An Hybrid Approach to QoS Evaluation

Danilo Ardagna[1], Marco Comerio[2] Flavio De Paoli[2], and Simone Grega[2]

[1] Politecnico di Milano
[2] Universitá degli Studi Milano Bicocca

Abstract. Usually, the process of development of services available as web applications considers only functional requirements. Since, an ever-growing number of users take advantage of different kinds of communication channels and devices, this process must be revised by considering new aspects: quality of service (QoS), user profiles and technical characteristics of channels. In previous works, we proposed a methodology that provides a rational to formalize the redesign process of existing services to support multi-channel access. This paper extends our approach and highlights how the QoS dimensions can be considered quantitatively in the different phases of the methodology. Moreover, an hybrid approach that allows the QoS evaluation, during the development of a service, is proposed.

1 Introduction

Usually, a service available as a web application is characterized by the functionalities that it provides to the final users. Therefore, during the development of these services, only functional requirements are considered. Actually, the possibility to deliver multi-channel applications underlines the necessity to characterize each service with a well-defined set of qualities of service (QoS). In fact, even if a service fulfills all of its functional requirements by providing the required features, it can still be unacceptable if, for example, availability is too little, performance is too poor, or usability does not meet end-user expectations. Therefore, traditional development processes need to be rethought to take into account non-functional requirements of different nature (technological, social, organizational, etc.) at the right stage of the development. In the MAIS (Multichannel Adaptive Information Systems) project [6], we are defining a design methodology [7] that addresses quality aspects in multi-channel contexts. Our methodology, described in depth in the second section, underlines the phases of the development process in which these quality requirements must be considered. Our approach associates some QoS (for example, security, usability and adaptability) to the analyzed service and permits the developers to evaluate the feasibility of delivering these qualities to the final users. The evaluation considers the technical characteristics of the available devices (for example, screen size and resolution, audio power and available memory), the context in which the service is used and aspects related to user profile (UP).

The definition of a quality model will form a proper foundation for identifying, analyzing, and specifying the large number of quality requirements. This quality model provides a tool that can be used to turn these general high-level quality requirements into detailed measurable descriptions. Such model is based on an ontology of qualities that helps in classifying and evaluating a quality with a precise description/definition and with its relations with other qualities. Relations could be of different nature: a quality could be a composition of other simpler qualities, a quality could be a refinement of another, two qualities could be independent of each other and finally relations could reflect different perspectives (provider, users, mediator, . . .). Such relationships, extracted from the ontology, can be modeled by a QoS tree in which the root represents the analyzed quality, the children nodes are the composing qualities and the leaves are the composing technical characteristics.

The ontology that is under construction in the MAIS project specifies for each quality the following attributes: independency, observability, controllability, negotiability and measurability. The independency attribute states whether the quality is primitive or composed. In the latter case the composing qualities must be explicit by the composition rules needed to calculate the analyzed quality. Such rules include linear composition, which requires the definition of weights associated with composing qualities, and non-linear composition, which requires the definition of functions or explicit tables. The others attributes state whether the quality is observable, controllable or negotiable. A quality is observable when the user may only measure its value. Instead, if the user may also express a preference the quality is defined as controllable. Finally, a quality is negotiable when it's possible to establish a process of negotiation between the user and the platform. Moreover, the attribute measurability states how the quality is measured (metric, method of measure, max and minimum value).

It's necessary to underline that, except independency and measurability, the others attributes may have different values according to different factors. A first factor deals with the quality of the service delivered by a provider with respect to the quality perceived by the client. A second factor deals with the context (business or technological) in which a quality is considered. A third factor deals with the service domain: a quality which is relevant in a certain domain could be irrelevant, or even not measurable, in others. For this reason, the ontology has to be instantiated according to application domain, the context and the prospective of analysis.

The remainder of the paper discusses an hybrid approach to evaluate the value of a QoS. Section 2 will present an overview of the reference methodology. Sections 3 and 4 will detail the QoS evaluation using a running example. Conclusions are drawn in Section 5.

