Impact Analysis of Goal-Oriented Requirements in Web Engineering

José Alfonso Aguilar[1,2], Irene Garrigós[1], and Jose-Norberto Mazón[1]

[1] Department of Software and Computing Systems
University of Alicante, Spain
[2] Computer Science Faculty
University of Sinaloa, Mexico
{ja.aguilar,igarrigos,jnmazon}@dlsi.ua.es

Abstract. Due to the continuous changes and heterogeneous audience of the Web, a requirement engineering stage is crucial for Web development. Importantly, this stage should consider that Web applications are more likely to rapidly evolve during the development process, thus leading to inconsistencies among requirements. Therefore, Web developers need to know dependencies among requirements to ensure that Web applications finally satisfy the audience. The understanding of requirement dependencies also helps in better managing and maintaining Web applications. In this work, an algorithm has been defined in order to deal with dependencies among functional and non-functional requirements to understand which is the impact of making changes when developing a Web application.

Keywords: Impact analysis, goal-oriented requirements engineering, Web engineering.

1 Introduction

Requirements in Web engineering tend to rapidly evolve due to the dynamic nature of the Web [1]. This continuous evolution may lead to inconsistencies among requirements which hinder Web developers from understanding the impact of a change in the Web application. To tackle this problem, dependencies among requirements should be explicitly considered, thus better managing and maintaining requirements in Web applications. Dependencies are essential elements among the requirements of a real software system to achieve certain stakeholder goals. On the other hand, inconsistencies are the negative dependencies among the set of requirements, caused by the fact that requirements often originate from stakeholders with different or conflicting viewpoints [2].

Impact analysis is the task of identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change [3]. We define a "change" as any modification on a Web requirement. Usually, impact analysis has been done intuitively by Web applications developers, after some cursory examination of the code and documentation. This may be sufficient for

B. Murgante et al. (Eds.): ICCSA 2011, Part V, LNCS 6786, pp. 421–436, 2011.

small Web applications, but it is not enough for sophisticated ones. In addition, empirical investigation shows that even experienced Web applications developers predict incomplete sets of change impacts [4].

Therefore, to effectively manage changes in Web applications, information about dependencies among requirements should be considered in order to know how changes in a Web requirement affect the other requirements, thus making the right design decisions.

Usually, impact analysis is performed only on functional requirements, leaving aside the non-functional requirements as shown in [5,6]. According to [7], we believe that non-functional requirements must be considered in the Web application development from the very beginning of the development process. It is worth noting that the impact analysis should be done on both kind of requirements: functional and non-functional.

Functional requirements describe system services, behavior or functions, whereas non-functional requirements, or quality requirements, specify a constraint on the system or on the development process [8]. Functional requirements are related to *goals* and *sub-goals* in goal-oriented modeling. Non-functional requirements are named *softgoals* in goal-oriented modeling to represent objectives that miss clear-cut criteria. Specifically, finding the right tradeoff for non-functional requirements is an important step for achieving successful software [9,10], i.e., a Web application without passwords is usable, but not very secure, increased usability reduce security or increased security reduce usability. Therefore, it is necessary to identify the dependencies among non-functional and functional requirements for their achievement. Unfortunately, finding this tradeoff is not a trivial task due to the conflicts that commonly arise among them. Interestingly, the recent inclusion of goal-oriented techniques in Web requirements engineering [11,12,13,14] offer a better analysis in Web application design, due to the fact that requirements are explicitly specified in goal-oriented models. This has allowed the *stakeholders* to understand among the design decisions that can be taken to satisfy their goals and evaluating the implementation of certain functional and non-functional requirements in particular. However, this is not enough to ensure that the Web application satisfies the goals.

This paper presents a goal-oriented proposal for supporting Web developers analyzing the impact of a requirement change in Web applications in order to provide information about the different design alternatives. Therefore, more informed design decisions can be made for developing a Web application that fully-satisfies goals, while there is a tradeoff among softgoals.

The remainder of this paper is structured as follows: Section 2 presents some related work relevant to the context of this work. Section 3 describes the proposal for goal-oriented requirements analysis where is found the contribution of this work and introduces a running example for demonstration purposes. The algorithm for impact analysis in goal-oriented requirements is presented in Section 4. The application of the algorithm to perform the impact analysis is described step by step in Section 5. Finally, the conclusion and future work is presented in Section 6.

2 Related Work

In our previous work [15], a systematic literature review has been conducted for studying requirement engineering techniques in the development of Web applications. Our findings showed that most of Web engineering approaches focus on the analysis and design phases and do not give a comprehensive support to the requirements phase (such as OOHDM [16], WSDM [17] or Hera [18]). We can also conclude that the most used requirement analysis technique is UML use cases (applied by OOWS [19], WebML [20], NDT [21] and UWE [22]).

