REVISTA COLOMBIANA DE COMPUTACIÓN
Volumen 10, número 2
Págs. 99 - 129

A Classification framework for Software Requirements Prioritization Approaches[1]

Nadina Martinez Carod* Alejandra Cechich*
Fecha de Recibido: 19/04/2006 Fecha de Aprobación: 15/03/2008

Abstract

The task of eliciting requirements has became extremely difficult because stakeholders have different perspectives on an expectation on a system. Besides, the time to obtain the final product is limited. To overcome this situation, a requirements ranking may help in planning releases by indicating which functions are critical and which ones can be added, and in what order, over successive releases. The prioritizing process must hold stakeholder satisfaction considering high-priority requirements first. However, practical experience shows that prioritizing requirements is not as straightforward task as the literature suggests. Considering that, this paper has two goals: the first one is to present a classification framework for software requirements prioritization approaches (emphasizing differences and similarities among eleven selected approaches); and the second one is to show the approaches' weaknesses and to propose possible improvements for future research on this line.

Keywords : *Software Requirements Prioritization, Cognitive Informatics.*

This work is partially supported by UNComa project 04E/059
* Universidad Nacional del Comahue, Departamento de Ciencias de la Computación, Neuquén, Argentina 8300. Email: namartin, acechich@uncoma.edu.ar

1 Introduction

Requirements engineering takes care of activities which attempt to understand the exact needs of the users in a software system and to translate such needs into precise and unambiguous statements, which will be subsequently used in the development of the systems. In most cases, defects of the software are originated in the requirements phase. Once defects are embedded in the requirements, they tend to resist removal. According to Young [36], 85% of the defects of developed software is originated in the requirements. The common and more imp ortant types of requirement errors are incorrect assumptions (49%), omitted requirements (29%) and inconsistent requirements (13%).

As part of Requirements Engineering, "Elicitation" is the phase where an analyst collects information from the stakeholders, clarifies the problems and the needs of the customers and users, tries to find the best solutions, and makes its planning on what software system will be developed. Elicitation is the process of acquiring all relevant knowledge needed to produce a requir ement model of a problem domain. In elicitation, to get well-defined requirements, a consensus among the different stakeholders is needed. There are several elicitation techniques in the literature [29][23][36], however every technique faces the same problem: each stakeholder has different requirements and priorities, which potentially produces conflicting situations. In these cases, stakeholders must negotiate the "right requirements" [11][28] which implies prioritization of software requirements. Nevertheless, often the strategies implemented to solve conflicts among stakeholders are inadequate; for example, weighting requirements can be problematic because sometimes weights are inconsistent and lead to confusion about which are the most essential customer requirements. More sophisticated methods, such as the AHP, and the Cost-Value [30][20], have received some interest in the application of elicitation procedures, and simpler decision -making techniques [5][16], or visualization techniques [15] have been found out to be appropriate to resolve disagreements promoting a cost-effective use. In any case, clearly defining a way of balancing preferences on requirements is essential to the elicitation process.

On the other hand, the requirements elicitation techniques have widely used a family of goal-oriented requirements analysis (GORA) methods [1][4] [10][14][18] as approaches to refine and decompose the needs of customers into more concrete goals that should be achieved. Particularly, a proposal called AGORA [17] extends a version of a Goal-Oriented Requirements Analysis Method by considering detecting and resolving conflicts on goals; the work in [17] considers greater priority when there exists a dependency between requirements, and these interdependencies

can be identified before they are negotiated. More recently, the Goals - Skills -Preferences Framework [13] is used to generate a customizable software design; or techniques from Cognitive Informatics try to find solutions to communication problems during all stages of software engineering [34][35][24] . Some comparisons of elicitation methods have clarified common features. Firstly, the comparative study by Thomas and Oliveros [33] is centralized in properties and limitations of five of the most significant methods for eliciting requirements in goal-oriented requirements engineering. This comparison is organized from the viewpoint of goal acquisition with special emphasis in goal elicitation. Secondly, based on an evaluation framework and influenced by an industrial application, Karlsson [20], characterizes six different methods for prioritizing software requirements. The objective of Karlsson's evaluation is outlining the methods' behaviour for a particular experience, thus the results obtained are not supposed to be generalized by any environment for any application. This evaluation framework is based on inherent characteristics, objective measures and subjective measures.

In this paper, we focus on design and cognitive aspects as main features to characterize different approaches to prioritise requirements, aiming at identifying possible improvements to the processes. Let us briefly clarify the meaning of cognitive aspects in our context. Cognitive informatics (CI) is an interdisciplinary field composed by the intersection of a number of existing disciplines like psychology, philosophy, linguistic and computer science. In [34], Wang define CI as "a branch of information and computer science that studies computing by cognitive methodologies and studies cognitive science by informatics and computing theories". CI can be studied from the artificial intelligence as from the software engineering area. Artificial intelligence studies the mechanisms of natural intelligence and the architecture of the brain [31][3][35] often ignoring psychological aspects of intelligence. On the other hand software engineering is interested in explaining the mechanisms and processes of memory learning and reasoning. Particularly, people have different behavioral characteristics and there are many learning style models to classify people according to those characteristics. In [24], we have proposed analyzing learning and communication aspects of a particular stakeholder by using the Felder-Silverman learning style model. Our argument for doing this is the identification of an analogy between stakeholders and roles in learning style models during elicitation processes, where each stakeholder must learn from others. Therefore, how a particular approach addresses stakeholders' learning and communication features is considered here as the basis for evaluating cognitive aspects.

This paper introduces an extension of the framework presented in [26]. The remainder of the paper is structured as follows. Section 2 describes the conceptual framework to evaluate several proposals. In Section 3 we introduce some approaches of prioritizing requirements, and describe them in terms of the framework. Conclusion and future work are presented in Section 4.

2 A Classification and Comparison Framework

Our Classification framework, depicted in Fig. 1, is structured in two building blocks, design features and cognitive features.

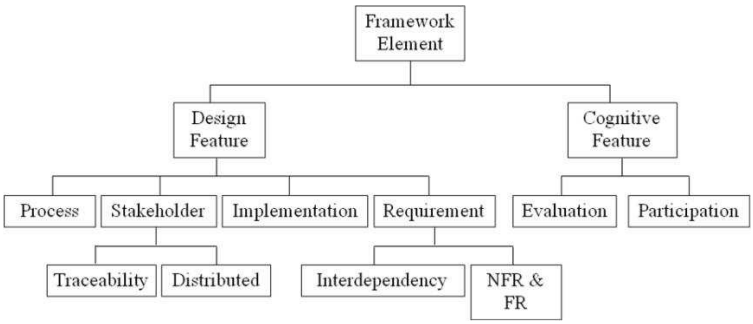

Fig. 1. A conceptual framework for comparison

Our classification framework, depicted in Figure 1, is structured into two building blocks – *design features* and *cognitive features*.

The *design category* is composed of four elements which consider different specifications: *Process, Stakeholders, Implementation* and *Requirements*. The specific features of each prioritization requirement method are categorized by the *Process* element. It considers answering some questions, such as: Does the process detect inconsistency?, Is the process referred to as a systematic or a rigorous process? How we address the problem of dealing with different priorities? Conceptually, is it based in goal decomposition? Does it use a priority or an importance order?

The framework also characterizes how prioritizing methods consider *stakeholders*. There are two parameters to be analyzed: the former refers to the kind of information the method provides with respect to stakeholders. Does the method analyze which stakeholder prioritized a goal, and which priority degree was assigned? The second parameter

considers stakeholders geographically distributed. The *implementation category* depends on the method's scalability and dynamism, i.e. usability. It is influenced by how many and which calculus the method uses, and by the performance of the method with a huge number of requirements. It is considerably important whether the method is supported by tools, as well as a reference to spread projects it was applied. The framework considers information that can demonstrate the method's success in pilot studies.

