Goal-Based Requirements Analysis

Annie I. Antón
(anton@cc.gatech.edu)
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0280, USA

Abstract

Goals are a logical mechanism for identifying, organizing and justifying software requirements. Strategies are needed for the initial identification and construction of goals. In this paper we discuss goals from the perspective of two themes: goal analysis and goal evolution. We begin with an overview of the goal-based method we have developed and summarize our experiences in applying our method to a relatively large example. We illustrate some of the issues that practitioners face when using a goal-based approach to specify the requirements for a system and close the paper with a discussion of needed future research on goal-based requirements analysis and evolution.

1 Introduction

Goal-based methods stress the need to characterize, categorize, decompose and structure goals as requirements, but usually fail to offer strategies to identify goals, taking it for granted that the goals have already been documented [1] [4] [12]. If goals have not been previously documented, how do we first identify them and know when they are all completely specified? This is the question that our on-going research addresses.

This paper discusses some aspects of the Goal-Based Requirements Analysis Method (GBRAM) that we have developed in the context of its use in a real study. The method is useful to identify, elaborate, refine and organize goals for requirements specification. We also highlight some of the issues that practitioners would need to consider based on our observations and experience.

In the remainder of this section we elaborate the issues that we will address later. Section 2 provides an overview of the concepts and process elements of our goal-based approach. In Section 3 we illustrate some of the choices that practitioners may face when using a goal-based approach. Finally, in Section 4 we discuss lessons learned and suggest some practical recommendations.

1.1 Evolution of This Research

This research was initially motivated by our study of scenario analysis [8]. Our experiences demonstrated that scenarios are useful for uncovering and elaborating requirements, and for answering questions that are not easily answered using other techniques [8]. We also identified the need to investigate goal-based heuristics that suggest what scenarios to construct, analyze and elaborate further. This led us to apply the goal decomposition and scenario analysis model in the context of Business Process Reengineering [1] demonstrating that both scenario and goal analysis are valuable for an effective analysis effort.

Subsequently, we used the goal refinement method of [4] to specify the requirements for a software requirements validation environment. Our aim was to sketch the structure of such a specification in terms of the goals to be automated and their operationalization into high-level actions to be performed by software, hardware and users. Using the goal refinement method of [4] to establish the requirements for a requirements validation system that uses the Inquiry Cycle [7] [8] as the fundamental methodology, we determined the main goals of the system only with difficulty and not as straightforwardly or methodically as the literature suggests (see Section 1.2). The refining of high level goals, without a description of the system to be built, is not obviously systematic and appears to require guess work and inventiveness.

The research discussed in this paper stems from the need for better heuristics and procedural guidance for initially identifying and constructing goals, especially those involving the relationship between goals and scenarios. Based on our experiences, we have identified a set of issues that practitioners should consider, and we present them here. Our ultimate goal is to provide prescriptive advice to the practitioner in the form of goal-identification heuristics and a set of recurring questions to guide the process. Further studies will enable us to develop this catalog.

1.2 Related Work

Although there has been relatively little attention paid to the process of acquiring goals for system development, we certainly are not the first to address the issue. To position our work, we will briefly survey some of the more relevant previous work.

0-8186-7252-8/96 $5.00 © 1996 IEEE
Proceedings of ICRE '96

Traditional systems analysis focuses on *what* features (i.e. activities and entities) a system will support. Goal-based approaches focus on *why* systems are constructed providing the motivation and rationale to justify software requirements. The notion of focusing on the *why* is not new, although organizing requirements around goals is (e.g. "Context analysis," introduced in [9], determines the reasons *why* a system is to be created and the *purpose* of the system according to different viewpoints). A viewpoint captures a particular responsibility performed by a participant at a particular stage in the development process. The ability to represent viewpoints and distinguish between them is especially useful during requirements analysis. Viewpoints allow conflicts to be surfaced early in the requirements process, rather than burying them, so that they may be resolved. Finkelstein et. al. [6] propose the use of viewpoints as both an organizing and a structuring principle in software development.

