International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 6 (2014) pp. 248 - 256
© MIR Labs, www.mirlabs.net/ijcisim/index.html

Goal Based Requirements Analysis Using WebURN

Shailey Chawla[1], Sangeeta Srivastava[1] and Deepak Malhotra[2]

[1] Department of Computer Science, University of Delhi
North Campus, Delhi-110007
shaileychawla, sangeeta.srivastava@gmail.com

[2] Consulting, IT, Mazars, Gurgaon, India
deeps2281@gmail.com

Abstract: **Web applications have specific functional and non-functional requirements owing to their worldwide presence and heterogeneous audience. It has been proved that including NFRs from early requirements analysis builds a product that needs lesser changes, is coherent with the expectations of stakeholders and reduces the design and development flaws. The existing Web engineering approaches, however, fail to analyze the non-functional requirements and their influence on the system. User Requirements Notation is a standard that combines goals, softgoals and scenarios for early requirements analysis. URN is a combination of Goal Requirements Language (GRL) which annotates goal modeling and UCM (use case maps) for depicting scenarios. In this paper we initially present, Web specific User Requirements Notation (WebURN), an enhancement of User Requirements Notation for Web specific functional and non-functional requirements, which is driven by goals and softgoals. The WebURN notation likewise, consists of WebGRL notation and WebUCM notation. In our framework, first, the goals and softgoals are captured and represented using base WebGRL diagram. Thereafter, it is refined in detail for every functional requirement. Simultaneously, the softgoals corresponding to each functionality based requirement are also analyzed. Finally, the WebGRL and WebUCM diagrams are validated and checked for any inconsistencies. Additionally, to support the Requirements engineer throughout the process, guidance is provided on the development of Goal driven Requirements Analysis diagrams based on WebURN. A tool has been developed to automate the above stated steps.**

Keywords**:** User Requirements Notation (URN), Goals, Scenarios, Web Requirements Engineering, GORE**.**

I. Introduction

Web applications have invaded our lives in all aspects, whether it is banking, online shopping, information seeking, business showcasing, e-learning or entertainment. Hence, there has been increased focus on improving and innovating the engineering of Web applications such that they capture the user's expectations and stakeholders goals comprehensively. The Web engineering approaches have been developed in the last decade to cater to specific needs of web applications like heterogeneous and vast audience, difficult elicitation, global appeal, and navigation and personalization issues. These approaches [1-5] focus on developing web applications keeping in mind web specific functional requirements. However, non-functional requirements are not catered properly in these approaches, and even if so they are treated superficially. The web applications are designed mostly keeping in mind its presentation aspects in limited time frames, the burden comes mostly on Web designers, overlooking the detailed requirements analysis and sometimes ignoring the stakeholder's expectations. However, with the current scenario, detailed requirements analysis has become very important so that Web applications build are closer to the expectations of stakeholders and cost of re-doing the development process is saved. It is important hereto to undergo comprehensive requirements analysis to capture stakeholder's goals to the maximum. For doing so, Goal oriented Requirements Engineering techniques [6] [7] [8] are the closest approaches for fulfilling goals of Actors and detailed analysis of requirements. The GORE techniques enhance the requirements analysis in many ways, the conflicts between requirements can be detected early and design alternatives can be evaluated and selected to suit the requirements. GORE is an approach used to uncover, analyze, and describe stakeholder goals leading to software and system requirements [9]. This popular approach, supported by a large international community for more than 15 years, has proven successful in assisting requirements and software engineers in their activities by enabling novel types of analyses over non-functional properties of systems and facilitating the documentation of design rationale [9]. Numerous languages and notations have been developed over the years to model goals and their relationships in an explicit way. More recently, the Goal-oriented Requirement Language (GRL) has become an internationally recognized standard for goal-oriented modeling, as part of a new Recommendation of the International Telecommunications Union named User Requirements Notation (URN) [8]. URN being the latest and standardized notation for Goal and Scenario based requirements analysis, our work is based on this notation.

In this paper we present a Web specific Goal-oriented approach for detailed requirements elicitation and analysis. Incorporation of Goals in early requirements analysis helps in making design decisions, resolving conflicts, prioritizing requirements and focusing on softgoals for achieving better satisfaction of the stakeholders. The detailed requirements analysis and inclusion of NFRs during early analysis help the requirements engineer and the Web designer in many ways. They have good insight of what's to be developed before the development and design of the Web application.

The rest of the paper is organized as follows. In the next section we go through the related work in the area of Goal oriented Web requirements engineering. In the third section we form the background of our framework where we explain the User Requirements Notation and our previous work. Next, we explain the Web specific User Requirements Notation. In section 5, the algorithms for elicitation, refinement and validation of Goal and Scenario diagrams using WebURN are explained with the help of a case study on an Online Book Shop. We conclude our work hereafter and specify the future work.

II. Related work and Background

Goal oriented Requirements Engineering approaches have been extensively used in Information systems development over the last decade. However, with their potential of comprehensive requirements analysis, they haven't been explored much in Web applications domain. The works that we came across that actually applied GORE approaches to Web application Engineering are AWARE [10] and the works in [11] [12]. In AWARE, D. Bolchini has applied i star based approach to static hypermedia based applications. The goal analysis and softgoal representation and dependencies have been captured using i star and it has been transformed to a web engineering methodology. The Functional Requirements have been classified in the web context and softgoals have also been depicted. However, the effect of the dependencies between the softgoals has not been evaluated in this work.