The *Requirements* element analyzes functional and non functional requirements as well as interactions among requirements – *interdependency* represents requirements interaction. Some methods calculate cost and benefit figures for individual requirements, but if there are significant interactions among requirements the situation becomes more complex. As an example, if two requirements in a method can be achieved by sharing the same solutions to sub-problems, then the cost of attaining both of them may be significantly less than the sum of their individual costs. Therefore, the main key is whether the method can handle requirements' interdependencies. FR & NFR analyses study if the methods are well suited for functional and non functional requirements.

The *cognitive aspects* cover the evaluation of cognitive features as participation and negotiation among stakeholders during the whole process. *Evaluation* studies what personal characteristics serve to establish priorities. *Participation* includes defining how priorities were assigned (subjective or objective) from personal experiences and interviews to ensure the success of the developed method.

To compare those features, we have applied a systematic method to validate and evaluate several proposals: the DESMET method [22]. Particularly, its feature analysis allows the framework to be expressed in terms of a set of common attributes, characteristics or features. To judge the relative order of merit of a specific feature, it is classified in a common judgement scale: *Mandatory (M), Highly Desirable (HD), Desirable (D) and Nice to have (N)*. Then the involved methods have to be judged according to the level of support of a particular feature.

There are two types of features: (1) simple features, that are either present or absent, and are assessed by a simple YES/No nominal scale; and (2) compound features, where the degree of support offered by the method must be measured on an ordinal scale.

A diffe rent score must be assigned to simple and compound features. The following generic judgment scale is used to assess a method for a

particular compound feature: (0) No support – the feature is not supported; (3) Moderate support – the feature is supported in some specific cases; and (5) Strong support – the feature is supported in all cases.

An analysis based on accumulating the absolute scores must assess the relative importance of features. This analysis uses the importance assessment as a weighting fac tor. Although there is no defined rationale for determining appropriate weights, we use the following ones: Mandatory features (10), Highly desirable (6), Desirable (3), and Nice to have (1). Once each method has been scored for each feature of the framework by using a common scale, the results for the methods have to be compared to decide their relative order of merit.

Once each method has been scored for each feature in the framework, using a common scale, the results for the methods have to be compare d to decide their relative order of merit.

2.1. Prioritization Approaches

In this section we briefly introduce some approaches on requirements prioritization presented in the literature.

AGORA is an extended version of the Goal-Oriented Requirements Analysis Method [17]. Goal based techniques begin with highest level goal identifications and derive them into sub - goals. Initial goals can be considered as the needs of the customers. Usually, a goal is decomposed into a number of more specific sub goals. The sub-goals are connected to their parent goals with directed edges. There are two types of decomposition corresponding to the logical combination of the sub-goal. The AND – decomposition, unless all of the sub-goals are achieved, their parent goal cannot be achieved or satisfied. The OR - decomposition, when at least one sub-goal is achieved, its parent goal can be achieved. The decomposition of goals can continue too many different levels of abstraction, creating a goal hierarchy. In many situations a sub -goal may be instrumental to more than one goal. AGORA goal graph is an attributed AND-OR graph where attribute values (contribution values and preference matrices) are added to goal graphs. The contribution of an edge stands for the degree of the contribution of the sub -goal to the achievement of its parent goal, while the preference matrix represents the preference of the goal for each stakeholder. The preference matrix is attached to a goal. The contribution value expresses how many degrees the sub -goal contributes to the achievement of its parent goal, and the higher the value is, more contribution the sub-goal provides. Each stakeholder does not only attach the preference value on his own, but also estimates

the preference values of other stakeholders. As a result, these preferences are represented in the form of a matrix. The stakeholders attach the value subjectively.

The ***Analytical Hierarchy Process*** (AHP) Model was designed by TL Saaty as a decision making aid [30]. It involves building a hierarchy (ranking) of decision elements (candidate requirements) and then making comparisons between each possible pair in a matrix. This weighs each element within a cluster (or level of the hierarchy) and a consistency ratio (useful for checking the consistency of the data). The Analytic Hierarchy Process compares alternatives in a stepwise fashion and measures their contribution to the main objective of the process [20]. The AHP assumes that the problem under investigation can be structured as an attribute hierarchy with at least three levels. In the first level, the overall goal is described. The second level describes the different competing criteria that refine level 1. The third level is used for the selection from competing alternatives. At each level of the hierarchy, a decision maker performs a pair wise comparison of attributes assessing their contribution to each of the higher level nodes to which they are linked. These comparisons involve preference ratios (for actions) or importance ratios (for criteria). The expert decides, for each pair, a number that represents the importance of a term respect to other term of the pair to describe the domain. The overall method consist of: Firstly the candidate requirements are reviewed by the requirements engineers for completeness and to ensure that they are stated in an unambiguous way. Secondly, stakeholders apply the AHP' pair -wise comparison method to assess the relative value of the candidate requirements. Thirdly requirements engineers use AHP' pair-wise comparison to estimate the relative cost of implementing each candidate requirement and use AHP to calculate each candidate requirement's relative value. Finally, stakeholders analyze and discuss the candidate requirements. Based on this discussion, software managers prioritize the requirements and decide which will actually be implemented.

The ***Cost-Value Approach***, designed by Karlsson and Ryan prioritizes requirements according to their relative value and cost [20]. The method is based on an analytical technique and provides a clear indication of the relative costs and values of all candidate requirements. In this approach *Value* is interpreted in relation to a candidate requirement's potential contribution to customer satisfaction with the resulting system. *Cost* is the cost of successfully implementing the candidate requirement. To investigate candidate

requirements, it uses AHP to calculate each candidate requirement's relative value and implementation cost, and plots these on a cost–value diagram. The stakeholders use the cost–value diagram as conceptual maps for analyzing and discussing the candidate requirements. Based on this discussion, software managers prioritize the requirements.

The ***Win-Win*** approach [11] is a negotiation process where people do not get everything they want but they can reasonably assure of getting whatever it is to agree them. It consists in a set of principles, practices, and tools, which enable stakeholders to work out a mutually satisfactory set of shared commitments [2]. In this methodology stakeholders express their goals as win conditions and if everyone concurs, the win conditions become agreements. When stakeholders do not concur, they identify their conflicted win conditions and register their conflicts as issues. In these cases, stakeholders must find out options for mutual gain. Options are iterated and turned into agreements when all stakeholders concur. The stakeholders are in a Win-Win equilibrium condition when the agreements cover all of the win conditions and there are no outstanding issues. The Win-Win approach defines a set of activities guiding stakeholders through a process of gathering, elaborating, prioritizing, and negotiating requirements. The overall method consist of a repeatable requirement negotiation process: First the team builds a clean list of win conditions, then, they categorize them among the stakeholders, and organize them into predefined buckets. Stakeholders have to refine agreements into more measurable requirements.

Quantitative Win-Win is a quantitative evaluation of alternatives of the Win -Win approach to support decision-making [28] that uses an iterative approach. The added value of this approach is its ability to offer quantitative analysis as a backbone for actual decisions. The method consist of three components: Firstly it uses the Analytical Hierarchy Process for a stepwise determination of the stakeholders' preferences in quantitative terms. Secondly, these results are combined with methods for early effort estimation. Thirdly, it reflects the increasing knowledge about the requirements at each iteration cycle. Each one consists of six consecutive steps. In the first step the requirements subset of the original set is defined by a threshold value level assigned by the experts. In the second step, the preferences are computed from the perspective of the overall business value, applying AHP and resulting in normalized vectors of weights. During the next step the same task is done from the perspective of the individual stakeholders. The two previous steps are arranged as a matrix M with step 3 as rows and step 2 as column vector. Then, the original set of requirements is extended by the

added requirements in later stage. At last this method checks feasibility of requirements.