Yu's strategic dependency model [12] provides the rationale for networks of actors in which agents depend on each other to achieve goals, perform tasks and furnish resources. The strategic dependency model supports the process of suggesting, exploring and evaluating alternative solutions and helps identify what is at stake, for whom, and what impacts are likely if a dependency fails. In this paper we discuss two types of dependencies: goal- and agent-dependency.

1.3 Practical Issues

Goals are useful for organizing and justifying software requirements, but two practical questions need to be addressed: *How are goals identified?* and *What happens to the requirements when goals change?* In this paper we discuss goals from the perspective of these two themes: *goal analysis* and *goal evolution.*

Goal Analysis

Goal analysis is the process of exploring gathered documentation, ranging from information about the organization (i.e. enterprise goals) to system specific information (i.e. requirements), for the purpose of identifying, organizing and classifying goals. It is often assumed that software systems are constructed with some goal(s) or purpose in mind [3]. However, what happens when the goal or purpose is not clear? Goals are often not given, so where do they originate? Enterprise goals do not always reflect what actually takes place [1]. Thus, it is important to gather as much information as possible in order to obtain a broad understanding of the domain, organization, process and system. The starting point, however, is often vague, perhaps comprising only one or two pages about the desired system. Given different types of documentation, how should we analyze them in order to identify and construct goals? We have investigated several related techniques: scenario analysis, identification of goal obstacles and constraints, and goal operationalization. As we will demonstrate, these techniques are not only useful for analyzing goals but also for elaborating and refining them.

Goal Evolution

From the developer's viewpoint, we wish for goals and software requirements to remain as stable as possible. Although it is true that requirements are volatile and constantly changing, much iteration in the refinement process is due to the requirements simply being misunderstood and/or misinterpreted [3]. Each stakeholder has different, and sometimes conflicting, requirements and priorities. Often the strategies for conflict identification and resolution are inadequate.

In our experience, goals are characteristically more stable than the processes, organizational structures and operations of a system which continuously evolve [1]; this is why we emphasize them. Nevertheless, goals evolve gradually and informally depending on the changing needs, circumstances and goal priorities of stakeholders.

The concepts and process elements of our goal-based approach (GBRAM) are discussed in the following section.

2 Goal Analysis and Evolution

In this section, we establish a common ground for discussion by defining some key concepts and terminology and by providing a brief overview of our goal-based approach discussing its characteristic process elements.

2.1 Concepts and Terminology

We define the following concepts:

- *Goals* are high level objectives of the business, organization or system. They capture the reasons why a system is needed and guide decisions at various levels within the enterprise.
- A *requirement* specifies how a goal should be accomplished by a proposed system.
- *Operationalization* is the process of defining a goal with enough detail so that its subgoals have an operational definition.
- *Achievement goals* are objectives of some enterprise or system. For example, a university course registration system may need to satisfy the goal of enrolling students in courses before the first day of class each semester. The object of the goal is course registration, which is exactly what the stakeholders believe the purpose of the system to be.
- *Maintenance goals* are those goals that are satisfied while their target condition remains true. They tend to be operationalized as actions or constraints that prevent certain states from being reached.

- *Agents* are the entities or processes that seek to achieve goals within an organization or system based on the implicit responsibility that they must assume for the achievement of certain goals. For example, given an Electronic Meeting System (EMS), a meeting initiator is the agent responsible for calling, or initiating, a meeting.
- *Constraints* are requirements that must be met for goal completion. A constraint places a condition on the achievement of a goal.
- *Goal decomposition* is the process of subdividing a set of goals into a logical subgrouping so that system requirements can be more easily understood, defined and specified.
- *Scenarios* are behavioral descriptions of a system and its environment arising from restricted situations. They exemplify behaviors enabling hidden needs to be uncovered and are useful for evaluating design alternatives and validating designs.
- *Goal obstacles* are behaviors or other goals that prevent or block the achievement of a given goal. Abstracting and identifying goal obstacles allows one to consider the possible ways for goals to fail and anticipate exception cases.