In another more recent work [13], authors have extended i* modeling approach for web specific functional requirements. After early requirements analysis using i*, they have transformed into a Web engineering approach. The authors have also made an attempt at optimizing the non-functional requirements using Pareto Optimizing Algorithm [14]. However, the non-functional requirements have not been classified or explored in details.

Seeing the potential of Goal oriented Requirements Engineering techniques and the importance of Web application in today's world made us study this area in detail. And though there have been aforesaid work, it has been realized that softgoals or NFRs play a major role in design decisions and prioritization of requirements, thus including them from early phases of requirements engineering would improve the web applications.

Amongst other GORE approaches we have chosen URN because currently it is the only standard that incorporates Goals and Scenarios together. URN has the elements of popular GORE approaches i* and NFR. Following text, describes URN in brief, so that our approach and notation can be better understood.

User Requirements Notation [8] is currently the only standard by International Telecommunication Union for Goal based early requirements engineering. It is a combination of Goal Requirements Language (GRL) for reasoning with goals and Use Case Maps (UCM) for creating Scenarios. User Requirements notation aims to capture goals and decision rationale that finally shape a system and model dynamic systems where behavior may change at run time. GRL is *Goal Requirements Language* that focuses on Goal analysis. It help in defining the goals including the non-functional requirements, evaluating them, resolving conflicts etc. UCM stands for *Use Case Maps* that are the visual notation for scenarios. UCM notation employs scenario paths to illustrate causal relationships among responsibilities. The combination of GRL and UCM as depicted in Fig. 1 helps to improve the definition of new goals and satisfy them.

GRL as described by Amyot in [8] supports five kinds of intentional elements. **Goal**: Quantifiable high-level (functional) requirement (rounded cornered rectangle).**Soft goal**: Qualifiable but unquantifiable requirement, essentially non-functional (irregular curvilinear shape).**Task**: Operationalized solution that achieves a goal or that *satisfices* a soft goal which can never be fully achieved due to its fuzzy nature (hexagon). **Resource**: Entity whose importance is described in terms of its availability (rectangle). **Belief**: Rationale or argumentation associated to a contribution or a relation (ellipse). There are also five categories of intentional relations, which connect elements. **Contribution**: Describes how soft goals, tasks, beliefs, and relations contribute to each other. Each contribution can be qualified by a degree: equal, break, hurt, some-, undetermined, some+, help, or make. **Correlation**: Contribution that indicates side-effects on other intentional elements (dashed line). **Means-end**: Link for tasks achieving goals. Different alternatives are allowed. **Decomposition**: Defines what is needed for a task to be performed (refinement), always AND. **Dependency**: Link between two actors depending on each other (half-circle). UCMs have following basic concepts according to Amyot [8]. **Start point**: Captures preconditions and triggering events (filled circle).**Responsibilities**: locations where computation (procedure, activity, function, etc.) is necessary (cross).**End point**: Represents resulting events and post-conditions. **Paths**: Connects start points to end points and can link responsibilities in a causal way. **Component** represents an abstract entity (object, server, database etc.)(Rectangle).

- The User Requirements Notation depicts Generic Goals and Softgoals. For applying URN to Web Requirements Engineering, the notation should be able to depict Web specific functional and non-functional Requirements. The Web specific functional requirements have been classified by many authors [15] into Content, Navigation, Business Process, Adaptation or Personalization and Presentation Requirements. There's no such classification for non-functional Requirements for Web applications. In our paper [16], we have defined and classified the Web specific non-functional requriements according to various categories. The non-functional requirements for Product, Functionality, Project, Environment, Organization, Actor, Legal issues have been classified and defined in details. For example, for Actor's NFRs, user friendliness,