2 The Reference Methodology

In the MAIS project, we are developing a methodology for design/redesign services that addresses quality aspects in multi-channel contexts. In [3] and [4], we proposed a methodology for the process of re-design of existing services to adapt themselves to multi-channel contexts, i.e., to user profiles and technology environments. The methodology is based on existing specifications, and information about communication channels and available technologies. In these works we verified the efficiency of our methodology redesigning services for the information system of the Italian National Data Base of Bovine Registry (BDN). The need of keeping services available to a wide variety of final users (for example, keeper and veterinarian) makes the BDN an ideal case study to develop multichannel services. These services, available as a set of browser-based applications, need to be redesigned to adapt themselves to other channels as PDA, mobile phones and multi-frequency telephones. Actually, we want to review/enrich our methodology considering user quality requirements (UQR) and the QoS model previously presented. We want to describe how these UQR must be considered during the different phases of the development of a service. Moreover, the revised methodology considers two possible approaches: service design and service redesign. The difference among the two is that, in the former case, a comprehensive requirement solicitation and definition is needed to identify the functional aspects, while in the latter case, service redesign roots in existing functionalities that are typically available via browser.

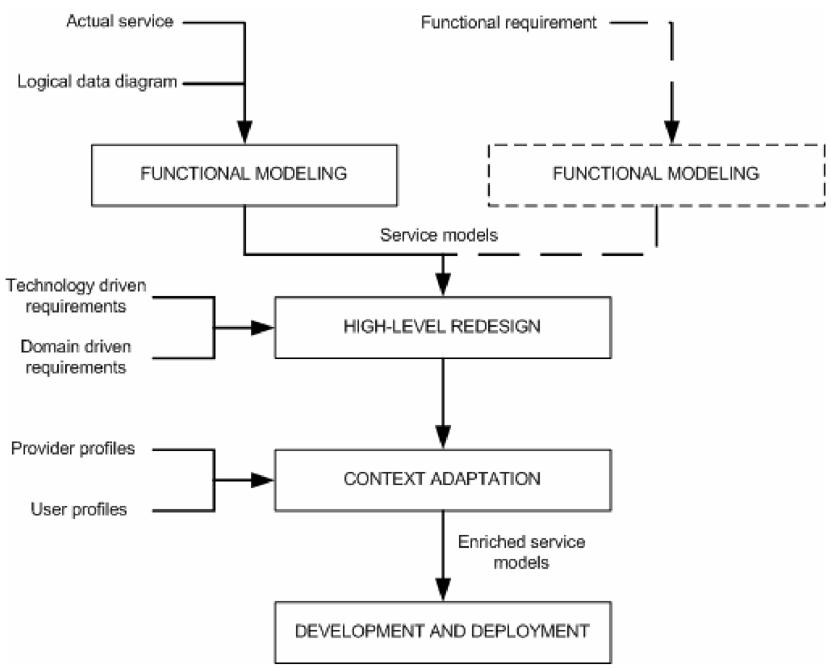

Fig. 1. The phases of the methodology

Figure 1 shows the phases of the methodology with inputs and outputs for each phase. "Functional service modeling" aims to deliver a complete set of UML diagrams that highlights the logical and operational structure of the service. This phase can start from an already existing service, hence concerning service redesign, or from functional requirements, hence concerning new service design.

The main objective of the second phase, "High-level redesign", is to redesign the service architecture in the light of new requirements promoted by the new channels and by domain characteristics. Special consideration is given to behavior modeling to address the interaction between the user and the service according to functional requirements. Moreover, in this phase quality requirements related to user and domain needs must be considered. Therefore, it is necessary to locate the quality dimension related to user quality requirements that are described in an high-level language. The ontology and the QoS model previously described are used to locate, define and classify the QoS in the right mode. The located quality is quantified and modeled using an extension of standard UML [uml:omg], proposed by OMG, that permits to model QoS, constraints and the relationships between them. The modified UML diagrams, that are the output of this phase, define the architecture of the service considering only abstract requirements. In other terms, no specific technologies or user characteristics are addressed.