2.2 Process Elements

Goal analysis concerns the exploration of documentation (for goal identification) followed by the organization and classification of goals. *Goal evolution* concerns the way goals change from the moment they are first identified to the moment they are operationalized in a system specification.

Goal Analysis

Goals may be extracted from various types of gathered information including process descriptions such as flow charts or Entity Relationship (ER) diagrams. It is useful to identify goals from process descriptions by searching for statements which seem to guide design decisions at various levels within a system or organization. When used as the exclusive source of information, process descriptions are insufficient for achieving thoroughness and completeness. Therefore, if possible, the practitioner should consider other possible sources such as transcripts of interviews with stakeholders. However, stakeholders tend to express their requirements in terms of operations and action rather than goals. Thus, searching for action words is a useful way to extract goals from stakeholder descriptions. For example, in a meeting scheduler, stakeholders may use action words such as 'schedule' and 'reserve' which give rise to goals such as: `Schedule Meeting` and `Reserve Room`.

In addition to goals, the agents, stakeholders and constraints must also be identified. How and when do we go about this? The most sensible approach is to identify the responsible agents as early as possible (when each goal is first identified) by determining what agents are ultimately responsible for the achievement or maintenance of a goal. For example, the goal `Schedule Meeting` is the responsibility of the Meeting Scheduler. Constraints are useful because they provide additional information regarding requirements that must be met in order for a given goal to be completed. How are such constraints identified? As a general rule, we identify constraints by searching for temporal connectives, such as *during*, *before* and *after*, or any variants thereof. Constraints may also be identified by looking for dependency relations. Consider the goal `Meeting Arranged` in the meeting scheduler system with the constraint: `Meeting room must be available during the meeting date/time.` Once the goals, agent responsibilities and stakeholders are identified and specified, the goals are then classified according to their target conditions and begin to evolve.

Goal Evolution

Goals evolve because stakeholders change their minds and refine and operationalize the goals into behavioral requirements. Over the course of time, a stakeholder's goals may change or, at a minimum, their goal priorities are likely to change. The first type of change is elaboration and the second is refinement.

Goal evolution is thus effected via goal elaboration and refinement. Useful techniques for goal elaboration are: identifying goal obstacles, analyzing scenarios [8] and constraints, and operationalizing goals. Identifying goal obstacles, in order to consider the possible ways for goals to fail, enables one to anticipate exception cases. When goal priorities change, scenarios facilitate the evaluation of these new priorities. Goals are further elaborated by considering the possible ways in which goals can be blocked and by identifying scenarios to develop an understanding of how the goals can be operationalized.

Goal refinement occurs when synonymous goals are reconciled, when goals are merged into a subgoal categorization, when constraints are identified, and when goals are operationalized. In the GBRAM, achievement goals are merged and listed according to their precedence relations and dependencies. This ordering enables us to determine a goal's preconditions and postconditions. It is beneficial to consider goal precedence relations such as *"Which goals need to be fulfilled first?"* Our approach differs from Yu's model [12] in that dependency relations are used primarily to order goals so they can be subsequently refined. We expect that further consideration of goal and agent dependency relations will yield deeper insights for conflict resolution but as yet we have not addressed this.

Goals are refined by eliminating redundancies and reconciling synonymous goals. For example, the goals `Meeting arranged` and `Meeting scheduled` are synonymous and can be reconciled. In our experience, the best approach is to eliminate redundancies after the goals (extracted from each source) have been merged into one ordered goal set. It is then easier to identify synonymous goals because they typically are listed adjacent to each other (or clustered) in the ordered set since they tend to share common precedence relations. Goals are also refined via elaboration. The operationalized goals, responsible agents, stakeholders, constraints and scenarios are ultimately consolidated into a set of goal schemas that can be easily translated into a requirements specification. The resulting artifact, while not formal in the strict sense, provides a textual representation of system requirements organized according to system goals.