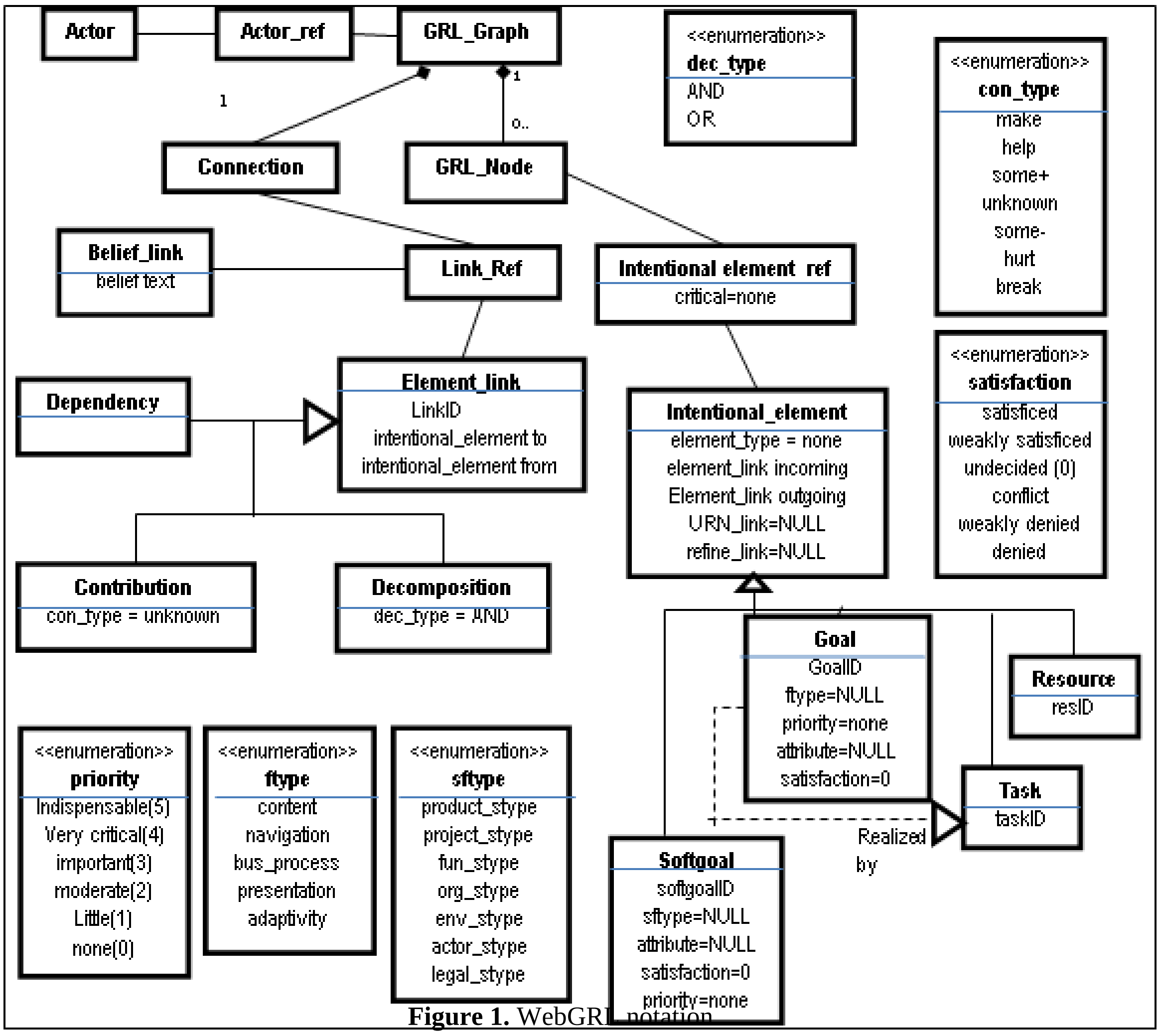


Figure 1. WebGRL notation

empathetic and Understandeability are defined. NFRs are also defined specific to each functional requirement like for navigation requirement the corresponding NFRs are Accessibility, Consistency, Predictability, Relevance and Convenience. For further details and definition the reader is referred to [16]. WebURN notation incorporates all the above classified functional and non-functional requirements in the form of Goals and Softgoals to comprehensively cover the Web application development needs.

III. WebURN notation

We have enhanced the existing standard URN notation to suit the web specific needs. The two parts of URN- GRL & UCM have been enhanced suitably. The extended notation for analysing goals is named as WebGRL and the Use case maps have been enhanced for incorporating navigation in the scenarios and termed as WebUCM. The enhanced notation clearly depicts Web specific Functional & Non-Functional Requirements. The metamodel on which the notation has been based is described in [17] and is depicted in Figure 1 and Figure 2 for WebGRL and WebUCM respectively. In our framework, first a Base WebGRL diagram is constructed that captures the goals from the stakeholders initially, it’s like the level 0 of refinement. At this level, the non-functional requirements pertaining to entire system are considered like product, project-specific and environmental factors, legal, actor’s NFRs. Non-functional requirements related to functional requirements like completeness and readability for content requirements, responsiveness and simplicity for business-process requirements etc. are defined in the next levels, where WebGRLs are constructed specific to that functionality for in depth analysis of functional requirements. WebUCMs are mainly created for Navigation and Business process requirements where detailed walkthroughs need to be explained. The work in our previous papers was depicted using the tool jucmnav [18] that supports User Requirements Notation. However, to exhibit extension of URN for Web applications and other added functionalities like validation, we have developed a separate tool supporting requirements analysis based on WebURN. The tool has been built on Microsoft Visio 2013 platform [19] and the programming has been done in Visual Basic. The framework for Goal based requirements analysis for web applications has been discussed in brief in [17].
To brief up the notation, the goals and softgoals shape is same, just a circle has been added with the initial of corresponding type of requirement.

For UCM diagrams, the stubs have been categorized into generic and business process stub. This is used when navigation UCM has to refer to a Business process or a transaction to be carried out. The responsibility shapes have been modified to include the navigation elements like menu's, image collections, index etc. The notation has tried to cover the main components required to explain and analyse navigation requirements. The designer would get a clearer picture of what he's going to create with the help of these web specific diagrams. The stencils that have been created for WebGRL and WebUCM in Microsoft visio have been shown in Figure 3.

example of an online book shop that sells books of various subjects online. It is an example of an e-business web application, primarily to gain profit by increasing the number of customers, and retaining the old ones by some incentives. The elicitation of goals is done through Algorithm 1 given below. Elicitation can be done by a questionnaire and input to the algorithm. In our work, elicitation can be done through a questionnaire that directly leads to inputs of this algorithm or this algorithm leads to the establishment of Base WebGRL diagram. This algorithm is required so that the goals and softgoals of the stakeholders are gathered in a focused manner which is specific to Web applications and emphasizes on