Furthermore, none of the aforementioned Web engineering approaches perform the analysis and modeling of the users' needs for ensuring that the Web application satisfies real goals, i.e. users are not overloaded with useless functionalities, while important functionalities are not missed. We believe that these facts are an important issue that limits a broaden use of these approaches. In this sense, to the best of our knowledge, the only approaches that use goal-oriented requirements analysis techniques for Web engineering have been presented in [23,24]. Unfortunately, although these approaches use the *i** modeling framework [25,26] to represent requirements in Web domain, they do not benefit from every *i** feature. To overcome this situation, our previous work [13] adapts the well-known taxonomy of Web requirements presented in [27] for the *i** framework.

Regarding approaches that consider non-functional requirements from early stages of the development process, in [28] the authors propose a metamodel for representing usability requirements for Web applications. Moreover, in [7] the authors present the state-of-the-art for non-functional requirements in model-driven development, as well as an approach for integrating non-functional requirements into a model-driven development process by considering them from the very beginning of the development process. Unfortunately, these works overlook how to analyze and evaluate the impact among functional and non-functional requirements. However, some interesting works have been done in this area [29] and [30]. These works evaluate *i** models based upon an analysis question (what-if) and the human judgment. To this aim, this procedure uses a set of evaluation labels that represent the satisfaction or denial level of each element in the *i** model. First of all, initial evaluation labels reflecting an analysis question are placed in the model. These labels are then propagated throughout the model by using a combination of set propagation rules and the human judgment. The results of this propagation are interpreted in order to answer the stated question. Unfortunately, these general approaches have not been adapted to Web engineering.

The motivation regarding this proposal relies in the fact that, the works previously mentioned are focused on how to analyze *i** models to answer a particular question (what-if) without considering the goal satisfaction (the organizational objectives). Thus, our proposal is focused on how to evaluate the impact derived from a change in a *i** requirements model in the Web engineering field. For this purpose, in the *i** model the requirements are classified according the taxonomy of Web requirements defined in [27]. Also, our proposal brings out alternative

paths to satisfy the goals bearing in mind the softgoals tradeoff, hence, considering the softgoals from the beginning of the Web application development process.

To sum up, there have been many attempts to provide techniques and methods to deal with some aspects of the requirements engineering process for Web applications. However, there is still a need for solutions that allow an impact analysis by considering non-functional requirements.

3 Goal-Oriented Requirements Analysis in Web Engineering

This section briefly describes our proposal to specify requirements in the context of a Web modeling method by using *i** models [14]. As a goal-oriented analysis technique, the *i** framework focuses on the description and evaluation of alternatives and their relationships to the organizational objectives. This proposal supports an automatic derivation of Web conceptual models from a requirements model by means of a set of transformation rules [31]. The proposal presented in this paper is defined upon our Web engineering method A-OOH *(Adaptive Object Oriented Hypermedia method)* [32] although it can be applied to any other Web engineering approach.

Next, we shortly describe an excerpt of the *i** framework which is relevant for the present work. For a further explanation, we refer the reader to [25,26]. The *i** framework consists of two models: the strategic dependency (SD) model to describe the dependency relationships (represented as ⊣◁⊢) among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The SR model (represented as ◌) provides a detailed way of modeling internal intentional elements and relationships of each actor (◯). Intentional elements are goals (⬭), tasks (⬡), resources (▭) and softgoals (☁). Intentional relationships are means-end links (⊣▷⊢) representing alternative ways for fulfilling goals; task-decomposition links (—┼—) representing the necessary elements for a task to be performed; or contribution links (help / hurt →) in order to model how an intentional element contributes to the satisfaction or fulfillment of a softgoal.

Although *i** provides good mechanisms to model actors and relationships between them, it needs to be adapted to the Web engineering domain to reflect special Web requirements that are not taken into account in traditional requirement analysis approaches. To do this, in first place, we use the taxonomy of Web requirements presented in [27]:

- **Content Requirements.** With this type of requirements, is defined the website content presented to users. For example, in a book on-line store some examples might be: "book information" or "book categories".
- **Service Requirements.** This type of requirement refers to the internal functionality the system as Web application should provide to its users. Following the example of the Content Requirements, for instance: "register a new client", "add book to cart", etc.

- **Navigational Requirements.** A Web system must also define the navigational paths available for the existing users. In this sense, some examples are: the user navigation from index page to "consult products by category" or to "consult shopping cart" options.
- **Layout Requirements.** Requirements can also define the visual interface for the users. For instance: "present a color style", "multimedia support", "the user interaction", among others.
- **Personalization Requirements.** The designer can specify the desired personalization actions to be performed in the final website (e.g. "show recommendations based on interest", "adapt font for visual impaired users", etc.)
- **Non-Functional Requirements.** These kind of requirements are related to quality criteria that the intended Web system should achieve and that can be affected by other requirements. Some examples can be "good user experience", "attract more users", "efficiency", etc.