3 Experience with the GBRAM

We used the GBRAM to analyze the goals for a Career Track Training System (CTTS). The CTTS was part of a business reengineering project for an Air Force Base (AFB) in which no analysis was conducted prior to our investigation. We observed the process to be extremely fragmented, taking what appeared to be an enormous amount of time and effort, involving dozens of people across numerous organizational boundaries. Approximately 155 AFB employees are enrolled in training each year. The requirements for the process and the desired system illustrate typical information system requirements problems. For example, the need to register persons for training sessions is analogous to a university course registration system. In this section, we summarize our experience using the method described above and employ CTTS examples to illustrate the issues that we found important to consider.

3.1 Goal Analysis

The issues that one faces during goal analysis are illustrated in this section.

Identifying goals

By analyzing the three sources of information available to us (a textual introductory statement, a textual process scenario and flow charts of the existing process), we initially identified 36 goals. Some of these goals are shown in Tables 1 and 2.

Maintenance Goals	Agent	Stakeholders
G_1: Career tracks provided	AFB	AFB
G_2: Tax payers money spent efficiently	AFB	AFB, DoD, empl
G_3: Training coordinated	AFB	AFB, empl

Table 1: Maintenance Goals, Agents and Stakeholders

The introductory statement and process scenario, together, comprised a one page textual description. The introductory statement described the high level mandates that drive the system and organization. The process scenario described the employee certification acquisition process. The flow charts represented the current process as described during interviews with stakeholders. The medium, or language, in which the information was expressed is less important than the actual information provided. The introductory statement was the most declarative of the three sources while the process scenario provided a step by step description of the process (sharing more commonality with the flow charts). To identify goals, each statement is analyzed by asking, *"What goal(s) does this statement exemplify?"* and/or *"What goal(s) does this statement block or obstruct?"* The identified goals are worded to emphasize the state that is true, or the condition that holds true, when the goal is achieved. Wording goals this way makes it easier to reason about any preconditions and postconditions on the goals and any corresponding system operations.

Classifying maintenance and achievement goals

Tables 1 and 2, respectively, show a sample of the maintenance and achievement goals that we identified for the CTTS (Table 2, unlike Table 1, shows the information that needs to be recorded to operationalize the goals; i.e. obstacles and scenarios). Achievement goals were identified and classified by asking *"Is completion of this goal self-contained?"*, *"Does this goal denote a state that has been achieved or a desired state?"* and *"Does achievement of this goal depend on the completion of another goal?"* Goals G_1, G_2 and G_3, in Table 1, were extracted from the CTTS Introduction, and classified as maintenance goals by asking *"Does this goal ensure that some condition is held true throughout all other goal operationalizations?"* and *"Is continuous achievement of this goal required?"* For example, G_2 in Table 1 (`Tax payers money spent efficiently`) must be achieved on a 'continuous' basis. The AFB business process mandates that career tracks be provided in order to ensure that tax payers' money be spent efficiently. This goal characterizes a condition which must be held true. Maintenance goals are also identified by searching for keywords, such as 'provide' and 'supply,' that suggest a continual state within the system.

Maintenance goals were more likely to appear in organizational and policy level descriptions than in detailed descriptions of current processes. Interestingly, no maintenance goals were identified from the current process descriptions. In contrast, 9 maintenance goals were identified from the CTTS textual description. This supports the findings reported in [1] that organizational goals are often not reflected in operational strategies.

Identifying agents and stakeholders

Agents are identified to determine which agent is responsible for ensuring the achievement of a goal at any given time. Stakeholders are identified to determine which agents claim a stake in each goal and develop an understanding of the different viewpoints involved in the system for conflict resolution. More than one