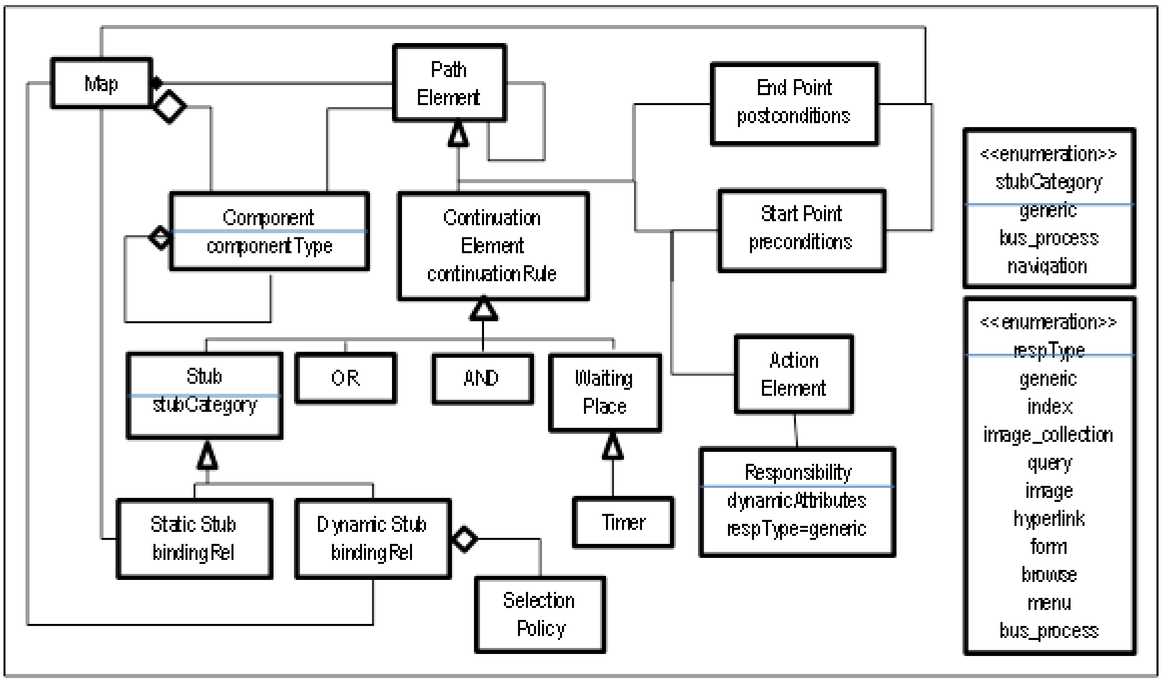

Map
Path Element
End Point postconditions
Component componentType
Continuation Element continuationRule
Start Point preconditions
<<enumeration>> stubCategory generic bus_process navigation
<<enumeration>> respType generic index image_collection query image hyperlink form browse menu bus_process
Stub stubCategory
OR
AND
Waiting Place
Action Element
Responsibility dynamicAttributes respType=generic
Static Stub bindingRel
Dynamic Stub bindingRel
Timer
Selection Policy

Figure 2. WebUCM notation

IV. Process of Goal based Elicitation and Analysis of Web application Requirements

We shall explain the series of algorithms based on an example for better understanding of our framework for goal driven Elicitation and analysis of Web application Requirements. The steps would be as follows:

1. Elicitation and Analysis of Goals and Softgoals for Generating Base WebGRL diagram.

2. Validation of Base WebGRL diagram.

3. Transformation and Refinement of Base WebGRL diagram into WebGRL diagrams of specific functionality.

4. Validation of each WebGRL diagram.

5. Generation of WebUCM diagram for depicting Scenarios.

6. Validation of WebUCM diagram.

Please note that validation of WebGRL diagram can be done by same algorithm for Base WebGRL diagram and Functionality specific WebGRL diagrams. We take the better elicitation of softgoals of the stakeholder. It brings out requirements such that repetitive reinforcement of the elicitation procedure is minimized. In other techniques, the requirements engineer is normally given a canvas where he can drag and drop the items and do the analysis, where a lot is left on the software engineers and then, it is re-verified with the stakeholders. Here, the model elicits in such a way that not many cycles are required for reaching the final analysis model. The algorithm validated the WebGRL diagram constructed by the user and points out the errors to him. The framework for Web application using a GORE based approach starts from early requirements analysis, the approach aids in the elicitation process, enhances the analysis and reasoning by including the softgoals and their influence on other requirements from early stages of requirements engineering and validates the diagrams created by requirements engineer as well. The framework for Web Requirements Engineering using GORE approach had been discussed in our previous work in [17] has been depicted in Figure 4..