The ***Requirements Interdependencies*** technique (RI) uses conjoint analysis as a tool to determine stakeholder' preferences on an individual item and can be used to detect conflicts among stakeholders [8]. It considers the software project as a product with attributes (functional and non -functional) that define the class of a product. The technique studies the dependencies and correlations between the attributes. Different ways of process implementation can be evaluated varying the values of these attributes. The attributes have to be chosen in order to meet the stakeholders' expectations. Therefore it measures the utility of a given project realization as perceived by the stakeholders. This technique obtains individual utility functions for each stakeholder. Although there are by now various conjoint analysis techniques with different approaches on how to perform the product comparison, the predominant technique in industrial applications is the Adaptive Conjoint Analysis (ACA). ACA [12] allows evaluate product classes with a quite large number of attributes. The overall method consist of: Firstly the stakeholder ranks the level for each attribute individually indicating the importance of it. Next each stakeholder must choose between pairs of products presented adaptively. Based on stakeholders' preferences, the engineers can group people with similar characteristics together and determine a utility function for each person on the subset of attributes relevant to him. For each role, all individual utility functions have to be aggregated into a single utility function, this is done by averaging. It first identifies the most important attributes for each role. In a second step it studies how fixing the preferred attribute levels of a certain role affects the utility for the remaining roles. Although this method has the same philosophy of AHP, it has no explicit methodology to resolve it.

Quality Function Deployment method (QFD) is typically applied to small subsystems [5]. A customer desire is the quality demanded by the customer. A quality characteristic is a measurable attribute by which one can measure whether a customer is getting the demanded quality. A function is something the system must do to ensure the demanded quality. A function is defined in the form <verb, noun>. Quality characteristics and system functions intersect, to define a requirement variable of the form <function, attribute> which is equated to a constant to define a requirement. Requirement variables can be fixed to create requirements or they can be used as design. In QFD, each customer desire is given a value. Quality characteristics are defined through brainstorming to generate an affinity diagram. After forming a tree diagram of the chosen quality characteristics, those at

the lowest level are placed on the axis of a matrix. The customer desires are placed on the other axis. Each quality characteristic is compared with each customer desire to determine the correlation level. The product of the customer desire values and the correlations for a specific quality characteristic provide a value for that quality characteristic. This is interpreted as the value of a quality characteristic for a specific customer desire valuation. The vector of values of customer desire is transformed to a vector of values for quality characteristics using the customer desire/quality characteristic correlation matrix. The same process is used to identify functions, correlate them with customer desires, and transform customer desire values to function values using the customer desire function correlation matrix. Quality characteristics and functions can be ranked in terms of transformed customer value in order to prioritize tasks.

The ***Multi-Criteria Preference Analysis Requirements Negotiation*** (MPARN) is a systematic model to guide stakeholders from options to agreements using multi-criteria preference analysis techniques [16]. The process provides a systematic means of identifying the win conditions of all stakeholders, and the multi-criteria analysis quantifies each stakeholder' view of each option's performance on each criterion considered important. It cooperates with the artifacts of the win-win analysis. Preference analysis can be a useful tool in identifying the value of each alternative's features to individual group members. The MPARN process prioritizes stakeholders´ needs by identifying conflicts and exploring conflict resolutions. The overall process includes elicitation of win conditions, identification of conflicts between stakeholders, and exploring conflict - resolutions options. It also explores objective criteria where the process of identifying preference functions begins with expression of criteria of importance, followed by identification of options under consideration, scoring of these options on the criteria identified, and elicitation of tradeoff relationships. The options are assessed based on the criteria, once this list of criteria has been developed. Each stakeholder assesses each option's performance on each criterion. Scores are assessed for each option on each criterion. Then it obtains relative weights for criteria by each stakeholder. Many methods can be used as direct subjective evaluation, the SMART method [7], the ratio pair-wise comparison method or the geometric progression method. The prior steps will provide sufficient information to identify the preference ranking over the options for each stakeholder of the group, through a value function given earlier, the sum product of weights time scores for each option over all criteria. At last it realizes a post-analysis for agreements.

The ***Visualization*** technique uses visualization tools to requirements conflict identification and resolves problems with exploration of potential solution approaches [15]. The technique represents stakeholder perceptions, measures consensus among the perception, and visualizes the perceptions (support collaborative prioritization of requirements among a group of stakeholders using visualization aids). It proposes Clustering Analysis as a technique to identify stakeholder subgroups having different opinions. The evaluation of each stakeholder is recorded in a value pair of Importance-Difficulty. In order to determine the accepted level of agreements between any two stakeholders, it uses a measure, called "consensus factor", that is a function of the votes of the two stakeholders on all the influencing criteria associated with the issue. It defines thresholds for consensus measures to identify whether a situation is of complete or partial agreement. Visualization aids reveal, at a glance, the positions of all the stakeholders as well as conflict in perceptions. Usually in this visualization, a cluster of dots is used to represent the density of stakeholders having the same or similar perception as represented by their votes. The visualization shows the ranges of votes characterizing the different clusters, and how the current stakeholder has voted with respect to them. The identification of the classified groups helps stakeholders to understand the conflict situation. In a multi-issue, multi-stakeholder situation, the cluster identification provides a way to abstract essential knowledge about the voting patterns from massive detail. It provides structured support for formal elicitation of stakeholder knowledge.

The ***Goals-Skills-Preference*** framework presented in [13] is used to generate a customizable software design. This framework performs requirements analysis on user goals, skills and preferences which need to be identified by requirements engineers and made available to the user at run-time for possible adjustments. The process consists of some components: First, the method used by any elicitation technique to identify goals. The second component identifies the set of required skills, combining the set of required skills and the supported skills. The third component concerns user preferences. The result of this elicitation process is a set of preferences. In the analysis phase, the framework takes requirements as input and generates a set of ranked alternatives for the design phase. An alternative is defined as a set of tasks that together fulfill a set of target goals. In the design phase each alternative corresponds to a group of software components forming a particular architecture. Developers select a set of classes according to user profile, and the software configuration process can be performed by the user at run time.

The ***Psychotherapy for System Requirements*** approach consists of a series of items that can be used to assist the analysts and quality assurance of customer requirements [29][9]. This methodology is transferred from the discipline of psychotherapy to the field of requirements engineering. It can be practice in oral and written requirements. Although this set of rules reduce the risk of getting not well-defined requirements, and that have been implemented successfully in some projects; it only helps the analyst in the elicitation process. It is implemented using natural language in informal notation, and is not been considered as an acquisition technique since it is not supported by any specification language, or by any automated tool.

3 Characterizing Requirement Prioritization Approaches

3.1 Applying the Framework

This characterization has three scenes: the first one emphasizes features presented in all methods and classifies them according the way they are implemented in each method , next it considers simple features; and last it gives compound features details.

3.1.1. Features provided

There are some features that cannot be assessed according to the degree of importance because they are present in all methods. The way in which each method maintains these characteristics are detailed in Table 3. The features are as follows: 1. Method: The process is a method itself, it is composed of other methods or it is part of other methods, 2. *Calculus:* what type and number of calculations it makes, 3. *Negotiation Process*: how the method provides artefacts to support negotiation.

