^*;
- U is the uncertainty (if any) associated to the available information;
- R are the aggregation procedures used in order to aggregate the measures (where necessary) or the preferences (where necessary).

Problem Situation, Problem Formulation and Evaluation Model can be seen as results of a process (a decision process) at three different steps. Very little is known however on how such a process is structured and whether there are any invariants in this process. Such information could facilitate the identification of specific support tools for analysts and/or decision makers (clients) involved in decision processes. Our paper aims to contribute in such a direction providing a (non exhaustive) list of problem situations and relative problem formulations. The list is obtained on an empirical basis.

3 Problem Types

Typical situations of software evaluation include the following (the list not being exhaustive):

- keep or change;

- make or buy;
- commercial product evaluation;
- tender evaluation;
- software certification;
- software process evaluation;
- software system design selection.

In order to characterise such different problems we will use the problem situation concept and try to outline a problem formulation for each.

3.1 Keep or change

This situation arises when a particular software product is already in place and, due to certain business needs, it is questioned whether it is still valid or it should be replaced by a new product. Such business needs may be caused from pressure exercised by a competitor, new technology or market growth, and may lead to such requirements as additional complex functionality, management of new types of corporate data, interfaces with external devices and systems, and increased capacity in terms of number of users, volumes of customers, etc.

This kind of problem has a lot of similarities with the make-or-buy situation (see next sub-section) since, if the keep decision is taken, all necessary additional features will be implemented on the existing product. This decision means that the ability to develop software of adequate quality exists in the company. If the change decision is taken, interaction with one or more external suppliers will inevitably start. A supplier will procure the new software product and will provide support for the installation and training. Most probably a new hardware platform will be also needed (or an upgrade of the existing one). Another external supplier (could be the procurer) will undertake the customisation and adaptation of the product in the company. If the problem is very important to the company, in order to make the correct decision, the help of a consultant may be needed.

Activities related to this problem include benchmarking of the existing and new products and comparison of the former against the latter, as well as evaluation of development against acquisition cost and cost/benefit analysis.

A_{PS} Is a set of actors mainly limited only to elements (people, departments) internal to the organisation interested in the software product. External actors may be consultants.

O_{PS} Are strategic goals of the organisation, power distribution in the organisation.

RS Are knowledge, information, time, power.

A Is a set of possible decisions: {keep, change, deeper analysis}. However, the change option may contain more than one alternative, if more change scenarios are possible.

V The evaluation dimensions represent the various strategic goals of the organisation affected by such decision and the various evolution scenarios of the organisation and its context. Different alternatives are always evaluated as different scenarios.

Π The scope of the evaluation may be either a constructive description of the different scenarios, or a choice among them.

3.2 Commercial product evaluation

In this typical case, a number of commercially available software products must be evaluated. After the acquisition of one of them, customisation may be necessary.

A_{PS} In this case, besides the internal actors (internal client, acquisition manager, legal adviser, etc.), certain external actors may be present such as consultants, potential suppliers, legal authorities (as in the case of acquisitions in the public sector) and others.

O_{PS} Besides any individual interests of the participating actors, a corporate objective should be identifiable (why and what to buy) and an acquisition policy (vendor rating, co-makership, etc.).

RS An acquisition budget should be included among the other resources, even if it is not sure whether an acquisition will effectively occur.

A Is a set of offers concerning existing commercial software products, received on a more or less formal basis (call for offers or informal contacts with suppliers). Offer combinations may also be possible.

V The evaluation dimensions may include specifically costs, quality, compliance to international standards, adequacy to the acquisition policy, etc.

Π The scope of the evaluation could be one among the following:
- choose one of the products (unique important acquisition);
- rank the products (repetitive acquisition) in order to combine them with suppliers (acquisition policy);
- absolute evaluation in order to find the "good" products so as to enable further negotiations on them;
- absolute evaluation in order to reject the "bad" products before carrying a deeper analysis;
- synthetic description of existing products.