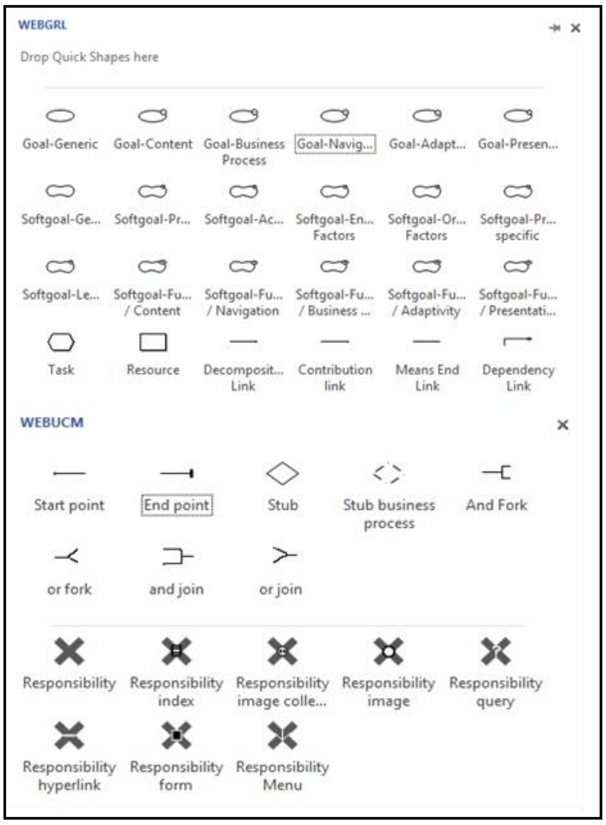


Figure 3 WebGRL and WebUCM Stencils created in visio

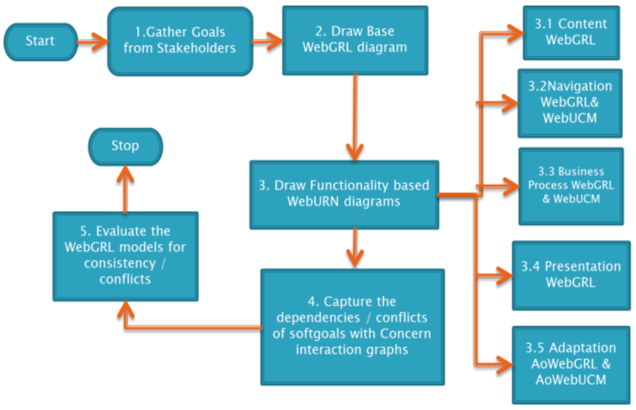

Start
1.Gather Goals from Stakeholders
2. Draw Base WebGRL diagram
3.1 Content WebGRL
3.2Navigation WebGRL& WebUCM
3.3 Business Process WebGRL & WebUCM
3.4 Presentation WebGRL
3.5 Adaptation AoWebGRL & AoWebUCM
3. Draw Functionality based WebURN diagrams
4. Capture the dependencies / conflicts of softgoals with Concern interaction graphs
5. Evaluate the WebGRL models for consistency / conflicts
Stop

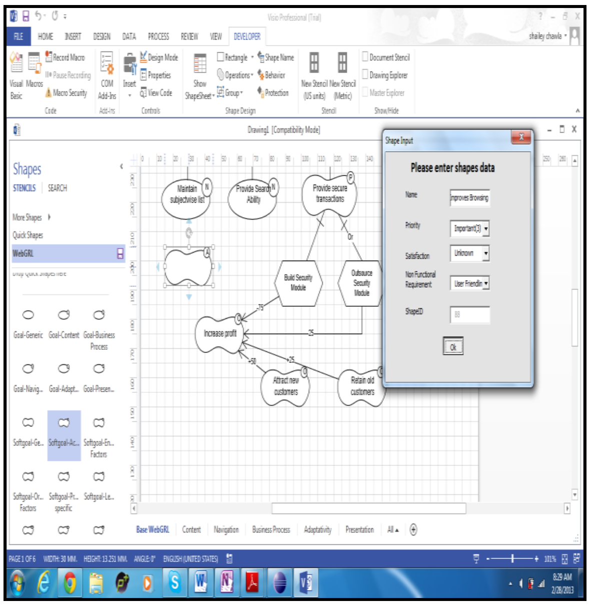
Figure 1 Framework of Goal based Web Requirements Engineering [17]

Algorithm 1: Algorithm for Elicitation of Goals and Softgoals for Generating BaseWebGRL diagram

```
Algorithm: BaseWebGRL_generate
Input: Intentional Elements :{ Goals, Softgoals, Resource, Tasks}, Links

GOAL *g[ ]=NULL;         //list of goals and sub goals
SOFTGOAL *s[ ]=NULL;     //list of softgoals
RESOURCE *r[ ]=NULL;     //list of resources
TASK  *t[]=NULL          //list of tasks
ELEMENT_LINK *L[ ]=NULL;       //list of links

print "Input the primary goals";
do
{
  On Drag and drop event of the goal of specific ftype
   g[i].id=unique_id();
   Update g[i].ftype from the shape
   Input g[i].attribute, g[i].ftype g[i].satisfaction, g[i].priority
   i++
 } while(event)
for each g[i] to be refined into subgoals
  {
  On Drag and drop event of the goal of specific ftype
   g[i].id=unique_id();
   Update g[i].ftype from the shape
  Input g[i].attribute, g[i].ftype g[i].satisfaction, g[i].priority
  i++
  Print "Use a decomposition link- AND or OR
  On drag and drop event of the link
  Update l[i].type, l[i].from, l[i].to, l[i].weight
    }
Print "Add the softgoals"
do{On drag and drop event of softgoal,s[i]
    Input s[i].attribute, s[i].priority
   On drag and drop event of the link(s)
  Update l[i].type, l[i].from, l[i].to, l[i].weight
}while(event)
Print "Add more goals/softgoals"
On drag and drop event of softgoal,g||s[i]
    Input g||s[i].attribute, g||s[i].priority
   On drag and drop event of the link(s)
  Update l[i].type, l[i].from, l[i].to, l[i].weight

  print "any goal need to be operationalized to task"
   for each s[i] || g[i]
        On drag and drop event of task
Update t[i].name
Print "means end link for operationalization of goals"
On drag and drop of link
Update l[i].from, l[i].to, l[i].weight, l[i].type
print "any resource needed for the web application"
 on drag and drop event of resource
update r[i].name
 On drag and drop of link
Update l[i].from, l[i].to, l[i].weight, l[i].type

Output the Base WebGRL diagram with the above inputs.
```

The algorithm for construction works according to the Guidance and instructions given in the tool on how to proceed for obtaining a WebGRL diagram. The user is instructed to place goals and then the subgoals, along with the links. The moment any shape is dragged on to the drawing area, a form pops up that asks the user of necessary details like the name, the priority, the satisfaction factor and generates a unique id for each shape when it is dropped. The information is input through a form, the name is portrayed in the diagram and the rest of the information is stored in the back end as an excel file. The excel file at the back records information about each and every shape and the links so that this information can be used later on, even when the diagram is not present. The snapshot of the tool when the algorithm asks for input at the event of drag and drop of a shape is given in Figure 5. The algorithm outputs the Base WebGRL diagram that depicts the general Goals and Softgoals of the stakeholder. Various functional goals and the softgoals of category: product, project specific and environmental factors are depicted at this level. A snapshot of excel file where shapes data gets recorded is also shown in Figure 6.