Identifying *stakeholders* – individuals or organizations who stand to gain or lose from the success or failure of a system – is a critical task. Stakeholders include customers or clients (who pay for the system), developers (who design, construct and maintain the system), and users (who interact with the system to get their work done). Users themselves are not homogeneous, and part of the elicitation process is to identify the needs of different user classes, such as novice users, expert users, occasional users, disabled users, and so on. In case of diverging opinions between stakeholders, they must work out to develop acceptable solutions for all people involved. One of traditional pitfall

during an elicitation process is that people interpret things in the light of their own background assumptions, uncertainty generates useless information. Besides, often developers cannot implement all requirements because of time and resource constraints. Instead, they focus on implementing firstly the requirements that are set as the most important. This fact implies a negotiation process to balance conflicting requirements. From this point of view, negotiation may be seen as the key of a successful software projects development. This is the reason why one classification feature is the way the method solves the incompatibilities between stakeholders' priorities. The stakeholders can probably agree on requirement priorities informally. Larger or more contentious projects need a more structured approach, which removes some of the emotion, politics, and guesswork from prioritization. All participants must agree on what they are saying when they approve the requirements, and they must understand the costs of making changes in the future. The ***negotiation process*** also includes renegotiate commitments when new requirements are accepted.

Important Features	Method	Calculus	Negotiation Process
AGORA	Although it is a complete method, it suggests the combination of methods as the Double Benefit, AHP and analysis of costs in order to prioritize the	It calculates variance matrices	It does not have an explicit methodology to make the negotiation between stakeholders
AHP	It is a method itself, and it is used as part of other methods	For N requirements, it computes a matrix comparison (N x N), a generation of the extreme vector of this matrix, a matrix of priority and an extreme vector, and a matrix of requirements multiplied by the relative vector of requirements. The process becomes tedious when there are more than 15 elements.	The stakeholders reach an agreement in the relative values of each pair of requirements
Cost-Value	It uses the AHP method twice, one for the cost and another one for values	Multiplies the calculations of the AHP method by 2	The stakeholders meet and agree on the relative values of each pair of requirements
Win-Win	Is a methodology itself	It does not make specific calculations	It considers the importance of the process as the facility of implementation for requirements prioritization

Important Features	Method	Calculus	Negotiation Process
Quantitative Win-Win	At the first stage it uses the AHP method, to obtain the relative values of the requirements	The calculations of the AHP method, the calculation of the effort and the calculations of the vector of importance of the classes of requirements in each iteration	It considers stakeholders with different perspectives from the business, it supposes that the preferences of the participants vary with the time
Requirements Interdependence	It uses existing methodologies of the Decision Making Process.	The method uses the ACA technique (Adaptive Conjoint Analysis), which implies a great number of calculations. Each participant must choose between pairs of products, which would add too many calculations.	It groups stakeholders with similar characteristics. The stakeholders choose independently the presented products
QFD	It is a method itself	It does not use too many calculations, although the majority are from the matrix of requirements	The stakeholders meet in sub-groups and define the requirements
MPARN	It is a systematic model, which cooperates with the artifacts of the win- win analysis	The calculations are according to the methods considered to assess scores (direct assessment, linear function, nonlinear function or geometrically progressing scale method) and the methods considered to assess relative criteria weights by stakeholder (direct evaluation, SMART, ratio pair-wise comparison or geometric progression method).	By using multi-criteria preference analysis techniques, which supplement the win-win process of systematically evaluating and negotiating conflict - resolution options by eliciting stakeholder preferences and criteria, as well as assessing how well each of the generated options performs on the set of criteria.
Visualization Issue	It is a method which applies visualization techniques both to requirements conflict identification and resolution problems	The calculations are only for the visualization tool, new calculations to determine priorities do not exist	In small projects all the stakeholders are negotiating together, in spread projects the stakeholders are grouping in order to negotiate
GSP	It is a methodological framework which involves goals, skills and preferences. It is a process that consist of three subprocesses	It makes a few numbers of calculations. The calculations are for determining the precedence of the alternatives, the method adds the relative values of the alternatives to calculate them	The negotiation process is by group meeting.
Psych. SR	It is not a method. It does not have a defined methodology to prioritize the requirements	It does not make calculations of any type	The stakeholders meet and define the requirements. The analyst uses this technique to help in obtaining the requirements This technique is only applied to specifications in natural language

Table 1. Characterization in terms of important features

3.1.2. Simple features

The simple features we considered to analyze processes are: Consistency: Specifies whether the process detect inconsistencies,

2.Rigorous: The process (method) is systematic or rigorous, 3.Goal decomposition: The process is based on goal decomposition, 4. Priority: Priority goals with precedence, 5. Requirements Interdependence: The process identifies dependences among requirements, 6. Objective: How the priorities are assigned: subjectively or objectively.

In Table 1 we can observe there is not a complete, simple, fast and reliable prioritizing approach. Neither of them provides specific tools to solve conflicts. Some approaches as Goals - Skill and Preferences (GSP) and AGORA are based on goals, others such as the Win -Win, Quantitative Win-Win and Visualization Issue technique, on negotiation processes, and some others such as QDF and MPARN are based on industrial and decision-making techniques. On the other hand, AHP and Cost-Value are based on pair wise comparison, and the Psychotherapy for System Requirements method is based on human interaction using natural language and is the only method that cannot establish priorities between requirements. AGORA, as the methods based on negotiation processes, can detect such inconsistencies. In these methods, we can see both win conditions and candidate requirements as initial goals. Considering this aspect, only the GSP and AGORA approaches allow decomposition from needs of the customers into sub-goals. Although both AHP as Quantitative Win -Win are reliable, they require a large number of mathematical calculations to prioritize few requirements. Only the Psychotherapy from System Requirements takes cognitive aspects into account allowing people specify what they really mean. However, it is not a formal or systematic method.

Generally speaking, the approaches use cognitive aspects only during the negotiation phase, where the analyst must reach mutual consensus. For example, the Cost-Value approach applies the AHP method to assess each candidate requirement relative value as implementation cost. In the case of the Quantitative Win -Win, the overall method consists of three main components: The first one is the use of the AHP method, the second is the separation of the required importance level of candidate requirements, and thirdly the selection for requirements subset under given resource constraints.

Table 2 summarizes the previous discussion by characterizing the methods in terms of the framework's simple features we have introduced in section 2.

	Consistency (HD)	Rigorous/ Systematic (HD)	Goal decomposition (D)	Priority (M)	Require-ments Dependence (D)	Objective (D)
AGORA	By attaching Attribute values as preference matrices.	Rigorous process	It uses AND-decomposition and OR-decomposition	Priorities are based on conflicting goals	Only in goal decomposition	Attribute values are attached subjectively. But techniques as AHP can be used to obtain more objective values
AHP	By redundancy of pair-wise comparison	Systematic and rigorous method	No	Compares requirements in three hierarchical levels	No	Objective because it represents each term respect to other term.
Cost-Value	By redundancy of pair-wise comparison	Systematic and rigorous method	No	Idem as AHP	No	Idem as AHP
Win-Win	By analyzing the priorities with the Conflict Consultant tool.	Not rigorous or systematic	No	Detects priorities between the requirements	No	Objective because it must reach consensus among the stakeholders
Quantitative Win-Win	Between pairs of requirements (AHP process), eliminating some of them and checking the resulting set.	Systematic process	No	Detects priorities between the requirements	No	It is more objective than Win-Win because it adds a quantitative analysis
Requirements Interdependence	Although it Detects inconsistencies, it does not have an explicit methodology to correct them.	Not rigorous or systematic	No	Requirement precedence can be given	The process is based on requirements interdependence	It is subjective
QFD	It does not detect inconsistencies.	Not rigorous	No	Precedence can be given because it is based on assigning a numeric value to each requirement	No	Priorities are given subjectively
MPARN	It does not detect inconsistencies	Systematically negotiated agreements using the multi-criteria preference analysis techniques	No	It considers a precedence for each option through the preference function	No	Although the priorities occur in subjective form they return unfailingly objective
Visualization Issue	It does not detect inconsistencies	Nor systematic or rigorous	No	It considers a precedence that can be shared by one or several requirements	No	Priorities are given subjectively
GSP	It does not detect inconsistencies	Nor systematic or rigorous	Each goal is a node in a goal graph, and is decomposed in OR/AND relationships into sub goals	It considers a precedence when evaluating the alternatives	No	It is subjective. The first part of the process (identification of objectives) can be made by using any elicitation technique

	Consistency (HD)	Rigorous/ Systematic (HD)	Goal decompo-sition (D)	Priority (M)	Require - ments Dependen-ce (D)	Objective (D)
Psych . SR	Although it detects divergence between the stakeholders, it does not detect inconsistencies.	Nor rigorous or systematic	No	No	No	It is subjective

Information gathered during requirements elicitation often has to be interpreted, analyzed, modeled and validated before a complete and enough set of requirements of a system is collected. An important first step in improving the elicitation process is to recognize the *real* requirements.