As the considered Web engineering approach (A-OOH) is UML-compliant, we have used the extension mechanisms of UML to (i) define a profile for using *i** within UML; and (ii) extend this profile in order to adapt *i** to specific Web domain terminology. Therefore, new stereotypes have been added according to the different kind of Web requirements (see Fig. 1): *Navigational*, *Service*, *Personalization* and *Layout* stereotypes extend the *Task* stereotype and *Content* stereotype extends the *Resource* stereotype. It is worth noting that non-functional requirements can be modeled by directly using the softgoal stereotype. The UML-Profile for goal-oriented requirements using *i** has been implemented in Eclipse [33].

A sample application of the *i** modeling framework for Web domain is shown in Figure 2, which represents the SR model of our running example for the Conference Management System (CMS). The purpose of the system is to support

Fig. 1. Overview of the UML-Profile for *i** modeling in the Web domain implemented in Eclipse

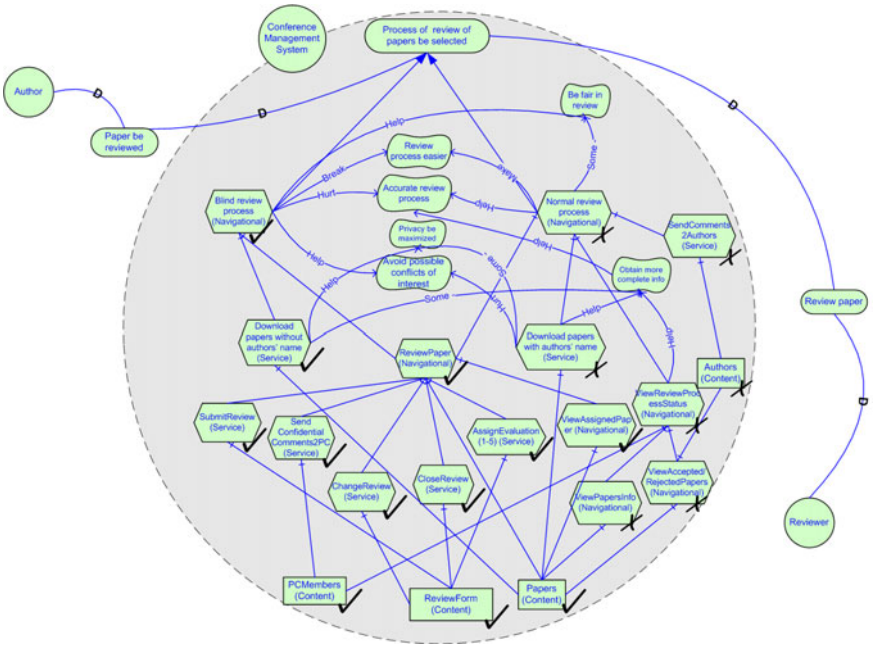

Fig. 2. Part of the Conference Management System requirements expressed in a SR and SD Models

the process of submission, evaluation and selection of papers for a conference [34].[1]

In Figure 2 is modeled a part of the CMS focused in the process of selecting the review process. Four actors participate in the CMS, but due to space limitations, for this example only the author, reviewer and system actors were considered. It is important to highlight that each element from Figure 2 corresponds to a requirements type from the taxonomy previously mentioned, i.e., the content requirement (Content) from the taxonomy is displayed with the notation *"Resource"* from *i** and the navigational (Navigational) and service (Service) requirements with the symbol *"Task"* from *i**, both with their respective associations (*decomposition-links*). A decomposition-link between two elements means that a requirement (*"Task"*) is decomposed in one or more sub-requirements (*"Sub-Tasks"*), Figure 2 depicts a correct scenario of the requirement decomposition by introducing the navigational requirement *"Blind Review Process"*, decomposed in two sub-requirements named *"Download papers without authors' name"* (Service) and *"Review Paper"* (Navigational). Also, the labels (✓) and (✗) are used to represent the requirements currently implemented in the Web application.

[1] The complete specification of the case study can be found at: `http://users.dsic.upv.es/~west/iwwost01`

Three actors are detected that depend on each other, namely *"Reviewer"*, *"Author"* and *"Conference Management System"*. The reviewer needs to use the CMS to *"Review paper"*. The author depends on the CMS in order to *"Paper be reviewed"*. These dependencies and the CMS actor are modeled by a SD and SR models in Figure 2.