Instead, the "Context adaptation" phase takes into account the actual deployment environment to evaluate and adapt the abstract assumptions with respect to actual technical characteristics of channels and user profiles. In particular, the quality assumptions perform in the high-level redesign phase must be evaluated. Therefore QoS trees, that show the analyzed qualities and the relationships among other qualities, are extracted. These trees are selected using the ontology and the QoS model. Moreover, it's necessary to consider for each class of domain users the aspects of the user profile (for example, experience and preference) that are related to the analyzed qualities. These characteristics and the previously extracted QoS trees are used to define the quality level requests by the different classes of final users. The hybrid approach, proposed in the following sections, is used to perform this task. This approach permits the quantification of a quality (root of the tree) using a bottom-up approach and linear/non-linear compositions. After this task, a comparison between the level of quality defined in the high-level redesign phase (service quality request) and the levels request by the different classes of users (user quality request) must be performed. The comparison permits, using information related to the analyzed context, the definition of the level of quality that the service must provide. The evaluation of the compatibility between this value and the available technology is made using the hybrid approach and the previously extracted QoS trees. If the evaluation has a negative result, it's necessary to resolve the discovered incompatibility. Examples of this task are proposed in the following sections. Instead, if the evaluation has a positive result, the context adaptation phase is completed and the output of the methodology is a set of UML diagrams that models the multi-channel service along with its quality characteristics. Such a model will be exploited to actually implement and deploy the service to make it available to clients.

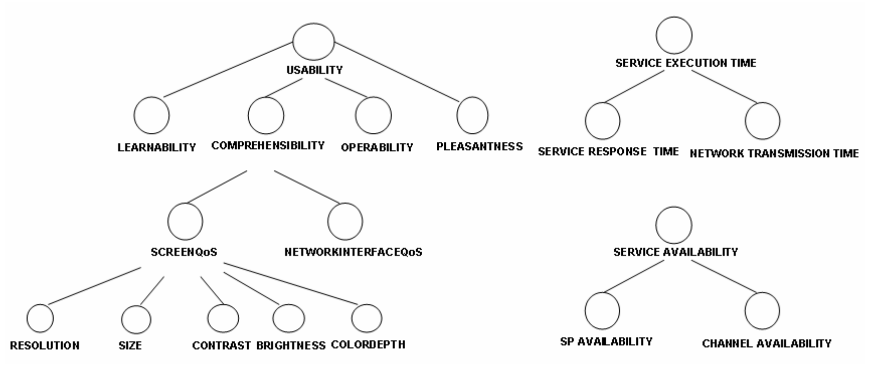

Fig. 2. Quality Trees

3 Quality of Service Evaluation

The QoS evaluation technique will be presented by a running example. Let us assume that the qualities of interest for the end user are: the usability, the service execution time and the service availability. Then, in the second phase of the methodology the QoS trees reported in Figure 2 are extracted from the ontology. A tree node n represents a quality dimension, while edges between a node n and its children indicate a dependency between quality dimension n and the quality dimension corresponding to its children. The dependency could be expressed by an explicit expression, for example the service execution time (i.e. the expected delay between the time instant when a request is sent and the time when the result is obtained) is given by the sum of the service response time (i.e., the time required to process the request by the Service Provider (SP) infrastructure) and the network transmission time (the time required to transmit request and response). In the same way, the service availability can be expressed by the product of the SP availability and the channel availability (the channel in the MAIS project includes the network, the network interface, the application protocol and the end-user device. In a mobile environment the channel availability could be lower than the network availability). We do not introduce any assumption on the properties of functional dependencies. Dependency could be linear, non linear or could even be expressed in tabular form if the explicit formulation is unknown. This latter situation happens very frequently in practice. Among the 321 quality dimensions classified in the MAIS project [2], 203 dependencies have been identified and there exists an explicit formulation only in 8% of total cases, while the dependency can be evaluated by running simulations (and hence expressed in tabular form) in almost 50% of total cases.

The dependency among quality dimensions could also be qualitative. In the example reported in Figure 2 usability depends on learnability, comprehensibility operability and pleasantness. Furthermore, comprehensibility depends on *ScreenQoS* and *NetworkInterfaceQoS*. *ScreenQoS* depends on the quality attributes of the device screen (resolution, size, etc).

In order to evaluate quantitatively the value of usability, the Simple Additive Weighting (SAW) technique is adopted as proposed in [5, 9]. The SAW method is one of the most widely used techniques to obtain a *score* from a set of dimensions having different units of measure.