3.3 Make or buy

This situation arises when a particular software product is required due to the business needs of a company and a decision must be made: should the product be acquired from a choice available in the market or should the product be developed under the control of the company? If the latter path is selected, then should the product be developed internally (assuming that the company possesses the necessary structure, e.g., a Design and Development group under an Information Systems Department) or should it be developed by an external software supplier? In both cases, the software might be developed starting from scratch or (if possible) it could be built based on an existing similar product or development platform. Strategic aspects are the availability of good (reliable, cheap, financially healthy) software suppliers, the existence of a supplier that has already worked for the company (with known quality, cost, respect of delivery schedules, know-how, available manpower), the will to start or continue autonomous software development.

A_{PS} Is a set of actors mainly limited only to elements (people, departments) internal to the organisation interested in the software product. External actors may be consultants or potential suppliers in the case where some buying scenarios have to be simulated.

O_{PS} Are strategic issues of the organisation, power distribution in the organisation.

RS Are knowledge, information, time, power.

A Is a set of possible decisions: {make alone, make outside, buy, deeper analysis, abandon}. It should be noticed that in this case the set of alternatives may be *fragmented*, in the sense that the decision maker may combine the acquisition of off-the-shelf software components with the development of ad-hoc ones.

V The evaluation dimensions represent the various strategic goals of the organisation affected by such decision and the various evolution scenarios of the organisation and its context. Moreover some financial dimension may emerge easily apart from some dimensions concerning timing and vendor rating. Different alternatives are evaluated as different scenarios.

Π The scope of the evaluation may be either a synthetic description of the different scenarios, or a choice among them.

3.4 Tender evaluation

Such a situation occurs mainly after an "external buy" decision is undertaken. Various suppliers are contacted in a formal way and offers are provided by them. We will assume that these offers concern software which has still to be written. They are projects for the creation of the desired software and not existing products.

A_{PS} Among the internal actors may be found the internal client (internal user who needs the software and indicated the requirements list), the evaluation team, the technical and the legal staff. The various tenders have also to be considered.

O_{PS} Among the interests and problems we have to include: the software specifications to be used, the call for tenders and the acquisition policy.

RS The acquisition budget (if any) should be included among the resources.

A Is a set of offers received.

V The evaluation dimensions are similar to the commercial product evaluation problem. However, since we have assumed that the offers concern non existing products (on which it is possible to make tests and measures), more qualitative dimensions may arise. Of course, the evaluation of the tender itself is a very important dimension. Moreover,

the evaluation of the proposed production process and the ability to monitor it should be taken into account.

Π Usually the scope is the selection of one of the tenders. Alternatively, it could be an absolute evaluation either in order to choose the "good" offers for further negotiations, or to reject the "bad" ones before the selection of a tender is decided.

3.5 Software certification

This is typically an activity carried on by a testing laboratory either specialised in specific application software, or of general purpose (i.e., dealing with commercial products).

A_{PS} The external actors include the software producers, national and/or international standard organisations and possibly the demander of the certification (a producer or a consumer or even the specialised press).

O_{PS} The set includes, among others, the different international quality standards, the certification process, the legitimation offered and demanded, the product(s) to be certified.

RS Knowledge, certification authority, time should be included among the resources.

A Is a set of products to be certified.

V The evaluation dimensions constitute an hierarchy of quality dimensions concerning either the software products, or the software production process itself.

Π Usually, the scope is an absolute evaluation with the objective to verify that a product is compliant (and to what extend) to a particular standard. In the case of relative comparison of widely spread commercial products (for instance in the case of tests published by specialised magazines), the scope may be a ranking.

3.6 Software Process Evaluation

Software products are manufactured through software development processes. In general, a software process consists of a software life cycle model and a methodology (a set of rules, techniques, tools) to build software. Various such processes have been proposed and used in practice by software

developers. Moreover, international standards for software processes are already becoming a reality. Evaluation of software processes is a type of problem that occurs frequently. In fact, software process evaluation is itself a typology of evaluation problems:

- a software manufacturer may wish to "design, adopt or adopt and tailor" a process, i.e., ask his research and development team to design their own process, adopt an already existing process as it is or adopt and customise one, respectively.
- a software manufacturer may wish to "keep, improve or change" his process (maintain the existing process, improve it or change it completely)
- a software manufacturer may wish to choose among n different alternative processes, for instance after having opted for the "adopt" option of the first case

A_{PS} In this case, the participating actors are all internal to the manufacturer. External actors may be international standard organisations and consultants that provide assistance for the acquisition of an existing process.