The goal of the CMS actor is *"Process of review of papers be selected"*. To fullfill this goal, the SR model specifies that one of the two navigational requirements: *"Blind review process"* or *"Normal review process"* should be performed. In this running example, the path to achieve the goal of the CMS actor is by means of the navigational requirement *"Blind review process"*, all the requirements implemented for this path are labeled with (✓). We can observe in the SR model that some navigational and service requirements are decomposed in other requirements, some of them affects positively or negatively some non-functional requirements, i.e., the service requirement *"Download paper without authors' name"* needs the content requirement *"Papers"*, also, affects positively the softgoal *"Privacy be maximized"* and in some negatively form the softgoal *"Obtain more complete info"*. This fact is very important to see how to satisfy the goal *"Process of review of papers be selected"* considering the Web application softgoals. Therefore, maximizing or minimizing the contribution from requirements to softgoals is a viable solution to find a path to fully-satisfy the goal.

4 An Impact Analysis Algorithm for Goal-Oriented Requirements in Web Engineering

In this section, we present a proposal which provides a way to analyze the impact of a change in an A-OOH conceptual model for the goal-oriented requirement approach for Web engineering described in the previous section. Therefore, the Web developer will be able to evaluate the effect of the change and select the best among several design options to fully satisfy goals based on maximizing softgoals.

The algorithm is designed to be applied in i^* requirements model considering the type of contributions made by the intentional elements to the softgoals. This algorithm allows to evaluate the impact in the requirements model resulting from removing any element of the A-OOH conceptual models. In this way, it is determined which new requirements should be included in the A-OOH conceptual models for maximizing softgoal satisfaction although some requirements have been removed. To this aim, some heuristics have been defined.

4.1 Heuristics

Some heuristics have been defined for determining the impact of the contribution links between intentional element and softgoals. Table 1 summarizes some terms for understanding the heuristics presented in this section. These terms correspond to the most common types of contribution links of the i^* modeling framework.

The *"Help"* contribution link is a partial positive contribution, not sufficient by itself to satisfy the softgoal. The contribution link named *"Hurt"* is partial negative contribution, not sufficient by itself to deny the softgoal. *"Some +"* is a positive contribution whose strength is unknown. *"Some -"* is the opposite contribution type to *"Some +"*, is a negative contribution whose strength is unknown. The *"Break"* contribution link refers to a negative contribution enough to deny a softgoal. Finally, the *"Make"* contribution link is a positive contribution strong enough to satisfy a softgoal.

Table 1. The *i** contributions types

Heuristics Terms	***i**** **Contribution Type**
Strongly-positive	Help
Weakly-positive	Some +
Strongly-negative	Hurt
Weakly-negative	Some -
Dependent-negative	Break
Dependent-positive	Make

The heuristics defined are:

- **H1.** If the contribution of the requirement to remove is *strongly-positive*, and the contribution of the requirement to implement is *strongly-negative*, **do not** implement the requirement.
- **H2.** If the contribution of the requirement to remove is more *strongly-positive* than the contribution of the requirement to implement, but the contribution to be implemented is *weakly-negative*, the requirement **could be** implemented.
- **H3.** If the contribution of the requirement to remove is *strongly-negative* or *weakly-negative*, and the contribution of the requirement to implement is *strongly-positive* or *weakly-positive*, the requirement **should be** implemented.
- **H4.** If the polarity of the contribution of the requirement to remove is as negative as the contribution of the requirement to implement, the requirement to implement **should not be** implemented to maximize the satisfaction of the softgoal.
- **H5.** If the polarity of the contribution of the requirement to remove is as positive as the contribution of the requirement to implement, the requirement to implement **should be** implemented to maximize the satisfaction of the softgoal.
- **H6.** If the contribution of the requirement to remove is *Dependent-positive*, then the developer should consider whether that requirement should be removed or not considering the need to implement this softgoal.
- **H7.** If the contribution of the requirement to remove is *Dependent-negative*, then the developer should consider whether that requirement should be removed or not considering the impact of not implementing this softgoal.

4.2 Preconditions and Postconditions

The preconditions should be launched before the execution of the algorithm and can be only applied to the elements implemented in the requirements model described in Section 3. Specifically, these preconditions permit the execution of the algorithm when:

1. The requirement to remove does not affect the goal by the *"means-end"* contribution type.
2. When there is more than one *"means-end"* contribution type, it means that the impact analysis will be possible by means of the softgoals tradeoff.
3. The requirement to remove affects other requirements and these requirements (not shared) are not in the possible paths to satisfy the goal.

Moreover, there is one postcondition to be applied to the elements defined in the i^* requirements model. This postcondition is applied when a requirement has been selected to be implemented in the requirements model as an alternative solution for the satisfaction of the goal: if the requirement to be implemented has associated requirements, these requirements must be implemented automatically.