In the online book shop example, user inputs his primary goal of Selling books and subgoals of providing information, facilitating payments and maintaining customer details. At the first level the softgoals like User-friendliness, Security, Organizational objectives , project specific softgoals are input by the user.


Please enter shapes data

Name	Improves Browsing
Priority	Important(3)
Satisfaction	Unknown
Non Functional Requirement	User Friendli
ShapeID	

Figure 5 Snapshot of the tool in Microsoft Visio when a Softgoal -Actor has been dragged and dropped on the drawing area

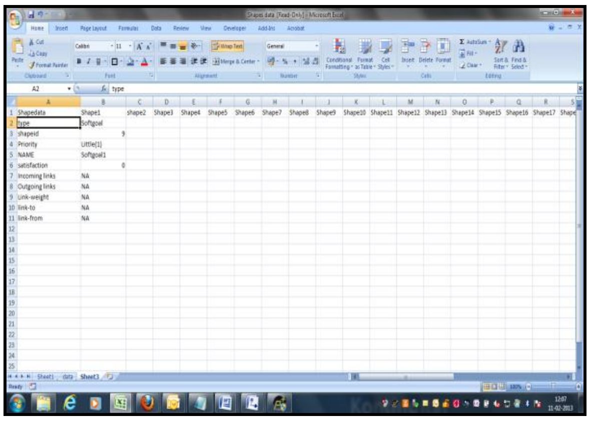

A	B	C	D	E	F	G	H
Shapedata	Shape1	shape2	Shape3	Shape4	Shape5	Shape6	Shape7
type	Softgoal						
shapeid	9						
Priority	Little(1)						
NAME	Softgoal1						
satisfaction	0						
Incoming links	NA						
Outgoing links	NA						
Link-weight	NA						
link-to	NA						
link-from	NA						

Figure 6 Snapshot of Excel file where data of all the shapes gets stored

The diagram created by the user should be validated for certain errors so that the mistakes are not carried forward to next levels. The errors like empty diagram, unconnected diagram components etc. are checked and diagram is validated for such errors. A validation algorithm checks for inconsistencies in the diagram and reports it to the user, refer Figure 7. Input is the WebGRL diagram with all the links and intentional elements in place. The output is a list of errors/ inconsistencies in the form of text as a dialog box after the validation is completed. The validation algorithm is given as follows:-

Algorithm 2: Algorithm for Validation of WebGRL diagram

```
Algorithm: WebGRL_Validate
Input: webGRL diagram
Output: Errors in text form

Intentional_element e[ j ]
Softgoal s[], task t [ ] , goal  g [ ], link l [ ]

For each e[j]
If e[j].incoming_link && e[j].outgoing_link ==NULL
    Print "The element is not connected"
If e[j].elementtype="softgoal"
    For each s[i]
    If(s[i].incoming_link.from != ( t[i] || g[i] )
Print "atleast one task or goal should contribute to softgoal"
For each l[i]
If l[i].from==l[i].to
    Print "from and to link shouldn't be same"
If l[i].from==NULL && l[i].to ==NULL
    Print "the link should have both ends connected to intentional elements"
```

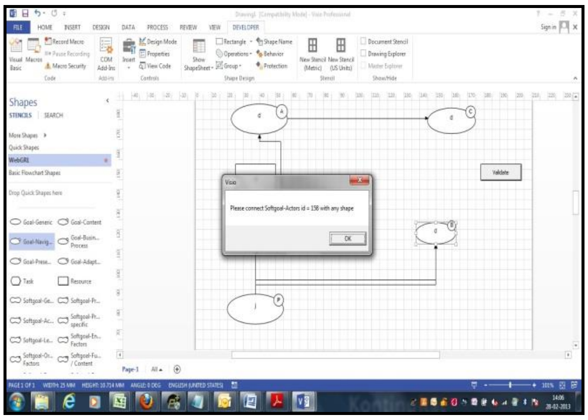

Please connect Softgoal-Actors id = 158 with any shape

Figure 7 Error being displayed when an element is not connected

After the BaseWebGRL diagram has been generated, for detailed analysis, it is refined for each functional requirement category i.e. for Content, Navigation, Presentation, Business Process and Adaptivity requirements. For refinement and further elicitation the following Algorithm is applied for each functional requirement. In each case, from the base webGRL the goals of the corresponding functional category like content, navigation etc. are captured along with their incoming and outgoing links and directly connected elements. The next drawing page consisting of these elements from Base WebGRL serve as guidance or initialization for further refinement. Then the user can enhance it by adding more elements. The same algorithm is repeated for Content, Navigation, Business Process, Adaptation and Presentation Requirements.

These algorithms give the designer details on various Web specific aspects of the system and greatly reduce the ambiguity of the application.