O_{PS} The set includes productivity and product quality of the manufacturer in the context of his(her) strategic issues concerning his(her) business. Moreover, the various existing processes may be considered.

RS Cost, time and various issues related to the ability to manage the transition to the new process.

A Is a set of possible decisions of the type: keep, change, adopt, adopt and tailor) or the different processes to adopt.

V The evaluation dimensions include cost (process set-up cost, investment in hw/sw infrastructure and tools and people training, process application cost), quality of delivered products, product development time, compliance to standards and/or target user requirements. Process learning curves should be taken into account.

Π The scope of the evaluation is the selection among one of the possible software development processes.

3.7 Software System Design Selection

The design phase of a software system development cycle produces a design solution for the problem stated during the user requirements definition phase. Typically, the design phase produces two main results, namely the architectural design and the detailed design. In both cases, often more than one alternatives are present and a decision must be taken.

Architectural solutions are differentiated through implementation of new or modification of existing system layers (user interfaces, processing layers, data layers, interfaces between systems). Such actions include allocation and/or implementation of functionalities in different systems, creation or modification of data flows between systems, data sharing between systems, etc. Infrastructure layers are also considered (data networks, operating systems, protocols, databases, etc.). Different information system architectures may vary in terms of cost (development, operation, maintenance costs), time (time to develop, operate, maintain), quality (e.g., system availability) and standards.

The detailed design phase follows the architectural design and produces a system view that, typically, consists of hierarchies of design units that pass parameters to each other, access stored data and implement the requested functions. Different detailed designs for the same architectural solution may differ in terms of development cost, development time, quality (e.g., coupling and cohesion in structured design) and previously set design standards (that may define various design limitations).

A_{PS} Internal actors are the members of an architecture/design group of a software producer. External actors may be the people responsible for the architecture development of the client's information system and the adherence to design standards.

O_{PS} Two main problems have to be included: the functional and non-functional user requirements to be respected and the design constraints (deriving from standards, from the technical environment, etc.).

RS The available budget for system development should be included among the resources, besides the availability of necessary tools (e.g., CASE tools).

A Is a set of identified design solutions. However, in the case of architectural design, we may emphasise that such a set results from the combination of different options concerning each architectural layer.

We face a problem known as "fragmented alternatives evaluation" (see [Vincke 1992]) as the evaluation of each combination is not simply the sum of the options' values included (e.g., operating system X is considered better than operating system Y, data base management system Z is considered better than data base management system W, but the combination of X and Z may not be the best solution).

V The evaluation dimensions are similar to the commercial products evaluation. However, since the offers are not products (on which it is possible to make tests and measures), more qualitative dimensions may arise.

Π The scope is a choice, since, normally, only one design solution is adopted.

The problem types presented, although covering a large range of possible situations, do not constitute an exhaustive list. Moreover, it is possible to find combined problem situations such as, for instance, combinations of the "keep or change" and the "make or buy" situation or the "software process evaluation" and the "commercial product evaluation" (tools for the support of specific process activities may have to be acquired following the selection of a process). However, it is clear that in all cases a specific problem formulation has to be defined, on the basis of which it will be possible to discuss with the client (decision maker) and reach his(her) consensus (besides the other actors involved).

4 Two real case studies

In the following we present two real case studies dealing with the evaluation, in the first case, of billing systems for a mobile telecommunication operator and, in the second case, of geographical information systems for a large hi-tech company. In both cases we will present the problem formulation and structuring part of the decision aiding process. We will therefore try to present how different problem formulations will provide quite different evaluation models which could lead to different final decision and selection. For such a reason we will simulate different problem formulations and discuss the resulting evaluation models. We may emphasise that such simulations have been partially conducted with the clients of the two studies in order to convince them about the suitability of the problem formulation adopted.

4.1 Selection of a Billing System

A new mobile telecommunication operator has been established in a small, but highly competitive European market. One of the basic operational tools of such companies is their billing system (*BS*). This system allows both a structured accountancy of the traffic and a flexible policy towards the existing and potential clients (enabling for instance a variety of services over the basic ones, the creation of packages of services oriented to specific market targets, the monitoring of each subscriber's traffic).