Achievement Goals	Agent	Stakeholders	Goal Obstacles	Scenarios
G_1: Career portfolio created	empl	empl, supervisor	1. Career portfolio not created	
G_7 : Avalable course slots announced	TSD	empl, TSD, HRD	1. Avail course slots not announced 2. No slots available 3. Empl course prefs not available	2.a All courses closed (max capacity) 2.b Course cancelled (no slots available)
G_8 : Courses which empl qualifies for identified	system	empl	1. Empl qualifies for no courses	
G_{12}: Course completed	empl	empl	1. Course not completed	1.a Empl drop out of course 1.b Empl never enrolls in course 1.c Employee fails course
G_{13}: Skills improved	empl	AFB, empl	1. Skills not improved	1.a Course not taken 1.b Course not completed 1.c Empl fails course
G_{17} : Approval granted	AFB	AFB, empl	1. Approval not granted	
G_{18}: Certification granted	AFB	AFB, empl	1. Certification not granted	1.a Course taken doesn't qualify empl

Table 2: CTTS Achievement Goals, Agents, Stakeholders, Obstacles & Scenarios

stakeholder can be associated with a goal. In G_{13} (Table 2) both the employee and AFB have an interest, or stake, in improving employee skill levels. However, consider G_1 (Table 1), the AFB claims the sole stake in providing career tracks for all their employees (`Career tracks provided`) since each employee is ultimately interested in their own *individual* training acquisition status and skill level. G_{18} in Table 2 (`Certification granted`) is of interest to both the AFB and each employee seeking certification. Both agents claim a mutual stake in this goal. However, the AFB, not the employee, is ultimately *responsible* for granting certification to the employee. This highlights the potential for a goal to prevent another agent from achieving a goal. For example, if G_{17} (`Approval granted`) is not achieved then an employee is unable to achieve G_{18} (`Certification granted`) which may, in turn, prevent an employee from enrolling in any subsequent courses.

3.2 Goal Evolution

Evolution of the CTTS goal set was marked by modifications and additions to the goals themselves. We now discuss how the CTTS goal set evolved after our initial goal analysis.

Reducing the size of the goal set and clarifying goals

We uncovered three approaches for reducing the size of the goal set: eliminating duplicate goals, refining goals based on system entities, and consolidating nearly synonymous goals.

Goals based on system entities should be refined. Initially, we identified the goal: `HRD process completed` which clearly called for refinement due to the need to develop an understanding of what the 'HRD process' entails. By asking the stakeholder *"What is 'HRD process' and why is it significant?"* we were able to define `HRD process` and determine that the goal (`HRD process completed`) can be decomposed into two goals: `Available course slots announced` (G_7 in Table 2) and `Qualified personnel identified`. Thus, in this specific case, we eliminated one goal and replaced it with two new goals.

Goals can be consolidated by reconciling synonymous goals. For example, we initially identified two similar goals: `Skills improved` and `Qualifications improved`. The goal `Qualifications improved` was 'absorbed' by the goal `Skills improved` (G_{13} in Table 2). The CTTS goal set evolution was affected by using such refinement techniques. The number of CTTS achievement goals was reduced from 27 to 19.

Using constraints

In the CTTS, several statements describe requirements that must be met for a goal to be achieved. For example, consider Constraint #1 in Table 3 (`Course must qualify employee to advance to another level`).

Constraints
C_1 : Course must qualify employee to advance to another level
C_2 : Certification enables empl to move up to higher certification level

Table 3: Constraints from CTTS Process Scenario

How was this constraint identified? The word 'qualify' is a key indicator that a requirement must be met. Before an employee can advance their certification level, they must take a course which officially qualifies them for advancement. Thus, the system must 'know' which courses an employee can take. Note that goal G_8 (`Courses which employee qualifies for identified`) was identified from this constraint and the responsible agent is actually the system.

Elaborating and refining goals

Goal dependency relations exist between pairs of goals. A *precedence* relation between goals G_1 and G_2, where goal G_1 must be completed before goal G_2, is expressed $G_1 < G_2$. Precedence relations are identified for each goal by asking *"What goals are prerequisites for this goal?"* and *"What goal(s) must follow this goal?"* Another useful method for determining precedence relations is to search for agent dependencies. For example, consider a payroll system where employees are paid by the hour. Before an employee can be paid, the employee's supervisor depends on the employee to provide him the necessary information (e.g. time sheet). In this case, an agent dependency exists between the employee and the supervisor. Our main interest in organizing achievement goals according to their precedence relations is to enable us to envisage goal operationalizations and consider possible elaborations and refinements.