Algorithm 3: Algorithm for transformation and refinement of BaseWebGRL diagram into Specific Functional WebGRL diagram

```
Algorithm:RefineBaseWebGRL
Input : BaseWebGRL diagram, Intentional
elements:{Goals:Content,Softgoals, Resource, Tasks} :Text, Links, URN
links,
Output: SpecificWebGRL:  graphical{}

//  Capture the Goals and corresponding links of Specific Functionality f
for each g[i].ftype=f        //GOAL in BaseWebGRL diagram with Matching
Functionality type
        Capture the goal g[i], its incoming and outgoing links and connected
Intentional Elements
for (i=1;i<=n;i++)
   print "Can g[i] be refined into subgoals(y/n)"
   input ch;
   if ch=='y'
       Input x //no. of subgoals
        Set   g[i].outgoing_link= decomposition.d1(x); //
       for (i=1;i<=x;i++)
         GOAL sg[i]=goal(name, ftype, incoming_link);
         for each (sg[i])
  print "can g[i] be refined into subgoals(y/n)"
               input ch;
               if ch=='y'
         Input x //no. of subgoals
         Set   sg[i].outgoing_link= decomposition.d1(x);
         for (i=1;i<=x;i++)
 GOAL sg[i]=goal(name, ftype, incoming_link)
         repeat while *sg[i].outgoinglink

for each sub-member SOFTGOAL, s[i]
    Input  s[i].soft_relevance
    If s[i].soft_relevance > 0 //
    Input s[i].attribute,  s[i].incoming_link , s[i].outgoing links.
    update the incoming / outgoing links of corres. goals.

    repeat if s[i].attribute need further refinement
print "any goal need to be operationalized to task(y/n)"
if yes
   for each s[i] || g[i]
        TASK t[i]=task(s[i]||g[i], meansend_link)
        update link of s[i]||g[i]
print "any resource needed for the web application"
RESOURCE r[i]=resource(s[i]||g[i], incoming_link, outgoing_link).
 if  "any g[i] needs to depicted as UCM for scenario based explaination"
     Insert a URNLINK in g[i]
On double_click event of URN link
                Call GenerateUCM( )

Output the  WebGRL diagram for current functionality, f  with the above
inputs.
```

The goal or task that need further explanation is depicted with a URN link, that leads to a scenario based UCM diagram construction. When URN link is provided at any task or goal for further explanation with the help of scenarios, the users are advised to put a URN link at that goal in the above algorithm. At the double click event of the URN link a UCM map is constructed. The following algorithm explains the inputs for creating a scenario for Use Case Maps. Figure 8 depicts a fragment of the UCM for the browse books goal/task. It is a WebUCM diagram that depicts the navigation scenarios. As depicted in the WebUCM notation above, there are different notations used for different navigable units like index, menu, image collection etc. Its significant to mention here that treatment of navigation is a very important part of Web application development and it has been studied in [20] that the navigation requirements that were studied during late analysis or design phase earlier need to be considered from the initial phases of Requirements Engineering.

Algorithm 4: Algorithm for construction of Use Case Maps for a walkthrough related to a specific Goal.

```
Algorithm: GenerateUCM
Input: Path p, Responsibility r, static_stub ss, dynamic_stub ds, or_fork of,
or_join oj, and_fork af, and_join aj
Output: UCM maps
On drag and drop event of a path
{   Set  p[i].start_point, p[i].end_point
              input p.startpoint(preconditions)
              Input p.endpoint( postconditions)
}
On drag and drop event of responsibility
{
    Generate a unique id r[i].id
    Input r[i].name
    Add r[i] to p[i]
}
On drag and drop event of stub
{   Generate  ss[i].id
    Generate ss[i].name
    Generate ss[i].urn_link
    Call function generate_UCM()
}
On drag and drop event of 'or fork' of[i]
    Generate n p[i].of[i].end_point[]  //different end points for different path
splits.
On drag and drop event of 'or join' oj[i]
    Combine p[i].of[i].end_point[]
On drag and drop event of 'and fork' af[i]
    Generate n p[i].af[i].end_point[]  //different end points for different path
splits.
On drag and drop event of 'and join' aj[i]
    Combine p[i].af[i].end_point[]
end
```

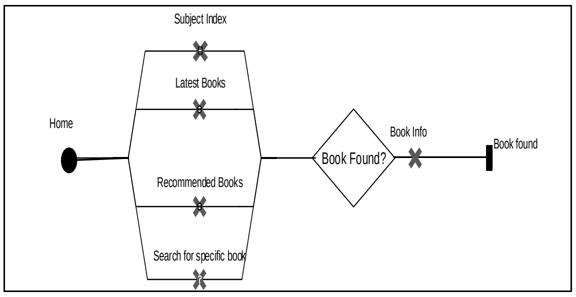


Figure 2 WebUCM for Browse book scenario (fragment)

The WebUCM diagrams can also be validated for any errors; the following algorithm validates the diagram for any incompleteness and inconsistency. If any discrepancies are found the errors are reported to the user in the form of text. The following algorithm explains the validation of WebUCM maps.

Algorithm 5: Algorithm for Validation of WebUCM diagram

```
Algorithm:  WebUCM_Validate
Input: webUCM diagram
Output: Errors in text form

Path p[ j ], Startpoint s, End point e[i]
Responsibility [r], Stub s[]
Or_fork of[], Or_join oj[]
And_fork af[ ], And_join aj[]
```

```
For each path p[j]
If p[j].s && p[j]->e ==NULL
    Print "The path should have both start and end point"
If p[j].of!=NULL XOR p[j].oj==NULL && p[j].end_point<=1
Print "Every OR fork should have corresponding OR join or lead to multiple
end points "
If p[j].af!=NULL XOR p[j].aj==NULL && p[j].end_point<=1
Print "Every AND fork should have corresponding AND join or lead to multiple
end points "
IF p[j]==NULL
    Print " empty diagram"
```

The algorithms and process explained in this section ensure a comprehensive and web requirements specific modeling. Though many authors have worked on separate requirements like [21][22][23], we have dealt with the problem of Web Requirements Engineering in totality considering all functional and non-functional requirements.