Many times two stakeholders agree on requirements with opposite meanings, which turns impossible the implementation of those requirements. These are called ***requirements inconsistencies***. Inconsistencies arise as a result of conflicts between requirements. Each inconsistency implies that some action is needed, to identify the cause and seek a resolution. We consider this action ***highly desirable*** because if a process can detect inconsistencies, then it probably achieves more reliable results. A concept related to the inconsistency is the ambiguity. Ambiguity means that a requirement statement can have several different meanings and the stakeholders cannot be certain of which is correct. A more insidious form of ambiguity results when multiple stakeholders interpret a requirement in different ways. Each stakeholder concludes that his or her interpretation is correct, and the ambiguity remains undetected until later.

If a method is ***rigorous*** and ***systematic*** it ensures the success of the process, providing robust and comprehensive steps and handling requirements consistently and effectively, which became this feature ***highly desirable***. It aids in the validity and verification and it is related intimately to the consistency of requirements.

There are a number of inherent difficulties in the process of identifying stakeholders and their needs. Stakeholders may be numerous and distributed. Their goals may vary and conflict, depending on t heir perspectives of the environment in which they work and the tasks they wish to accomplish. Stakeholders' goals may not be explicit or may be difficult to articulate, and, inevitably, satisfaction of these goals may be constrained by a variety of factors outside their control. *Goals* denote the objectives a system must meet. Eliciting high level goals early in the development process is crucial. However, goal-oriented requirements

elicitation is an activity that continues as development proceeds, as high level goals are refined into lower level goals. Therefore, the process based on ***goal decomposition*** is ***desirable*** in a prioritization process. Eliciting goals focuses the requirements engineer on the problem domain and the needs of the stakeholders, rather than on possible solutions to those problems.

All requirements are not top priority. Requirements specialists and developers must be trained not to make assumptions, not to make requirements decisions, and not to add features and capabilities to systems when they are not part of the real requirements. Discussion of requirements priorities improves communication between the customer and the developer and helps to resolve conflicts. We can see *Priority* as a key attribute of each requirement (that should be included in requirements database). It is not a direct measurement, at least not if it is assigned in an objective way. It will never be a single value; instead, a feature's priority is a combination of two or more measurements that interact and influence each other. The relative priority is an important attribute of each functional requirement. Various stakeholders might interpret high priority differently, leading to mismatched expectations about what functionality will be included in next releases. One difficult is that users are reluctant to prioritize requirements because they fear the developers will automatically restrict the project to the highest priority items and the others will never be implemented. It is important to compare the priority of each proposed requirement change against the body of requirements remaining to be implemented. Requirements prioritization plays a key role in the requirements engineering process, in particular with respect to critical tasks such as software release planning. Therefore, we consider ***mandatory*** the process of deriving an order relation on a given set of requirements, in order to assign a ***priority order***, with the ultimate goal of obtaining a shared rationale for partitioning them into subsequent product releases.

The different occurrences of requirements changes throughout the life cycle points out some dependencies among functional requirements. Understanding these dependencies may improve the requirements process. The dependencies among software functions can be evaluated by the number of common fault reports arose during the software life cycle. One hypothesis implies that if two functions are modified due to the same fault report, then there are some ***requirements interdependencies*** between them. Thus an analysis of such identified fault reports is ***desirable*** as it may give important information about requirements (it prioritizes methods looking at the connections and interrelationships among different requirements).

The process of negotiation involves both prioritizing requirements, and selecting the adapted set of requirements to be satisfied. One disadvantage detected is that in many methods only one stakeholder has the responsibility of estimating the relative requirements value, which became the process subjective. We suppose as ***desirable*** that the process considers the search of solutions to be as ***objective*** as can be possible.

The simple feature table (Table 3) has the following scores (considering a YES feature with score of 5 and a NO simple feature with a score of 0):

Simple Features	Consistency	Rigorous/ Systematic	Goal Decomposition	Priority	Requirements Dependence	Objective
Importance	*HD*	*HD*	*D*	*M*	*D*	*D*
AGORA	5	5	5	5	5	0
AHP	5	5	0	5	0	5
Cost-Value	5	5	0	5	0	5
Win-Win	5	0	0	5	0	5
Quantitative Win-Win	5	5	0	5	0	5
Requirements Interdependence	5	0	0	5	5	0
QFD	0	0	0	5	0	0
MPARN	0	5	0	5	0	5
Visualization Tssu	0	0	0	5	0	0
GSP	0	0	5	5	0	0
Psych. SR	0	0	0	0	0	0

Table 3. Score of simple features

3.1.3. Compound features

A specific set of compound features can be: 1.*Traceability*: Captures which stakeholder prioritizes some aspects and why, 2. *Distributed stakeholders:* Weather the stakeholders can work in a collaborative environment, 3. *Computational tools*: Supported by computational tools, 4. *Experience*: The method has been validated in a case study involving real requirements and it has been practiced in large scale development, 5. *Cognitive aspects*: Weather the requirements prioritization process can be adjusted based on stakeholders' profiles using cognitive aspects of stakeholders , 6. *Human experience* : Experience needed for implementation and the minimum number of interviews needed for successful results, 7. *NFR* : All methods consider functional requirements but only some of them are developed for both Functional Requirements and Non Functional Requirements (FR & NFR).

	Traceability (M)	Distributed Stakehol-ders (HD)	Tools (D)	Experience (D)	Cognitive aspects (HD)	Human experience (N)	NFR (D)
AGORA	It allows to maintain information of objectives prioritized by each stakeholder, using the preference matrix, but not why	No	It is still not supported by computation a tools	It has not been used in spread projects. The example proposed is a user accounting system on the Web	None	Although it requires little experience, also requires many interviews	It considers only functiona l require-ments
AHP	The process involves almost all the stakeholders, so it does not maintain information of whom considered each priority or why.	No	An extensive bibliography of reference and several computation a tools has been generated	It is applied by main companies and world-wide institutions	None	Although it does not need much experience, it needs several interviews to coordinate the relative values between the stakeholders	Although it is usually used for funct ional require-ments, it could also be used for non-functional ones.
Cost-Value	It does not maintain information of whom considered each priority or why	No	The second phase of the method is supported by program written in language C	It was used in several industrial projects	None	Interviews are necessary to coordinate the relative values between the stakeholders and to review the results of the cost-value diagrams	It is adapted for both types of require-ments
Win-Win	It is possible to know which participants prioritized certain objectives, but not why	Yes, it is designed to be able to be used in collaborative virtual environments	Supported by four generations tools: Win-Win, 2G W-W, 3G W-W, Easy W-W	It was used in industrial projects, with COTS products.	None	Although many interviews are needed, it does not require too much experience	It is adapted for both types of requirements
Quantitative Win-Win	It is possible to obtain which participants prioritized certain objectives, but not why	No, this method is fed up on the co-participation of the stakeholders to consider new require-ments	Some specific tools not widely used such as [6][27]. Boehm also created a prototype for his Win-Win spiral model	It was used in spread projects. It is widely used in industry, independen tly from the domain	None	Although it does not require too much experience, it requires too many interviews	It can be adapted to both types of require-ments
Requirement Interdepende nce	It does not maintain information of who assigned each priority or why	Yes, since stakeholders choose products indepen-dently	Parts of the method are supported by tools, nevertheless i does not exist a general software that fully support this methodology	It was used in spread projects, usually in industry	It consider s the political status of the stake-holders	It needs experience to make the process successful	It can be adapted to both types of require-ements