4.3 Impact Analysis Algorithm

This algorithm considers the contributions made by the intentional elements from the requirements model to find a path to fully satisfy, where possible, the main goal. To do this, the designer must have to find tradeoffs between the softgoals. The algorithm is presented next.

```
1  FUNCTION TradeOffAlgorithm (RequirementsModel)
2     TR= task to remove; TI= task to implement;
3     SN= new softgoal; IEList= intentional elements list;
4     ASList= affected softgoals list;
5     TIList= list of task to implement; Value= false;
6     P = PreConditions();
7     IF (P=true) THEN
8        IEList= CreateIntentionalElementsList(RequirementsModel);
9        IF(TR.Contributes2Softgoals()) THEN
10          ASList=CreateAffectedSoftgoalsList(TR);
11          FOREACH s FROM ASList:
12             TI= SearchTaskToApply(IEList);
13             Value= Heuristics(ASList,IEList,TI);
14             TI.AddValue(Value);
15             TIList.Add(TI);
16                IF (TI.Contributes2Softgoals(TI)) THEN
17                   ASList.add(SN);
18                END IF
19          END FOREACH
20          FOREACH v FROM TIList:
21             CalculateAverage(v);
22             IF(CalculateAverage(v)) THEN
```

```
23                    Implements(v);
24                  END IF
25                END FOREACH
26              END IF
27            PostCondition();
28            ELSE
29              ShowMessage(P.message());
30            END IF
31  END PROGRAM
```

Algorithm 1.1. Algorithm for impact analysis in goal-oriented Web engineering

A snippet of code that represents our algorithm for impact analysis of goal-oriented requirements in Web engineering is shown in 1.1. First, in lines 6 and 7 the pre-conditions are evaluated (Section 4.2), these must be "true" to proceed with the execution of the algorithm. Next, from lines 8 to 19, the algorithm creates a list of intentional elements. In these lines, all types of requirements from the requirements model are stored in this list. The next step is to extract those softgoals that receive a contribution from the requirement to remove, for each softgoal from the list, finding a non-implemented requirement and applying the heuristics introduced in Section 4.1. Each of these requirements must be stored in the list, and if it contributes to a softgoal, the softgoal must be stored in the list too. Then, lines 20 to 29 are used to evaluate each element from the list according to the weight of each element assigned by the heuristics to determine when a requirement must be implemented. Finally, the postcondition is executed and the alternative path to fully satisfy the goal from requirements model is obtained.

5 Performing the Impact Analysis

Lets suppose the following scenario: the Web developer decides deleting from the domain model of A-OOH the elements that correspond to the requirement *"Download papers without authors' name"*. It is necessary to know which other requirements are affected by this change. In addition, this action implies that the goal *"Process of review of papers be selected"* can not be satisfied. Thus, it is necessary to search for alternative paths in the i^* requirements model (if there any) in order to fully-satisfy the goal *"Process of review papers be selected"*. To this aim, our algorithm is triggered. The execution of the impact analysis algorithm is detailed next.

The first step to execute our algorithm consists of applying the preconditions. For this running example, the preconditions result true, it means that there is any problem to the algorithm has been executed.

Next, it is necessary to develop a list of the requirements (implemented or not) that contribute to any softgoal in the i^* requirements model (see Table 2). Also, if a softgoal contributes to other one, the softgoal must be added to the list too.

Table 2. The requirements contributions to softgoals

Requirements	*"S1"*	*"S2"*	*"S3"*	*"S4"*	*"S5"*	*"S6"*
"Blind review process"	Help	Break	Hurt	Help	-	-
"Download papers without authors' name"	**-**	**-**	**-**	**-**	**Help**	**Some -**
"Normal review process"	Some -	Make	Help	-	-	-
"Download paper with authors' name"	-	-	-	Hurt	Some -	Help
"View review process status"	-	-	-	-	-	Help
"Obtain more complete info"	-	-	Help	-	-	-

Table 2 highlights in bold the requirement to be removed. This table shows a requirements list (functional and non-functional) and their type of contributions to the softgoals where S1 corresponds to softgoal *"Be fair in review"* from requirements model, S2 to *"Review process easier"*, S3 represents *"Accurate review process"*, S4 conforms to *"Avoid possible conflicts of interest"*, S5 its the *"Privacy be maximized"* softgoal and S6 refers to *"Obtain more complete info"*.

The next step is to identify the number of softgoals affected by the requirement to be removed. If necessary, a list of the softgoals that receive a contribution from the requirement to be removed is made. In this example, the requirement to be removed is *"Download papers without authors' name"*, this one affects two softgoals: *"Privacy be maximized"* and *"Obtain more complete info"* S5 y S6 respectively (see Table 2).