Some years after the establishment of the company, the necessity to upgrade or to substitute the existing billing system became evident to the management. A decision process has therefore been triggered, and we have been asked to provide decision support. The following problem situation holds:

- The actors involved were:
  - the acquisition (A) manager;
  - the information systems (IS) manager;
  - the marketing and sales (MS) manager;
  - the software suppliers;
  - the IS consultants.

- The objects involved in the process were:
  - the market share of the company;
  - the policy towards the suppliers;
  - the company's internal organisation;
  - the billing system itself.

- The resources implied in the process included the necessary funds for the billing system, the knowledge about billing systems and the relations with the software suppliers. The available time was very short, since all decisions had to be taken in the less possible time due to the extremely competitive environment.

The strategic decision with which the management was faced consisted of choosing one among the following options: upgrade the existing BS, buy and customise an existing BS, buy a BS created ad-hoc for the company by an external supplier (*bespoke system*), develop an ad-hoc BS in collaboration with an external supplier. However, the management was not able to choose an option without analysing what the billing system would be eventually in all such options. We therefore provided three problem formulations (the

fourth option being the upgrade of the existing BS, was considered familiar) which we will call:
- B: buy (and customise an existing BS);
- M: make (externally a new ad-hoc BS);
- D: develop (a new ad-hoc BS in collaboration with a supplier).

In all the three cases, a call for tenders has been provided. The three problem formulations become:

1. $\Gamma_B =< A_B, V_B, \Pi_B >$ where:

   A_B: offers proposed by specific suppliers of existing BS accompanied by a proposal for the customisation phase.

   V_B: points of view of the evaluation:
   - costs (including training, insurance fees and payment conditions);
   - quality (based on ISO9126 and benchmarks on the proposed product);
   - timing (of delivery, test and installation);
   - installed base of the proposed BS (including performance reports on already installed BS of the same type).

   Π_B: ranking of the offers in order to enable further negotiations on the price.

2. $\Gamma_M =< A_M, V_M, \Pi_M >$ where:

   A_M: offers proposed by specific software developers with a different degree of experience in BS development.

   V_M: points of view of the evaluation:
   - costs (including training, insurance fees and payment conditions);
   - requirements satisfaction (client driven requirements);
   - timing (of delivery, test and installation);
   - type of the supplier - developer (taking into account the company's supplying policy);
   - consequences on the company's internal organisation (including project management).

   Π_M: selection of a supplier - developer with whom to establish a supplying process (consisting of benchmarks, tests, training and delivery).

3. $\Gamma_D =< A_D, V_D, \Pi_D >$ where:

   A_D: set of suppliers with whom it could be possible to co-develop a new BS.

   V_D: points of view of the evaluation:
   - costs (distinguished in internal and external costs);
   - requirements analysis and satisfaction;
   - timing (including the time in which the product could be ready for the market);
   - type of the supplier - developer (including company's supplying policy);
   - consequences on the company's internal organisation (including project management);
   - benefits for the company by entering the market of billing systems as a supplier itself.

   Π_D: selection of a co-developer to establish a co-makership policy and therefore a long-term collaboration.

The client finally chose the first problem formulation, implicitly accepting a pure buying policy with respect to the basic strategic choice. We are not going to explain such a choice. We would like to emphasise two observations:

1. From a general point of view, each problem formulation may generate a quite different evaluation model. The set of potential actions is different (existing BS in Γ_B, offers of non existing software in Γ_M, co-developing suppliers in Γ_D). The set of criteria may also be quite different (it is sufficient to notice that the "make" and the "development" option requires to consider as a criterion the implication of the information systems department in the development process, a fact that may alter the distribution of resources and responsibilities in the company's organisation or that the development option requires to evaluate the eventual benefits of "selling" the new billing system). The relative importance of the criteria may also be different, while the aggregation procedures in each model have to be adapted to the different problem statements and the different nature of the criteria.