First note that the leaves of the tree correspond to technical characteristic t_c of the device. Each device can be associated with a tuple $< t_1, t_2, \ldots, t_c, \ldots, t_C >$ where each value t_c assumes the value of the corresponding technical characteristic. Considering the range of values proposed for Resolution, Size and Color depth of available screens, examples of possible tuples are the following:

$T_1 =< 800x600, 5.0", \ldots, 32bit >$; …; $T_n =< 1024x640, 19.0", \ldots, 64bit >$

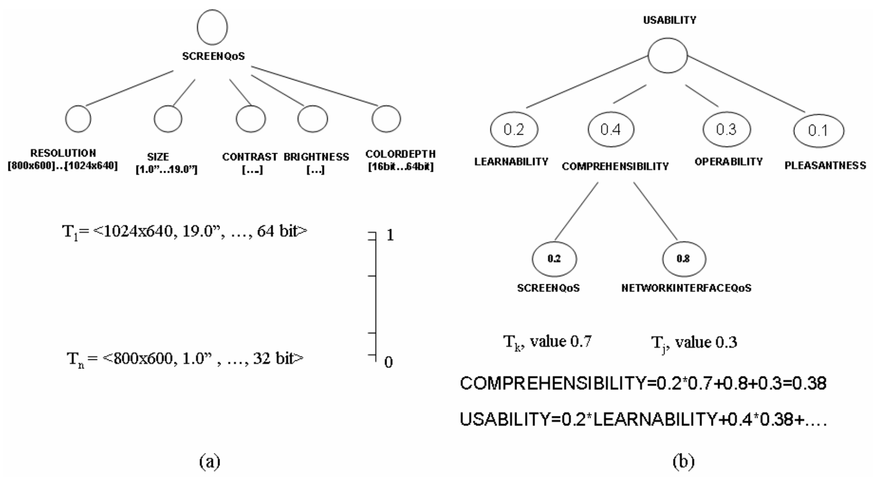
SCREENQoS
RESOLUTION [800x600]..[1024x640]
SIZE [1.0"...19.0"]
CONTRAST [...]
BRIGHTNESS [...]
COLORDEPTH [16bit...64bit]
T1= <1024x640, 19.0", …, 64 bit>
Tn = <800x600, 1.0" , …, 32 bit>
1
0
(a)
USABILITY
0.2
LEARNABILITY
0.4
COMPREHENSIBILITY
0.3
OPERABILITY
0.1
PLEASANTNESS
0.2
SCREENQoS
0.8
NETWORKINTERFACEQoS
Tk, value 0.7
Tj, value 0.3
COMPREHENSIBILITY=0.2*0.7+0.8+0.3=0.38
USABILITY=0.2*LEARNABILITY+0.4*0.38+....
(b)

Fig. 3. Simple Additive Weighting Technique

The domain expert associates each tuple with a value in $[0, 1]$, i.e. defines a function $v(T)$. This value represents the level of quality of the parent node when the corresponding tuple is selected. Note that, as shown in Figure 3a the assignment could be non uniform in the interval. In the proposed example, T_1 and T_n determine different ScreenQoS values. Note that, the mapping is domain dependent and could depend also on the user profile. In this latter case, the mapping can be modeled as a function $v(T, U)$ which associates the quality value v to a tuple T and the user profile UP.

Furthermore, each node of the tree is associated with a weight and the value or score of a parent node is calculated by multiplying the child's weight by its value and then summing across all children. Weights are normalized, i.e. their sums equals to 1, hence the quality value of every node ranges in $[0, 1]$. Figure 3b shows how to evaluate the value of comprehensibility when the device selected for the end user corresponds to screen T_k and networkInterface T_j. The process can be iterated for other dimensions (learnability, operability, etc.) and can be applied bottom up in order to obtain the value for the tree root, i.e., the service usability in the example.

4 Assumptions Evaluation

The quality evaluation technique allows verifying design hypotheses. If the technical characteristics are fixed, then the relevant quality values can be determined. A design hypothesis is verified if the technical characteristics provide quality values greater or equal to given threshold B_k fixed at design time (let us consider for simplicity positive quality dimension, i.e. attributes such that the higher the value the higher the quality for the end user). For example the service design is accepted if the technology characteristics guarantees 0.99% of availability and a usability greater than 0.7. Quality thresholds can be fixed a priori as a desired characteristic of the service but could also be determined by end user profiles.