For each softgoal that receives a contribution from the requirement to be removed, we search for a non-implemented requirement of which contribution compensates the possible elimination of the requirement to be removed. To do this, it is necessary to apply the heuristics defined in Section 4.1.

For example, the softgoal *"Privacy be maximized"*, according to Table 1, receives a *strongly-positive* contribution (Help) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, only the requirement *"Download papers with authors' name"* contributes (negatively) to this softgoal (*weakly-negative*, i.e. Some -). Therefore, applying the heuristics described in Section 4.1, specifically the heuristic number 2 (H2), the requirement *"Download papers with authors' name"* could be implemented.

Considering the softgoal *"Obtain more complete info"* according to Table 1, it receives a *weakly-negative* contribution (Some -) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, two requirements (positively) contribute to this softgoal, *"Download papers with authors' name"* and *"View review process status"* (*strongly-negative*, i.e. Help). Therefore, the heuristic H3 applies for this softgoal, thus, these requirements should be implemented.

After analyzing the softgoals contributions, the next step is searching for any softgoal in the requirements list that contributes to another softgoal. In this example, the softgoal *"Obtain more complete info"* makes a *strongly-positive* contribution (Help) to the softgoal *"Accurate review process"*, thus, the next step consists of searching for the requirement that makes a contribution to the softgoal and applying the heuristics. Therefore, the requirement that makes a

contribution to the softgoal *"Accurate review process"* is *"Normal review process"*, this contribution is *strongly-positive* (see Figure 2), hence, according to H3 this requirement must be implemented.

After these steps, Table 3 shows requirements that could be implemented to fully-satisfy the goal *"Process of review papers be selected"* after having removed the requirement *"Download papers without authors' name"*. Next, it is necessary to evaluate the heuristics assigned to each requirement to know what could be implemented.

Table 3. Non-implemented requirements that contributes to softgoals

Intentional element	*"S5"*	*"S4"*	*"S6"*	*"S3"*	*Result*
"Download papers with authors' name"	H2 (Some -)	H1 (Hurt)	H3 (Help)	-	Implement
"Normal review process"	-	-	-	H3 (Help)	Implement
"View review process status"	-	-	H3 (Help)	-	Implement

Table 3 shows the results after having performed the algorithm. In this table the requirements that must be implemented in order to fully-satisfy the goal *"Process of review papers be selected"* are shown. To do this, it is necessary to evaluate the contribution type of each requirement, i.e., the navigational requirement *"Download papers with authors' name"* negatively contributes (Some -) to the softgoal *"Privacy be maximized"*, thus hurting the softgoal *"Avoid possible conflicts of interest"* and helping the softgoal *"Obtain more complete info"*. Therefore, by using the human judgment the navigational requirement *"Download papers with authors' name"* can be implemented. For the navigational requirement *"Normal review process"*, it is easier to determine whether it can be implemented because it only contributes to one softgoal, the *"Accurate review process"*, hence its contribution is Help, therefore this requirement must be implemented. Finally, the navigational requirement *"View review process status"* positively contributes to the softgoal *"Obtain more complete info"*, consequently this requirement must be implemented.

The final step is to apply the postcondition from Section (4.2). In this running example, according to the postcondition, it is necessary to implement the navigational requirements *"View papers info"* and *"View Accepted/Rejected papers"* because these requirements are associated with the navigational requirement *"View Review Process Status"*. In addition, the content requirement *"Authors"* and the service requirement *"Send Comments to Authors"* must be implemented too in order to implement the alternative path to fully satisfy the goal *"Process of review papers be selected"*. Hence, the content requirement *"Authors"* is associated with the navigational requirement *"View Accepted/Rejected papers"* and the service requirement *"Send Comments to Authors"* is related with the navigational requirement *"Normal review process"*.

After finishing the execution of the algorithm, we obtain the requirements that are directly and indirectly affected by the deletion of the requirement *"Download papers without authors' name"*. Moreover, the algorithm can find out which requirements must be implemented to continue satisfying the goal considering the contributions received from the softgoals. In this running example the