2. From a software evaluation point of view, the different problem formulations lead to different models as well. In the Γ_B case, existing software products must be compared (even if the one chosen will be customised), a fact that allows the use of existing models (as the ISO9126

standard). Benchmark tests must be also performed. On the other hand, in the Γ_M case, the software artifact does not exist yet. The attention of the evaluation will shift to the requirements satisfaction during the software development, and therefore some quality requirements for the supplier have to be considered a priori. Finally, in the Γ_D case, the evaluation consists of the comparison of possible partners for software development, implying the comparison of the compliance of the partners software development process with the company's standards (assuming that they exist).

Moreover, the priorities among the different criteria and attributes will change from one problem formulation to another, independently of the uncertainty associated to the available or required information. Finally, in order to aggregate the different software measurements, different necessities arise from one problem formulation to another (for instance, in the Γ_B case, measurements may correspond to observations and therefore a functional aggregation can be allowed, while in the Γ_M and in the Γ_D cases, the measurements are predictions or estimations based on expert opinion, a fact that requires a different treatment.

4.2 Selection of a Geographical Information System

A large hi-tech company providing a network based service has decided to buy a geographical information system to store all data and knowledge concerning its network and related services. The acquisition consists of buying a number of licenses (more than 100) and the related workstation equipment. Actually, the desirable software product is a customised version of a GIS system, including data base facilities and other options (some of them mandatory). For this purpose a committee has been established, including the acquisition manager, the information systems manager, a lawyer (due to the importance of the allocated budget) and a delegate of the CEO (due to the strategic importance of the project). The information systems manager has been charged to provide information on the existing market products and, in his turn, he charged his staff to perform an analysis among the existing possibilities. However, it soon appeared that, on the one hand, the number of potential candidate products was incredibly high and that, on the other hand, the customisation needs were such that a deeper investigation was necessary. A call for tenders was arranged, to which a number of suppliers answered. At this point, a selection problem was expressed. Again we will not present in details the evaluation (for such details see

[Paschetta and Tsoukiàs, 2000]), but stress our attention on the decision aiding process.

The problem situation appeared as follows.

1. The actors involved in the process were:
   - the information systems manager;
   - the acquisition manager;
   - the technical staff supporting the IS manager;
   - external consultants;
   - the suppliers.

2. The objects of the process were:
   - the technological expansion of the company;
   - the acquisition budget of the company;
   - the legitimation of the technical staff;
   - the GIS products present in the market;
   - the internal organisation of the company.

3. The resources implied in the process concerned mainly the knowledge about GIS and the authority of the involved managers. The time was relatively relaxed although a decision was expected in a year time.

Two different problem formulations have been considered. In the first case, the problem consists of choosing directly a supplier to procure the product and customise it. In the second case, the problem consists of providing a technical evaluation of each offer. The two problem formulations become:

1. $\Gamma_C =< A_C, V_C, \Pi_C >$ where:

   A_C: offers proposed by specific suppliers of existing GIS products, accompanied by a proposal for the customisation phase.

   V_C: points of view of the evaluation:
   - the cost of each offer;
   - the "quality" of each offer;
   - the satisfaction of internal and external norms by each offer;

   Π_C: selection of a supplier.

2. $\Gamma_S =< A_S, V_S, \Pi_S >$ where:

   A_S: offers proposed by specific suppliers of existing GIS products accompanied by a proposal for the customisation phase.

V_S: points of view of the evaluation:
- different quality dimensions (requirements satisfaction);
- the results of different tests on the proposed software;
- the "quality of the supplier";

Π_S: associate each offer to a profile on an ordinal scale of quality.

The reader may notice that the problem formulation Γ_S can be considered as part of the problem formulation Γ_C since it concerns one of the points of view of the latter, i.e., "quality". Actually, this is what happened; the technical staff worked on the problem formulation Γ_S and the results have been used by the IS manager in the meeting of the evaluation committee where the problem formulation Γ_C was adopted. Nevertheless, what we are interested in, is to highlight the fact that, as far as software evaluation is concerned, the two problem formulations could lead to different evaluation models and more precisely that the point of view on quality in V_C does not necessarily coincide with the result of Γ_S. It was only a specific choice of the evaluators to make them coincide.