In the MAIS framework a user profile is a set of characteristics of the users that can be exploited for further customization of services. The study and the determination of the user profiles require a preliminarily in-depth analysis of habits, preferences, behaviors, which is out of the scope of this paper. Profiles define service requirements for individual and group of users. In order to improve the system adaptability and usability of the provided services, specific peculiarities of each user should be highlighted. An example of personalization is given by the analysis of Activity Participations and Body Functions of each user as presented in [12]. For example, if the user profile reveals a poor education in a specific field, the system should be able to supply a simplified interaction mode to access a service in that field; this could be obtained by avoiding expert terminology and using exemplifying figures.

The quality evaluation of an end user profile starts with the creation of the UP/QoS Matrix that defines the dependencies between the QoS previously described and the UP characteristics. An example of QoS/UP Matrix is reported in Table 1.
QoS dimensions (Comprehensibility, Learnability, Operability, Pleasantness) are the columns of the matrix, user profile dimensions are ($Is\_ltf\_pref.$, Is_ltf_skills, $ICF\_relational\_capabilities$, $ICF\_body\_function$, $Expertise$, $Deliverypre-ferences$) the rows of the matrix. The "X" highlights the dependency between a QoS and a UP dimension. In the proposed example, the Operability dimension is related to Is_ltf_skills (ability to perform a particular operation), ICF_rela-

QoS UP	**Comprehensibility**	**Learnability**	**Operability**	**Pleasantness**
Is_ltf_pref.	**X**			**X**
Is_ltf_skills		**X**	**X**	
ICF_rel_capab.			**X**	
ICF_body_funct.	**X**	**X**	**X**	
Expertise	**X**	**X**	**X**	
Delivery_pref.	**X**	**X**	**X**	**X**

Table 1. Qos/UP Matrix

tional_capabilities (capacity to interact with the system), *ICF_body_function* (physical and psychological condition of the user), *Expertise* and *Delivery* preferences.

In a second phase (see Table 2), a weight is associated with each identified dependency with a procedure similar to the SAW technique discussed for QoS trees. Weights are numeric values that represent the level of influence between QoS and UP dimensions. Numeric values are domain dependent and therefore assigned by domain experts. The sum of weights of each columns has to be equal to 1.

QoS UP	**Comprehensibility**	**Learnability**	**Operability**	**Pleasantness**
Is_ltf_pref.	**0.1**			**0.5**
Is_ltf_skills		**0.2**	**0.2**	
ICF_rel_capab.			**0.2**	
ICF_body_funct.	**0.4**	**0.5**	**0.2**	
Expertise	**0.3**	**0.2**	**0.1**	
Delivery_pref.	**0.2**	**0.1**	**0.3**	**0.5**

Table 2. Qos/UP Matrix Evaluation

If for example we assume that the value of *Is_ltf_pref*, *ICF_body_funct*, *Expertise* and *Delivery_pref* are equal to 0.2, 1, 0.8 and 0.8 then the value of comprehensibility required by the user profile is 0.82. If we consider the example above shown in Figure 3, then the set of technical characteristics selected to deploy the service does not satisfy the user profile requirements and the design hypotheses have to be revised.

The assumptions revision proceeds by identifying the most violated constraints. A different set of technical characteristic should be identified in order to improve the quality measure which corresponds to the most violated constraint and which does not introduce further constraints violation. In our framework this phase is supported in a semi-automatic manner as it will be discussed in Section 4.1.

Our quantitative approach allows also evaluating the lower bound to be provided by each leaf in order to satisfy assumptions. In the example above by considering the weight assignment 0.2, 0.8 for ScreenQoS and NetworkInterface-QoS, one can determine that in order to satisfy UP requirements by modifying only the ScreenQoS attribute, then ScreenQoS has to be set equal to 2.9 (this can be derived by the relation $0.2 * ScreenQoS + 0.8 * 0.3 >= 0.82$), which is impossible since the value attributed to every node can be at most 1. Vice versa, if the NetworkInterfaceQoS is set equal to 0.85, then the UP constraints is satisfied and hence the design hypothesis is verified.

If, vice versa, design hypotheses are verified, then in the same way we can determine for each leaf the range such that constraints are satisfied.

In the MAIS framework we are implementing a semi-automatic tool which support the designer in the assumption revision which will be presented in the next Section.