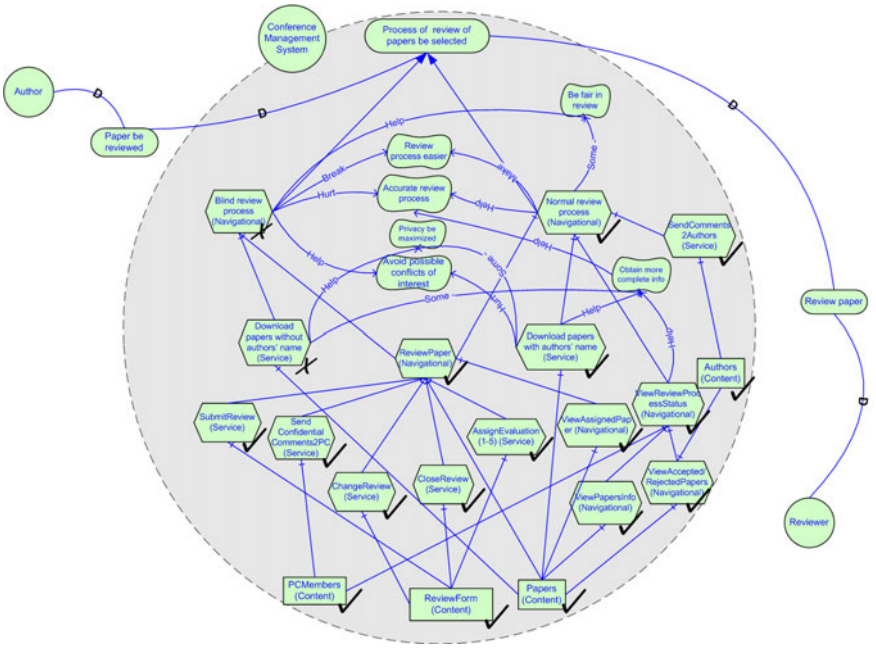

Fig. 3. Conference Management System requirements expressed in a SR and SD Models with the alternative path to fully satisfy the goal *"Process of review papers be selected"*

requirements to implement are: *"Download papers with authors' name"*, *"Normal review process"* and *"View review process status"*. Finally, according to the post-condition the requirements *"View papers info"*, *"View Accepted/Rejected papers"*, *"Authors"* and *"Send Comments to Authors"* must be implemented too. Figure 3 shows the final requirements model with the alternative path implemented to fully-satisfy the goal *"Process of review papers be selected"*.

6 Conclusions and Future Work

Due to the dynamic idiosyncrasy of the Web and its heterogeneous audience Web applications should consider a requirement analysis phase in order to reflect, from the early stages of the Web application development process, specific needs, goals, interests and preferences of each user or user type, and due to the fast evolution of the Web, possible changes in those requirements should also be managed.

In this work, we have presented a methodology based on the i^* modelling framework to specify Web requirements. Moreover, an algorithm to analyze the impact derived from a change done in the requirements model is presented.

Benefits of applying the algorithm described in this work include both the analysis of the impact derived from a change in a conceptual model and the

ability to find an alternative path to fully-satisfy the goal by means of the softgoals tradeoff.

Nevertheless, according to [7], the softgoals are not considered with sufficient importance from the early stages of the development process. In this context, our proposal makes a contribution to the requirements analysis field considering the softgoals, hence it allows the designer to make decisions from the very beginning stages of the development process that would affect the structure of the envision website in order to satisfy users needs. Therefore, the designer can improve the quality of the requirements model analyzing the balance of the softgoals with the stakeholders.

Our short-term future work includes the definition of a metamodel to help to record the relationships among functional and non-functional requirements to adapt the tradeoff algorithm presented in this work.

Finally, note that this work has been done in the context of the A-OOH modeling method, however it can be applied to any Web modeling approach.

Acknowledgments. This work has been partially supported by the following projects: MANTRA (GV/2011/035) from Valencia Ministry, MANTRA (GRE09-17) from the University of Alicante, SERENIDAD (PEII-11-0327-7035) from Junta de Comunidades de Castilla La Mancha (Spain) and by the MESOLAP (TIN2010-14860) project from the Spanish Ministry of Education and Science. José Alfonso Aguilar is subventioned by CONACYT (Consejo Nacional de Ciencia y Tecnología) Mexico and University of Sinaloa, Mexico.

References

1. Ginige, A.: Web engineering: managing the complexity of web systems development. In: SEKE, pp. 721–729 (2002)
2. Zhang, W., Mei, H., Zhao, H.: A feature-oriented approach to modeling requirements dependencies (2005)
3. Arnold, R., Bohner, S.: Impact analysis-towards a framework for comparison. In: Proceedings of Conference on Software Maintenance, CSM 1993, pp. 292–301. IEEE, Los Alamitos (2002)
4. Lindvall, M., Sandahl, K.: How well do experienced software developers predict software change? Journal of Systems and Software 43(1), 19–27 (1998)
5. Zhang, S., Gu, Z., Lin, Y., Zhao, J.: Celadon: A change impact analysis tool for Aspect-Oriented programs. In: Companion of the 30th International Conference on Software Engineering, pp. 913–914. ACM, New York (2008)
6. Gupta, C., Singh, Y., Chauhan, D.: Dependency based Process Model for Impact Analysis: A Requirement Engineering Perspective. International Journal 6
7. Ameller, D., Gutiérrez, F., Cabot, J.: Dealing with non-functional requirements in model-driven development. In: 18th IEEE International Requirements Engineering Conference, RE (2010)
8. Sommerville, I.: Software Engineering, 6th edn. Addison-Wesley, Reading (2001)
9. Boehm, B., In, H.: Identifiing Quality-Requirement Conflicts (2002)