In fact the first attempt to model quality was in the V_C frame (the distinction among the two problem formulations was not yet clear). The ISO9126 standard was adopted in order to obtain a ranking of the offers on the six basic criteria (indicated by ISO9126) and then get a final overall ranking. Such an approach was not satisfactory for two reasons:

1. the six basic criteria of the ISO9126 standard were not well adapted to the particular case of the customised GIS offers (for instance the purpose of the GIS to buy was such that some criteria were meaningless, while other important features were neglected);

2. the type of comparisons provided using the ISO9126 results in a ranking which is a relative evaluation of the offers and not an absolute one (in the sense that we obtain a is better than b, but not that a is good and b is bad).

It was only under the problem formulation Γ_S that what the technical staff was looking for appeared in a clear way:

- Make explicit the body of knowledge that the staff had concerning GIS technology, through an hierarchy of attributes, i.e., those dimensions with which they were comfortable and able to provide some measurement (although often just ordinal) with a certain confidence.

- Manage to define a global scale of quality (although ordinal) on which it was possible to give an absolute evaluation of each offer.

Last but not least, the change of perspective in the two problem formulations led to the use of different aggregation techniques. In the Γ_S case, it was necessary to compare each offer to certain external profiles (the values on the ordinal scales) and therefore a sorting procedure was adopted to each level of the hierarchy of attributes. In the Γ_C case, the acquisition manager requested the establishment of trade-offs between cost and quality and therefore a multiattribute value function was proposed (for details about the MCDA methodology, see [Vincke 1992], [Roy 1996]).

5 Discussion

The purpose of this study was to establish whether it could be possible to use the problem typology as a basis for the construction of Intelligent Front Ends (IFEs) in decision support systems used in software evaluation problem situations (see [Vlahavas et al. 1999, Stamelos et al. 2000]). The idea was that such IFEs should be able to provide the user a basis for constructing her/his evaluation model though a guided questioning/answering on a number of features characterising her/his problem.

Such DSSs contain pre-defined evaluation modules so that as soon as the user identifies her/himself in one of the above mentioned problem formulations a number of issues of the evaluation model are already established (such as some of the evaluation dimensions or criteria, a set of aggregation procedures etc.). For example, consider a problem formulation that falls in the Make or Buy type, which involves cost as an evaluation dimension. A DSS module would propose a possible composition of cost into cost for acquisition, customisation, training, operation and maintenance. Further on the module would propose to the user an approach for the difficult task of estimating the cost for customisation. Such an approach might be based on the characterisation and sizing of the software through Function Point Analysis and, subsequently, through the comparison with other similar historical cases for which the cost is already known ([Shepperd and Schofield, 1997]). In this way the user is guided to cope with one important evaluation dimension in a structured and informed way.

However, the experimental validation of such DSSs (see [Vlahavas et al. 1999, Stamelos et al. 2000, Paschetta and Tsoukiàs, 2000]) showed that, although the users appreciate the existence of such pre-defined modules, they rarely use them as such. It seems that they are used more in order to understand

a number of issues (usually of quantitative nature) than as real evaluation models. Users try to define a definitely custom evaluation model.

From such an experience and the two case studies, it is possible to get two empirical results.

1. Trying to apply pre-defined software evaluation models on such general issues as software quality may be a wrong approach. The more general the characteristic of the software to evaluate the more, it is context dependent, in the sense that the problem formulation may produce different elements in the evaluation model. To take just as an example the concept of software quality, we claim that it could be a useless effort to define a unique model of quality. This is not to say that quality standards are useless, but to argue that they should be used as guidelines and as a basis for customisation. It is clear that the problem situation and the problem formulation adopted will provide the specific contents for the quality model.

   Further on, we want to stress that the definition of a quality model has to include objects which are often neglected such as the aggregation procedures. The choice of such procedures is very important, they are part of the model and their choice has to be as accurate as for the rest. In fact quality models are evaluation models and a rigourous definition of all their components is necessary.

2. We claim that an accurate definition of the problem situation may provide the software engineer involved in software evaluation the necessary elements by which (s)he may be able to build useful and reliable evaluation models. In this process an important aspect is the generation of a set of realistic problem formulations on which the consensus of the client has to be reached. Evaluating a software can be a resource consuming activity and therefore must have a clear goal and purpose.