V. Conclusion and Future Work

In this paper, we have exhibited the Goal driven Web Requirements Engineering approach using URN. We have developed tool support in Microsoft Visio that supports the construction, analysis, refinement and validation of the WebURN diagrams. The approach supported with the tool guides and helps the Requirements Engineer and the Web designer to get better insight of the Web application that is to be developed. The Softgoals have been incorporated in the early requirements analysis and their influence and collaborations with other goals are analyzed that thus focus not only on functionality of the application but also on the qualitative aspirations and constraints of the system.

Our future work includes applying reasoning to WebURN, that is, the goal based notation of Web Requirements Engineering. The reasoning would help in resolving conflicts and selecting design alternatives. We would also enhance and improve the tool for ease of use. Further we can also include aspect oriented modeling techniques for modeling personalization requirements because these are crosscutting in nature.

References

[1] A. Bonnaccorsi. "On the Relationship between Firm Size Baresi L, Garzotto F, Paolini P. Extending UML for modelling web applications. In: Annual Hawaii international conference on system sciences, Maui, USA, 2001. p. 1285–94.

[2] Rossi G. An object oriented method for designing hypermedia applications. PhD Thesis, University of PUC-Rio, Rio de Janeiro, Brazil, 1996.

[3] Schwabe D, Rossi G. Developing hypermedia application using OOHDM. In: Workshop on hypermedia development processes,methods and models (Hypertext 98), Pittsburgh, USA, 1998.

[4] Ceri S, Fraternali P, Bongio. Web modelling language (WebML): a modelling language for designing web sites. Conference WWW9, Mayo. Comput Networks 2000;33(1–6):137–57.

[5] UWA requirements elicitation: model, notation, and tool architecture, 2001. www.uwaproject.

[6] Mylopoulos, J., Chung, L., Yu, E.: 'From Object-Oriented to Goal-Oriented Requirements Analysis'. Communications of the ACM 42(1) (1999).

[7] Castro, J., Kolp, M., Mylopoulos, J.: Towards Requirements-driven Information Systems Engineering: the Tropos Project. Information Systems 27, 365–389 (2002)

[8] ITU-T, Recommendation Z.151 (11/08): User Requirements Notation (URN) – Lanuage Definition.

[9] van Lamsweerde, A. Requirements Engineering: From Craft to Discipline. In:Proc. 16th ACM SigSoft Int. Symp. on the Foundations of Software Engineering (FSE'2008), Atlanta, USA; 2008.

[10] Bolchini, D., Paolini, P., "Goal-Driven Requirements Analysis for Hypermedia-intensive Web Applications", Requirements Engineering Journal, Springer, RE'03 ,pp. 85-103.

[11] Jaap et al, e-Service design using i* and e3 value modeling, vol 23, 2006, IEEE software.

[12] Azam et al 2007, Integrating value based requirements engineering models to WebML using VIP business modeling framework.

[13] Aguilar, Garrig´os, I., Maz´on., Trujillo, J.: An MDA Approach for Goal-Oriented Requirement Analysis in Web Engineering. J. Univ. Comp. Sc. 16(17),2475–2494 (2010)

[14] José Aguilar, Irene Garrigós, Mazón: A Goal-Oriented Approach for Optimizing Non-functional Requirements in Web Applications. ER Workshops 2011: 14-23.

[15] Koch, N., Escalona, M.: Requirements Engineering for Web Applications – A Comparative Study. Journal of Web Engineering 2(3), 193–212 (2004).

[16] Shailey Chawla, Sangeeta Srivastava, Punam Bedi, "GOREWEB Framework for Goal Oriented Requirements Engineering of Web Applications", S. Aluru et al. (Eds.): IC3 2011, CCIS 168, pp. 229–241, Springer-Verlag Berlin 2011.

[17] Chawla, Shailey, and Sangeeta Srivastava. "A Goal based methodology for Web specific Requirements Engineering." *Information and Communication Technologies (WICT), 2012 World Congress on*. IEEE, 2012.

[18] Mussbacher, Gunter, and Daniel Amyot. "Goal and scenario modeling, analysis, and transformation with jUCMNav." *Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*. IEEE, 2009.

[19] Support for Microsoft Visio http://support.microsoft.com/ph/937.

[20] Escalona, M. J., et al. "The treatment of navigation in web engineering."*Advances in Engineering Software* 38.4 (2007): 267-282.

[21] Tomayess Issa , Andrew Turk , Applying Usability and HCI Principles in Developing Marketing Websites, IJCISIM, VOL-4, 2012, pp. 76–82

[22] Zhendong Ma, ChristianWagner, RobertWoitsch, Florian Skopik, and Thomas Bleier, Model-driven Security: from Theory to Application, IJCISIM, Vol-5, 2013 pp. 151 – 158

[23] Punam Bedi and Sumit Kumar Agarwal, Aspect-oriented Trust Based Mobile Recommender System, IJCISIM, Vol-5, 2013,pp. 354 – 36.

Author Biographies

Shailey Chawla Shailey Chawla is a research scholar at University of Delhi, India. She is pursuing PhD in computer Science on the topic of Web requirements

Engineering. Her other research areas include web content mining, business intelligence and usability analysis.

Sangeeta Srivastava She is an Associate Professor at Bhaskaracharya College of Applied Sciences. She is a Doctorate in Computer Engineering and has also done M. Tech. Her research areas include Method Engineering, Requirements Engineering and software engineering..

Deepak Malhotra Deepak Malhotra is working in Mazars, India. His area of work includes quality analysis and audits for software applications. He is proficient in many programming languages and has research interest in Quality in Software applications.