10. Elahi, G., Yu, E.: Modeling and analysis of security trade-offs - a goal oriented approach. Data and Knowledge Engineering 68(7), 579–598 (2009); Special Issue: 26th International Conference on Conceptual Modeling (ER 2007) - Six selected and extended papers
11. Nuseibeh, B., Easterbrook, S.M.: Requirements engineering: a roadmap. In: ICSE - Future of SE Track, pp. 35–46 (2000)
12. Bolchini, D., Mylopoulos, J.: From task-oriented to goal-oriented web requirements analysis. In: WISE 2003: Proceedings of the Fourth International Conference on Web Information Systems Engineering, p. 166. IEEE Computer Society, Washington, DC, USA (2003)
13. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: An mda approach for goal-oriented requirement analysis in web engineering. J. UCS 16(17), 2475–2494 (2010)
14. Garrigós, I., Mazón, J.-N., Trujillo, J.: A requirement analysis approach for using i* in web engineering. In: ICWE, pp. 151–165 (2009)
15. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: Web Engineering approaches for requirement analysis- A Systematic Literature Review. In: 6th Web Information Systems and Technologies (WEBIST), Valencia, Spain, vol. 2, pp. 187–190. SciTePress Digital Library (2010)
16. Schwabe, D., Rossi, G.: The object-oriented hypermedia design model. Communications of the ACM 38(8), 45–46 (1995)
17. De Troyer, O.M.F., Leune, C.J.: Wsdm: a user centered design method for web sites. Comput. Netw. ISDN Syst. 30(1-7), 85–94 (1998)
18. Casteleyn, S., Van Woensel, W., Houben, G.-J.: A semantics-based aspect-oriented approach to adaptation in web engineering. In: Hypertext, pp. 189–198 (2007)
19. Fons, J., Valderas, P., Ruiz, M., Rojas, G., Pastor, O.: Oows: A method to develop web applications from web-oriented conceptual models. In: International Workshop on Web Oriented Software Technology (IWWOST), pp. 65–70 (2003)
20. Ceri, S., Fraternali, P., Bongio, A.: Web modeling language (webml): a modeling language for designing web sites. The International Journal of Computer and Telecommunications Networking 33(1-6), 137–157 (2000)
21. Escalona, M.J., Aragón, G.: Ndt. a model-driven approach for web requirements. IEEE Transactions on Software Engineering 34(3), 377–390 (2008)
22. Koch, N.: The expressive power of uml-based web engineering. In: International Workshop on Web-oriented Software Technology (IWWOST), pp. 40–41 (2002)
23. Bolchini, D., Paolini, P.: Goal-driven requirements analysis for hypermedia-intensive web applications, vol. 9, pp. 85–103. Springer, Heidelberg (2004)
24. Molina, F., Pardillo, J., Toval, A.: Modelling web-based systems requirements using wrm. In: Web Information Systems Engineering (WISE) Workshops, pp. 122–131. Springer, Heidelberg (2008)
25. Yu, E.: Modelling Strategic Relationships for Process Reenginering. PhD thesis, University of Toronto, Canada (1995)
26. Yu, E.: Towards modeling and reasoning support for early-phase requirements engineering. In: RE, pp. 226–235 (1997)
27. Escalona, M.J., Koch, N.: Requirements engineering for web applications - a comparative study. J. Web Eng. 2(3), 193–212 (2004)
28. Molina, F., Toval, A.: Integrating usability requirements that can be evaluated in design time into model driven engineering of web information systems. Adv. Eng. Softw. 40, 1306–1317 (2009)
29. Horkoff, J., Yu, E.: Evaluating Goal Achievement in Enterprise Modeling–An Interactive Procedure and Experiences. The Practice of Enterprise Modeling, 145–160 (2009)

30. Horkoff, J., Yu, E.: A Qualitative, Interactive Evaluation Procedure for Goal-and Agent-Oriented Models. In: CAiSE Forum
31. Aguilar, J.A., Garrigós, I., Mazón, J.-N.: Modelos de weaving para trazabilidad de requisitos web en a-ooh. In: DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Espanol de Informatica (CEDI), Valencia, Espana, SISTEDES, pp. 146–155. SISTEDES (2010)
32. Garrigós, I.: A-OOH: Extending Web Application Design with Dynamic Personalization. PhD thesis, University of Alicante, Spain (2008)
33. Eclipse (2010), `http://www.eclipse.org/`
34. Pastor, O.: Conference procedings. In: IWWOST (2001)