However, an important issue remains open. Is it possible to give a more formal definition of the software evaluation process? At a first glance it seems that the identification of the problem situation and the construction of the problem formulation are activities where more craft is requested than formalisms. This is less true in the construction of the evaluation model since in this case the axioms of measurement theory, preference modelling and aggregation, criteria coherence, alternative independence, etc. are applicable, which provide the basis for a "correct" and "meaningful" model. But what about the first two activities?

Although an exhaustive answer is impossible in the frame of this paper, we will try to outline some elements at least as far as the construction of the problem formulation is concerned, given an identified problem situation. The reader is reminded that a problem formulation Γ is a triplet $\langle A, V, \Pi \rangle$ where A is the set of alternatives, V is the set of points of view and Π is a problem statement (the scope of the evaluation).

- *Construction of* A. Normally, in order to build the set of alternatives, we have to identify among the participating actors $\mathcal{A}_{PS}$ the actors having an active role, besides the power (or the authorisation) to suggest solutions. They will indicate the frame in which it is possible to establish the set of feasible solutions. Going back to the GIS case, it is easy to observe that in order to establish the set A a call for tenders has been realised, procedure which actually specified in this context the above suggestion.

- *Construction of* V. Such a set can be constructed considering the combination of $\mathcal{A}_{PS}$ and $\mathcal{O}_{PS}$. The latter can be viewed as the concerns of the participating actors. Of course a problem formulation cannot represent all concerns of all actors and a choice has to be done. Once the actor(s), whose preferences and values are going to be considered, are chosen, the objects (the concerns) associated to them will provide the basis for the construction of V. Such a set has to represent in fact the points of view of the "decision maker" or "client". However, it may happen that concerns belonging to other actors may participate in the construction of the set V in order to prevent conflicts and facilitate further negotiations in a later stage of the decision process.

- *Construction of* Π. Here all elements of PS are involved. First of all, because the problem statement has to be associated to the actor(s) qualified as decision maker or client. Second, the concerns of such actors, besides helping us in establishing the set V, will help us also in the definition of Π. Third, because the resources allocated by the client to his/her concerns will affect such a definition. More specifically we can observe that the different concerns of the client can be associated to more general (or strategic) concerns up to a unique "problem" which will provide the problem statement requested.

The previous informal description is far from providing a complete procedure for the construction of Γ. However, we believe that it contains the essential elements for such a procedure.

6 Conclusion

In this paper, we presented a (partial) list of software evaluation problem situations identified on an empirical basis, using the concepts of problem situation and problem formulation presented in the IUSWARE approach.

A primary objective of such a description was to highlight the great variety of situations under which a software evaluation may occur and how completely different evaluation models can be defined. In order to provide empirical evidence to our claim, two real case studies are briefly reported, illustrating such differences.

Further, we emphasised the lack of any formal aid in identifying the problem situation and the associated problem formulations. Although it is not possible to provide a complete answer to such a problem, we introduced some elements (as essential components of any procedure that may be established). This is also one of our main research directions for the future. Another interesting research issue are the side effects of evolutive choice sets, i.e., sets of alternatives that change dynamically during the decision making process.

A more general problem concerns the fusion of different knowledge basis for the construction of the problem formulation and the relevant evaluation model. As already mentioned in the paper at least three different sources of knowledge are combined: the intuitive knowledge of the actors concerned, the domain knowledge on the problem situation and the analyst's methodological knowledge. The study of the formal properties of such knowledge and the precise way by which it might be possible to combine them represents a future research challenge.

References

[Basili 1995] Basili V.R., "Applying the GQM paradigm in the experience factory", in N. Fenton, R. Whitty and Y. Iizuka eds., *Software Quality Assurance and Measurement*, Thomson Computer Press, London, 1995, 23-37.

[Beroggi, 1999] Beroggi G., *Decision Modeling in Policy Management. An Introduction to the Analytic Concepts*, Kluwer Academic, Dordrecht, 1999.

[Blin and Tsoukiàs 2001] Blin M-J., Tsoukiàs A., "Multicriteria Methodology Contribution to Software Quality Evaluations", *Software Quality Journal*, vol. 9, 113 - 132, 2001.