	Traceability (M)	Distributed Stakeholders (HD)	Tools (D)	Experience (D)	Cognitive aspects (HD)	Human experience (N)	NFR (D)
QFD	It does not maintain any type of information from the stakeholders	The geometric nature of the process allows working better with isolated groups	This technique is partially supported by tools.	It has been applied successfully from 1991 in the industry of health	It considers the political status of the stakeholders	It needs experience to make the process successful.	It can be adapted to both types of requirements
MPARN	Yes, as in the Win-Win method, it is possible to obtain which participants prioritized certain objectives, but not why. Preference analysis can be a useful tool	No	The MPARN offers supports for generation and negotiation planning, for criteria exploration and assessment of scores and criteria	It does not mention any spread project	None	Similar to the Win-Win method. It does not require too much experience	It can be adapted to both types of requirements
Visualization Issue	Although the different priorities assigned from each requirement are known, it is no possible to know who assigns each priority or why	Yes, authors are even working to improve this item	Currently working on the elaboration of supporting tools, inspired by (DCPT)	It has not been used in real-world projects for case studies	None	Although it does not need much experience, it needs several interviews to negotiate priorities	It is thought for functional requirements
GSP	No. As the criteria of all the participants are joined together, it doe not register who prioritized each requirement	No	There is no tool yet. It is an on-going project.	It is applied to a case study involving traumatic brain injury patients	Yes, but it does not use it as a weight to mediate.	It needs much experience and many interviews to determine for each user, goals, skills And preferenc	It is developed only for functional require-ments
Psych. SR	No. As the criteria of all the participants are joined together, it does not register who prioritized each requirement	No.	It does not make calcula-tions of any type. It is not supported by tools	It is used in many small projects, but it is not used in great projects.	It does not consider cognitive characteri stics of any of the partici-pants	It does not need much experience, which is obtained in two or three days of training	It can be adapted to both types of require-ments

Table 4. Characterization in terms of compound features (Cont.)

For each compound feature the values ***0,3,5*** have the following meanings;

- *Traceability*: "0" indicates that it is not possible to determine which stakeholder (or what group of stakeholders) prioritized each aspect; "3" indicates that it is possible to determine who prioritized some requirements, but the reason cannot be determined; and "5" is used to score the methods that keep the reason why each participant prioritized requirements.
- *Distributed stakeholders:* "0" indicates that the methods do not support collaborative environments; "3" indicates the methods are supported by distributed groups (i.e., Visualization Issue and QFD); and "5" indicates the method can operate with stakeholders in a collaborative environment (i.e., Win -Win, and Requirements Interdependence).
- *Computational tools*: "0" indicates methods with no computational support (i.e., Psych. P.R.); "3" indicates both – only some processes of the method are supported by computational tools or the computational tools are partially implemented; and "5" indicates the method is completely supported by computational tools.
- *Experience*: "0" means the method has not been empirically validated; "3" indicates small experiences/ projects with real requirements; and "5" indicates the method has been used in spread projects,
- *Cognitive aspects: "0"* means the method does not consider cognitive characteristics in any aspect; *"*3" indicates methods which consider cognitive aspects but they do not use them in order to average weights (i.e., GSP); and *"*5" indicates methods where the weights of stakeholders' perceptions can be adjusted based on stakeholder profiles (i.e., QFD).
- *Human experience: "* 0" is assigned to the methods that require much experience and a great number of interviews (or too long processes); "3" is assigned to processes that although do not require much experience, they require a great number of interviews; and "5" is for processes that do not require previous experience nor several interviews (i.e., only Psych. P.R.)
- *Non functional requirements:* "0" is for the methods that cannot be used for nonfunctional requirements (i.e., AGORA, Visualization Issues, and GSP); "3" is for methods that can use non functional requirements; and *"*5" is assigned to methods thought for both types of requirements.

In Table 5, we judge the degree of support of the compound features on an ordinal scale (0: no support; 3: moderate support and 5: strong support).

mpound Features *Importance*	Traceabilit y M	Distributed Stakeholder s HD	Computation Tools D	Experience D	Cognitive aspects HD	Human experience N	NFR D
AGORA	3	0	3	3	0	3	0
AHP	0	0	5	5	0	3	3
Cost-Value	0	0	3	5	0	0	5
Wim-Win	3	5	5	5	0	3	5
Quantitative Win - Win	3	0	3	5	0	3	3
Requirements Interdependence	0	5	3	5	0	0	5
QFD	0	3	3	5	5	0	3
MPARN	3	0	5	3	0	3	3
Visualization Issue	3	3	5	3	0	3	0
GSP	0	0	3	0	3	0	0
Psych. SR	0	0	0	3	0	5	3

Table 5. Score of compound features

Requirements traceability is defined as the ability to describe and follow the life of a requirement, in both a forward and backward direction (from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases). Traceability includes providing requirements management and maintaining information of which stakeholders prioritize requirements and why, in order to facilitate requirements verification. Besides, sometimes developers or project managers agree to make suggested changes without thinking carefully through the implications. The change might turn out to be more complex than anticipated, take longer than promised, be technically or economically infeasible, or conflict with other requirements. Every change in requirements will consume resources. Anyway, managing changing requirements is a process of recognizing change through continued requirements elicitation, reevaluation of risk, and evaluation of systems in their operational environment. ***Traceability*** involves providing techniques and tools for controlling the impact of changes in different parts of the project. Typical changes in requirements specifications include adding or deleting requirements. The process for dealing with requirements changes, as the environments that support this process, is considered ***mandatory*** because it helps to scope the possible impact of changes.

Often, users claim to be too busy to spend the time it takes to iteratively gather and refine the requirements. Although researchers have noted the importance of communication among stakeholders, they continue studying distributed requirements elicitation. Some approaches may

help to minimize the impact of these problems. One of them, the CSCW (Computer-Supported Cooperative Work), is the area that takes into account human behavior as well as the technical support that people may need to work as a group in a more productive way. GroupWare is the software used for communication and collaboration in workgroups. Many organizations have adopted a decentralized, team-based, distributed structure, whose members communicate and coordinate their work through information technology. Groupware tools now permit powerful means of communication, allowing groups to develop distributed software engineering activities. As the groups are several and heterogeneous, the process which support ***distributed stakeholders*** are ***highly desirable*** since it is common that participants involved in a software development project must elicit requirements in a scene where stakeholders are geographically distributed. Thus, the managing of the distance between members of a development group is an important issue added to the requirements elicitation process.

Is our intention that the prioritization process be supported by ***computational tools***; the argument for that is that sometimes this feature avoids paralyzing the process (or making the process not too heavy). If requirements development seems to go on forever, you might be a victim of analysis paralysis. In the context of software development, computer lays a particularly important role. Theoretical computer science provides the framework to assess the feasibility of requirements, while practical computer science provides the tools by which software solutions are developed. Since software is a formal description, analysis of its behavior is amenable to formal reasoning. A system will change the activities that it supports.

The process must be proved in real projects (experience). At least this is a ***desirable*** item since many processes are good theoretically but they are practically impossible to be implemented.

In a scene where stakeholders are geographically distributed, considering the characteristics of interpersonal communication and the virtual area where it is carried out, the importance of applying interdisciplinary approaches, such as Cognitive Engineering, is currently increasing.