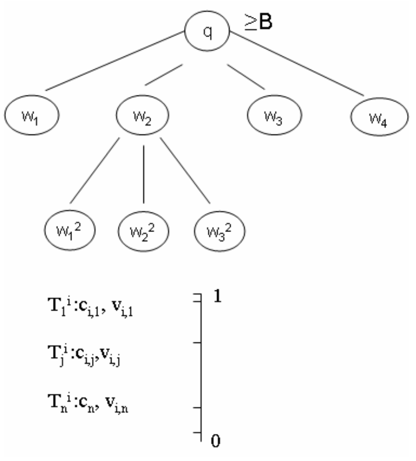

Fig. 4. Hypothesis Revision as an Optimization Problem

4.1 Revising Design Assumptions

The service design/re-design can be modeled as an optimization problem which can be formulated as: identify the set of choices for the technical characteristic relevant for the end users and for the service requirements which minimizes design costs.

Let us consider the quality tree reported in Figure 4 and let us assume first that the quality tree extracted includes only qualitative dependencies, hence the quality value for the root attribute can be determined by a linear expression. Let us indicate with w_k the weights associated with the quality attributes of the first level of the tree, while w_l^k are the weights of the second level associated with node k. Let us assume that the overall value of the quality tree depends on a single set of technical choices I. Let us indicate the tuples for technical choice i as $T_1^i, T_2^i, \ldots, T_j^i \ldots, T_n^i$. Every alternative j for the technical choice i can be associated with:

- $v_{i,j}$: the quality value, assigned by the domain expert, for alternative j;
- $c_{i,j}$: the cost associated with alternative j.

For example, if the technical choice i is the end user device the cost $c_{i,j}$ is the cost of provisioning of a given client device (which is proportional to the number of end users). If the technical choice i is the network bandwidth, $c_{i,j}$ is the cost of the network connection. Let us indicate with $x_{i,j}$ the binary decision variable of our model. $x_{i,j}$ is equal to 1 if the j alternative for the technical choice i is selected and 0 otherwise. The optimization problem can then formulated as:

P1)
$$\min \sum_{i\in I} \sum_{j=1}^{n} c_{i,j} x_{i,j}$$
$$\sum_{i\in I} x_{i,j} = 1; \qquad \forall i \tag{1}$$
$$\sum_k w_k \sum_l w_l^k v_{i,j} x_{i,j} \geq B \tag{2}$$
$$x_{i,j} \in \{0,1\}$$

where the constraints family 1 guarantees that exactly one alternative for each technical choice is selected, while equation 2 guarantees that the quality value provided by the solution is greater than the requirement B, hence the design hypothesis is verified.

The problem above is a NP-hard linear integer programming problem [8]. If constraint 1 is relaxed, then problem P1) is a knapsack problem. The classical 0-1 Knapsack Problem (KP) is to pick up items from a knapsack for maximum total value, so that the total resource does not exceed the resource constraint W of the knapsack. Let there be M items with values $c_1, c_2, \ldots, c_M$ and the corresponding resources required $w_1, w_2, \ldots, w_M$. Mathematically KP can be formalized as:

$$\max \sum_{m=1}^{M} c_m y_m$$
$$\sum_{m=1}^{M} w_m y_m \leq W$$
$$y_m \in \{0,1\}$$

setting $x_{i,j} = 1 - y_{i*n+j}$ and considering that for every programming problem with objective function $F(x)$, the solution of the problem $\min F(x)$ is also the solution of the problem $-\max F(x)$ then by relaxing constraint family 1, P1) is a KP. P1) is NP-hard, hence the design/redesign problem is NP-hard even if we consider only one quality tree and we assume that the quality dependencies are qualitative and hence the relation which can be derived by applying the SAW technique are linear.

In real projects, the design/re-design methodology faces several quality trees and non linear dependencies among quality variables. We are developing a local search approach which is based on the following steps:

– if the design hypothesis is violated, find a feasible solution by focusing iteratively on the most violated constraint;
– the feasible solution obtained in the first step (or the solution which corresponds to the design hypothesis if it is verified) is improved by exploring the current solution neighborhood in order to to find a quasi-optimum solution;
– the optimization technique implements a quality tree partitioning, in order to solve with integer linear programming tools, problems for qualitative dependencies.

We are developing an hybrid optimization approach which interleaves the solution of linear integer programming problems with non-linear problems. We only require to be able to evaluate the value of a quality variable from its children and this requirement is always satisfied since in the worst case scenario quality dependencies are expressed by enumeration, i.e., in tabular form.