[Boloix and Robillard 1995] Boloix G., Robillard N.P., "A Software Evaluation Framework", *IEEE Computer*, vol. 28, (1995) 17-26.

[Bouyssou et al., 2000] Bouyssou D., Marchant Th., Perny P., Pirlot M., Tsoukiàs A., Vincke Ph., *Evaluation and Decision Models: a critical perspective*, Kluwer Academic, Dordrecht, 2000.

[Cardenas-Garcia and Zelkowitz 1991] Cardenas-Garcia S., Zelkowitz V., "A management tool for evaluation of software designs", *IEEE Transactions on Software Engineering*, vol. 17, (1991), 961-971.

[Giakoumakis and Xylomenos 1996] Giakoumakis E.A., Xylomenos G., "Evaluation and selection criteria for software requirements specification standards", *Software Engineering Journal*, (1996), vol. 11, 307-319.

[ISO 9126] ISO/IEC 9126-1, Information Technology - Software quality characteristics and sub-characteristics (1996).

[Kitchenham 1987] Kitchenham B., "Towards a constructive quality model. Part 1: Software quality modeling, measurement and prediction", *Software Engineering Journal*, (1987), 105 - 112.

[Kontio 1996] Kontio, A., "A Case Study in Applying a Systematic Method for COTS Selection", *Proceedings of the IEEE International Conference on Software Engineering*, (1996), 201 - 209.

[LeBlank and Jelassi 1994] Le Blank L., Jelassi T., "An empirical assessment of choice models for software selection: a comparison of the LWA and MAUT techniques", *Revue des systèmes de decision*, vol.3, (1994), 115-126.

[Meskens 1994] Meskens N., "A knowledge-based system for measuring the quality of existing software", *Revue des systèmes de decision*, vol.3, (1994), 201-220.

[Morisio and Tsoukiàs 1997] Morisio M. and Tsoukiàs A., "IusWare, A methodology for the evaluation and selection of software products", *IEE Proceedings on Software Engineering*, vol. 144, (1997), 162-174.

[Mosley 1992] Mosley V., "How to assess tools efficiently and quantitavely", *IEEE-Software*, (1992), vol. 9, 29-32.

[Park and Lim, 1999] Park K., Lim C., "A Structured Methodology for Comparative Evaluation of User Interface Designs Using Usability Criteria and Measures", *International Journal of Industrial Ergonomics*, (1999), vol. 23, 379-389.

[Paschetta and Tsoukiàs, 2000] Paschetta E., Tsoukiàs A., "A Real World MCDA Application: Evaluating Software", *Journal of Multi-Criteria Decision Analysis*, vol. 9, (2000), 205-226.

[Poston and Sexton 1992] Poston R.M., Sexton M.P., "Evaluating and selecting testing tools", *IEEE Software*, (1992), vol. 9, 33-42.

[Roy 1996] Roy B., *Multi-criteria Methodology for Decision Aiding*, Kluwer Academic, Dordrecht, 1996.

[Shepperd and Schofield, 1997] Shepperd M.J., Schofield C., "Estimating Software Project Effort Using Analogies", *IEEE Transactions on Software Engineering*, (1997), vol. 23, 736-743.

[Stamelos et al. 2000] Stamelos I., Vlahavas I., Refanidis I., Tsoukiàs A., Knowledge Based Evaluation of Software Systems: a case study, *n Information and Software Technology*, vol. 42, (2000), 333-345.

[Tsoukiàs, 1997] Tsoukiàs A., "Sur la généralisation des concepts de concordance et discordance en aide multicritère à la décision", Mémoire HDR, Université Paris Dauphine, 1997, appeared also as Document du LAMSADE n. 117.

[Vincke 1992] Vincke Ph., *Multicriteria Decision Aid*, J. Wiley, New York, 1992.

[Vlahavas et al. 1999] Vlahavas I., Stamelos I., Refanidis I., Tsoukiàs A., "ESSE: An Expert System for Software Evaluation", *Knowledge Based Systems*, vol. 12, (1999), 183-197.

[Zahedi 1990] Zahedi F., "A method for quantitative evaluation of expert systems", *European Journal of Operational Research*, vol. 48, (1990), 136-147.