The cognitive aspects cover the evaluation of cognitive features as participation in negotiation among stakeholders during the whole process and not the cognitive techniques for knowledge acquisition of knowledge based system. The “evaluation” studies what personal characteristics serve to establish priorities weights. The participation of the groups includes defining how priorities occurred (subjective or objective) as the necessary personal experience and interviews to ensure the success of the developed method. There three concepts

related with Cognitive Engineering: *cognitive psychology, cognitive techniques and cognitive informatics*. The first concept, Cognitive psychology, provides an understanding of the difficulties people may have in describing their needs. Cognitive techniques include a series of techniques originally developed for knowledge acquisition in knowledge-based systems. Such techniques include protocol analysis (in which an expert thinks aloud while performing a task, to provide the observer with insights into the cognitive processes used to perform the task), laddering (using probes to elicit structure and content of stakeholder knowledge), card sorting (asking stakeholders to sort cards in groups, each of which has name of some domain entity), repertory grids (constructing an attribute matrix for entities, by asking stakeholders for attributes applicable to entities and values for cells in each entity). The last concept, cognitive informatics is an approach to face the problems of a requirements elicitation process. It is a a profound interdisciplinary research area that tackles the common root problems of modern informatics, computation, software engineering, artificial intelligence (AI), neural psychology, and cognitive science. One of the most interesting things found in cognitive informatics is that it embodies many science and engineering disciplines, such as informatics, computing, software engineering, and cognitive sciences, sharing a common root problem – how the natural intelligence processes information. Therefore, we consider ***highly desirable*** the evaluation of ***cognitive aspects*** to establish priorities weights.

One of the most important goals of elicitation is to find out what problem needs to be solved, and hence identify system boundaries . These boundaries define, at a high level, where the final delivered system will fit into the current operational environment. Identifying and agreeing a system's boundaries affects all subsequent elicitation efforts. The identification of stakeholders and user classes, of goals and tasks, and of scenarios and use cases, all depend on how the boundaries are chosen and on the human experience of developers and analysts. Requirements specialists are generally more familiar than other development staff with recent technology advances and also can help eliciting real customer needs and expectations based on the stated requirements. Sometimes the developers have not enough experience, (or the ones which have enough experience are too expensive for specific projects). Anyway it would be interesting (***nice***) to have methods or processes which imply not experts, or less ***human experience***.

The software requirements specification serves as a container for both the functional requirements and the nonfunctional requirements. The latter include quality attribute goals, performance objectives, business rules,

design and implementation constraints, and external interface requirements. It is ***desirable*** that the quality attributes, ***-non functional requirements***-(such as usability, efficiency, portability, and maintainability) can be elicited from users during the prioritization process.

3.2 Framework Analysis

In *Table 5* we can deduce that at least three characteristics considered fundamental (traceability, distributed stakeholders and cognitive aspects) are not supported (or are little supported) by the prioritization methods.

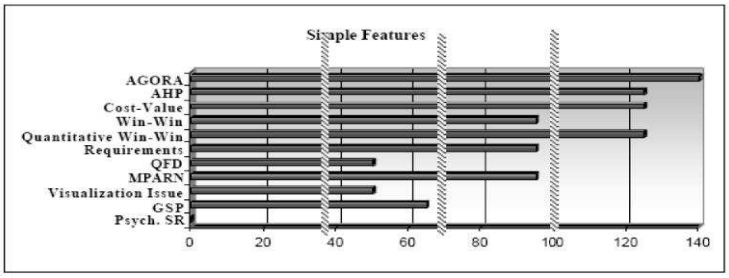

Fig. 2 Sum results of simple features

Then, each feature is assessed by calculating the product between its assessed value and the specific weight depending of its importance. Figure 2 shows graphically the comparative representation of the sum results by the classification methods, with respect to *Table 5* of simple features.

To simple features, the maximum value that can be assigned to a method would have the value of 155, obtained by considering the assigned weights of each feature product as 5 *(6+6+3+10+3+3) *5*. The simple features from *Table 1* and *Table 2*, can be analyzed from two viewpoints: the first one considering the most significant characteristics and the second according to the sum of their relative weights.

By analyzing the information above, we realize that the method to be discharged is Psych. S.R., because it does not have any of the mentioned characteristics. In addition, three levels in this classification could be defined. In the first level would be AGORA, since it supports ***M*** and ***HD*** features, the methods AHP, Cost-Value, Quantitative Win-Win would be in this level too, since they have ***M*** feature and some ***HD*** features. In the following level they would be the methods Win - Win, Requirements Interdependency and MPARN; and in the third level would be GSP, Visual Issue and QFD (they do not have characteristics ***HD***).

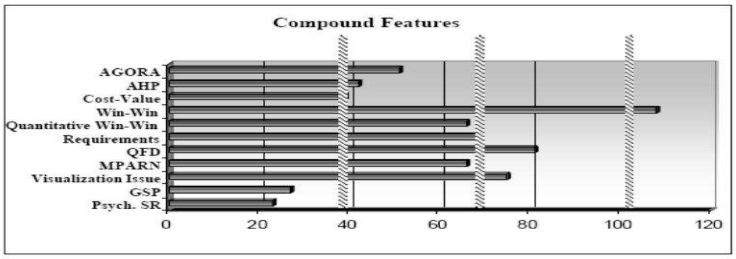

Fig. 3. Sum results of compound features

In the case of the compound features, the first analysis is more difficult to make, since we analyze aggregated features. Therefore we analyze only the sum of the relative weights values. In general with respect to these features, all the classification methods have values that defer much to the optimal ones. If we consider the sum of their relative weights values and we define ranks for each level we might agree on the following classification: Level 1 (160 - 100); Level 2 (99 – 66); Level 3 (65 - 38); Level 4 (37 – 0). With this ranking in mind, we find in the higher level the Win - Win method, then the following level includes the Quantitative Win-Win, Requirements Interdependence, QFD, MPARN, and Visualization Issue methods . Finally, at the third level we find AGORA, AHP and Cost-Value and, at the last level the methods GSP and Psych.SR, with very low values.

The sum of the values for all the methods, combining *Table3* and *Table5,* are shown in *Table 6*.

Method	Simple Features	Compound Features	Sum
Win-Win	95	108	203
AGORA	140	51	191
Quantitative Win-Win	125	66	191
AHP	125	42	167
Cost-Value	125	39	164
Requirements Interdependence	95	69	164
MPARN	95	66	161
QFD	50	81	131
Visualization Issue	50	75	125
GSP	65	27	92
Psych. SR	0	23	23

Table 6. Sum values

With all these graphs and tables in mind we can make a final comparison. Hence, Figure 4 graphically shows percentages obtained

by all the methods in relation to the maximum possible value, which is 315 and represents the 100% in a graphical representation.

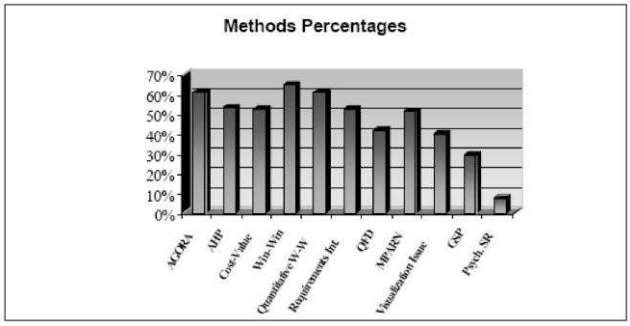

Fig. 4. Methods percentages

Fig. 5 shows the most relevant features of *Table 3* and *Table 5 (HD)* and *(M)*. As a conclusion, there is a lack of treatment for the following features: *cognitive aspect*s (14.55%), *traceability* (27.27%) and *distributed stakeholders* (29.09%).