5 Conclusions and Future Work

In our previous works we have proposed a methodology for the process of re-design of existing services. Current work is focused on two ongoing research activities. We are extending our methodology to support the design of new services in adaptive multi-channel applications [10]. This new methodology for Design and Re-design of Adaptive Multi-channel Service (DReAMS) is proposed in [11]. The second activity, presented in this paper, faces user quality requirements (UQR) and QoS issues. During the development of a service not only functional requirements but also UQR must be considered. We have pointed out how to perform this task in the different phases of the methodology and how UQR can be associated with a QoS value. Furthermore, we have proposed an hybrid approach which combines linear and non linear optimization techniques and allows verifying design assumption in order to evaluate if the available technologies guarantee the fulfillment of these UQR. The assumptions revision phase has been modeled as an optimization problem with the objective to identify the set of choices for the technical characteristic relevant for the end users and for the service requirements which minimizes design costs. Our research activity has now the aim to investigate deeply the following themes:

1. Consider different classes of end user with different ideal user profiles. Each class u is characterized by different aspects (e.g., expertise, capabilities) and requires a different B_u value for a given QoS dimension. So, the same quality attribute can be associated with multiple constraints: B (service quality request) defined in the high-level redesign phase and several B_u (user quality request) derived from user profiles. We want to analyze how these new considerations impact in the assumption evaluation process and optimization problem formulation. Moreover, we are evaluating the possibility to consider the statistical distribution of users characterized by the same profile. If the statistical distribution of users characteristics and requirements are considered, multiple UQR can be introduced and the optimization problem will

identify a set of solutions characterized by different costs and QoS levels, which will satisfy a given percentage of service users.
2. Develop a semi-automatic tool which supports the designer in the assumption revision process. This tool will implement the optimization problem presented in Section 4.1 considering the distribution channel model revised in [11].

Acknowledgment
The work presented in this paper has been partially supported by FIRB MAIS project - Multi-channel Adaptive Information Systems: models, methodology, qualifying object-oriented platform and architectures for the flexible on-line information systems.

References

1. M. Akbar, E. Manning, G.Shoja, S. Khan, "Heuristic solution for the Multiple-Choice", in Proc. of Conference on Computational Science, 2001.
2. C. Cappiello, P. Missier, B. Pernici, P. Plebani, C. Batini "QoS in multichannel IS: the MAIS approach". In Proceedings of the International Workshop on Web Quality, Munich, 2004.
3. M. Comerio, F. De Paoli, C. De Francesco, A. Di Pasquale, S. Grega, C. Batini "A Re-design Methodology for Multi-channel Applications in the Zootechnical Domain". In Proceedings of the Twelfth Italian Symposium on Advanced Database Systems (SEBD), S.Margherita di Pula (Italy), June 21-23, 2004.
4. M. Comerio, F. De Paoli, S. Grega, C. Batini, C. Di Francesco, A. Di Pasquale, "A service re-design methodology for multi-channel adaptation", in Proc. of the 2nd International Conference on Service Oriented Computing - ICSOC04, New York City, NY, USA, November 15-18, 2004.
5. W. Y. Lum, F. C. M. Lau, "User-Centric Content Negotiation for Effective Adaptation Service in Mobile Computing", IEEE Transaction on Software Engeneering, 1000-1111.
6. MAIS Project: http://black.elet.polimi.it/mais.
7. UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms. OMG report (September 2004).
8. L. Wolsey, "Integer Programming", John Wiley & Sons, 1998.
9. L. Zeng, B. Benatallah, M. Dumas, J. Kalagnamam, H. Chang, "QoS-Aware Middleware for Web Services Composition, IEEE Transactions on Software Engineering, 2004.
10. A. Maurino, S. Modafferi, B. Pernici. Reflective architectures for adaptive information systems, Proc. of First International Conference on Service Oriented Computing (ICSOC), Trento, Italy 2003, LNCS 2910 Springer 2003,pp 115-131
11. F. De Paoli, A. Maurino, C. Batini. A Methodology for Design and Re-Design of Adaptive Multi-channel Services. Submitted Research Paper.
12. P. Graziani, R. Billi, L. Burzagli, Gabbanini, Palchetti, E. Bertini, S. Kimani, L. Sbattella, Barbieri, Bianchi, C. Batini. Definition of User Typologies. (2003) MAIS project internal report R 7.3.1.