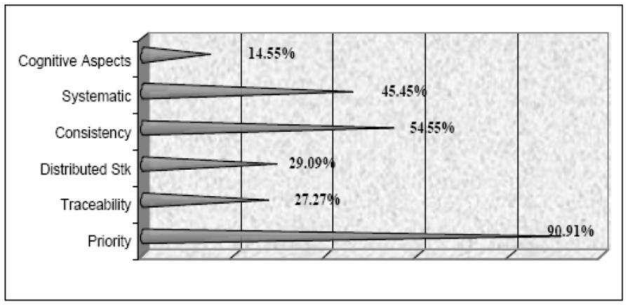

Fig. 5. Most relevant features

4 Conclusion and Future work

In this paper, we focused on cognitive aspects not for knowledge acquisition, but as participation of stakeholders as main features to be analyzed on the different approaches to prioritize requirements. The framework we proposed classifies methods for prioritizing software requirements. It was used to provide an overview of differences and similarities among different approaches. We conclude that requirements prioritization methods, in most cases, do not have cognitive aspects, they are distributed stakeholders partially supported and they cannot determine who nor how, prioritized some requirement.

The assignment of cognitive weights to each stakeholder may help assess a candidate group to be involved in prioritizing a set of requirements. Therefore, the requirements prioritization process can be adjusted based on stakeholders' profiles using cognitive aspects of stakeholders. We suggest improving communication and reduce misunderstandings based on Cognitive Psychology. Thus, we continue working on the cognitive area. Our future work will define weighting preference processes and selection preference processes, which will be supported by automatic tools for requirements prioritization.

References

[1] Antón A. *"Goal Based Requirements Analysis"* In Proceedings of the 2nd International Conference on Requirements Engineering (ICRE '96) IEEE software April 15 - 18, 1996

[2] Boehm B.W., Grünbacher P., Briggs B. *"Developing Groupware for Requirements Negotiation: Lessons Learned"*. IEEE Software, May/June 2001, pp. 46-55

[3] Chiew V. and Wang Y. *"From Cognit ive Psychology to Cognitive Informatics"*. In Proceedings of the Second IEEE International Conference on Cognitive Informatics (ICCI'03) London, UK, August 2003, pp 114-120.

[4] Dardenne A., van Lamsweerde A., and Fickas S, 1993. *"Goal - directed Requirements Acquisition"*. Science of Computer Programming Vol. 20, pp. 3 -50.

[5] Dean, Edwin. *"Quality Function Deployment for Large Systems."*, International Engineering Management Conference '92, Eatontown NJ USA, October 25 -28, 199.

[6] Eberlein. "Requirements Acquisition and Specification for Telecommunication Services" , PhD Thesis. University of Wales, Swansea, UK, 1997.

[7] Edwards, W. and Barron, F.H., *"SMARTS and SMARTER: Improved Simple Methods for Multiattribute Utility Measurement"*, Organizational Behavior and Human Decision Processes 60, 1994, pp. 306-325.

[8] Giesen J., Völker A., *"Requirements Interdependencies and Stakeholders Preferences"*, IEEE Joint International Conference on Requirements Engineering (RE'02). September 2002. pp 206 -212

[9] Goetz R. and Rupp C. *"Psychotherapy for System Requirements"*. Proceedings of Second IEEE International Conference on Cognitive Informatics (ICCI' 03).

[10] GRL homepage, http://www.cs.toronto.edu/k-m/GRL/

[11] Grüenbacher P. "Collaborative Requirements Negotiation with EasyWinWin" 2nd International Workshop on the Requirements Engineering Process, Greenwich, London IEEE Computer Society,2000. ISBN 0-7695-0680-1. pp. 954-690.

[12] http://www.sawtooth.com

[13] Hui B., Lisakos S., and Mylopoulos J.. *"Requirements Analysis for Customizable Software: A Goals-Skills -Preferences Framework"* . In Proceedings of the 11th IEEE International Requirements Engineering Conference, p p 117– 126, 2003

[14] I* homepage, http://www.cs.toronto.edu/km/istar

[15] In H. and Roy, S., *"Visualization Issues for Software Requirements Negotiation"* , IEEE International Computer Software and Applications Conference (COMPSAC 2001), Chicago, Illinois, USA, pp. 10 -15, October 2001.

[16] In H., Olson D., Rodgers T. *"A Requirements Negotiation Model Based on Multi - Criteria Analysis."* Fifth IEEE International Symposium on Requirements Engineering (RE '01). August 27-31, 2001. Toronto, Canada. pp 312.

[17] Kaiya H., Horai H., and Saeki M., *"AGORA: Attributed Goal - Oriented Requirements Analysis Method"* , In Proceedings of the IEEE International Conference on Requirements Engineering, 2002, pp. 13-22.

[18] KAOS homepage, http://www.info.ucl.ac.be/research /projects/AVL/ReqEng.html

[19] Karlsson, J. "Software Requirements Prioritizing". ICREE p. 110, Proceedings of the 2nd International Conference on Requirements Engineering ICRE, April, 1996.

[20] Karlsson, J. and Ryan, K. *"A Cost -Value Approach for Prioritizing Requirements"*. IEEE Software, Vol. 14(5): p. 67-74, September/October 1997.

[21] Kimura D. homepage. http:// www.dhushara.com/book/ socio/kimura/kimura.h tm.

[22] Kitchenham B. DESMET : *"A method for evaluating Software Engineering methods. and tools"*. Technical Report TR96 -09, ISSN:1353-7776, 1996.

[23] Leoucopoulos P. and Karakostas V. *"System Requirements Engineering"*, Mc Graw-Hill, 1995

[24] Martín A., Martínez C., Martínez Carod N., Aranda G., and Cechich A. *"Classifying Groupware Tools to Improve Communication in Geographically Distributed Elicitation"*. IX Congreso Argentino en Ciencias de la Computación, CACIC 2003, La Plata, 6 -10 Octubre 2003, pp. 942-953.

[25] Martinez Carod N. and Cechic A. "Applying Learning Style Models To Prioritize Conflicting Goals". (WICC 2004)- May'04.

[26] Martinez Carod, N. and Cechich, A. *"Classifying Software Requirement Prioritization Approaches"*. XI Congreso Argentino en Ciencias de la Computación, CACIC 2005, Entre Ríos, 6 -10 Octubre 2005.

[27] Reubenstein H.B. and Waters R.C.: *"The Requirements Apprentice: Automated Assistance for Requirements Acquisition"*, IEEE Transactions on Software Engineering.,

[28] Ruhe G., Eberlein A, and Pfahl D. "Quantitative WinWin - A Quantitative Method for Decision Support in Requirements Negotiation" Fraunhofer IESE, Germany, 2002, ISERN-02-05.

[29] Rupp C.. *"Requirements and Psychology"*. IEEE (Software May/June) pp.16-18

[30] Saaty T.L., 1990. *"The Analytic Hierarchy Process"* . McGraw-Hill.

[31] Shi Z., Shi J. *"Perspectives On Cognitive Informatics"*. In Proceedings of the Second IEEE International Conference on Cognitive Informatics. (ICCI´03), p p . 129-137, 2003.

[32] Soloman B and Felder R. homepage, http://www.engr.ncsu.edu/learningstyles/ ilsweb.html

[33] Thomas P., Oliveros A. *"Elicitación de Objetivos, un estudio comparativo"*. IX Congreso Argentino en Ciencias de la Computación, CACIC 2003, La Plata, 6-10 Octubre 2003, p p. 990-1002.

[34] Wang Y. "Cognitive Informatics: A New Transdisciplinary Reseearch Field". (2003)

[35] Wang Y. "On Cognitive Informatics". In Proceedings of the First IEEE InternationalConference on Cognitive Informatics. (ICCI´02), Calgary, Alberta, Canada, August 2002, pp 34-42

[36] Young R.. *"Recommended Requirements Gathering Practices"*. CroossTalk The Journal of Defense Software Engineering. April 2002. pp. 9 -12