In the CTTS we considered the prerequisites for each goal. For example, before a career portfolio can be reviewed, or updated, it must be created. Table 4 shows the ordering that resulted from our analysis of G_2 (`Career portfolio reviewed`). By asking *"What are the prerequisites for this goal?"* we determine that $G_1 < G_2$. We also considered agent dependencies: the supervisor depends on the employee to provide him the necessary information to decide whether or not to approve the employee's career portfolio course preferences so the IP can be made available. Thus in Table 4, $G_3 < G_4 < G_5$.

Achievement Goals	Agent	Stakeholders
G_2: Career portfolio reviewed	empl	empl, supervisor
G_3: Career portfolio updated	empl	empl, supervisor
G_4: Career portfolio made avail	empl	empl, DTM
G_5: Empl course prefs ready	empl	empl
G_6: IPs made avail	HRD	HRD, empl

Table 4: Ordered Achievement Goals

Identifying goal obstacles

The possible ways in which goals can by blocked are considered by assigning a trivial obstacle to each goal. This is helpful because it forces us to consider specific cases that must be handled due to activities which prevent goal completion. Additionally, for each CTTS goal, we asked *"Can the failure of another goal to complete cause this goal to be blocked?"* and identified two obstacles for G_{19} (Table 5). Other obstacles can be identified by asking *"What other goal(s) or condition does this goal depend on?"* Achievement of G_4 (`Career portfolio made available`), in Table 5, depends on whether or not the supervisor's concurrence is obtained. Thus, using the "state as true" naming convention, `Supervisor's concurrence not obtained` gives rise to the obstacle `Career portfolio not made available`. Since `Supervisor's concurrence not obtained` is an actual instance of this obstacle, it becomes a *scenario.* Herein lies the difference between obstacles and scenarios. Obstacles denote the *reason* why a goal failed. Scenarios denote concrete *circumstances* under which a goal may fail. Scenarios can, thus, be considered instantiations of goal obstacles.

Identifying scenarios

Obstacle identification begins to identify ways in which goals can fail. This information is elaborated further via scenarios. Consider G_7 (`Available course slots announced`) in Table 5: the trivial obstacle is `Available course slots not announced`. We analyze this obstacle by asking *"Why did this obstacle occur?"* and determine that one possible reason is that no slots are available (G_7 Obstacle # 2). Given the obstacle `No slots available` we investigate possible circumstances leading to this being the case by asking *"What are the circumstances under which this obstacle can occur?"* and identify two scenarios for Obstacle #2: `2.a All courses closed (max capacity)` and `2.b Course cancelled (no slots available)`.

Scenarios help uncover *hidden* goals, issues or goal obstacles. Consider G_{15} in Table 5: there are several possible circumstances leading to the occurrence of the goal obstacle `Submitted paperwork not reviewed` (`Paperwork submitted late`, `Paperwork not complete`, and `Paperwork not received`). By identifying these scenarios, we uncovered potential issues which may have otherwise gone overlooked. For example, if the paperwork is not received (or is not complete), what action should be taken? Should anyone be notified? Scenario analysis is also useful for determining the postconditions of different behaviors and goals. For example, by asking *"What happens if this goal isn't achieved?"* we can identify the different possible postconditions for each goal. Consider the goal `Course & personnel matched`, if this goal is not achieved, employees will be unable to take a course they specified in their preference lists and consequently cannot improve their skills.

Operationalizing Goals

Goal information must ultimately be translated into a requirements specification. This is done by consolidating the goal information into a set of goal schemas. Although a wide variety of representations are available [5] [10] [11], we use an informal model-based style based on the FUSION method [2]. Goal schemas specify the relationships between goals and agents in terms of events that cause a change of state. Goals and their operationalizations (actions) are specified as events in terms of preconditions and postconditions. A precondition must exist for the achievement of a goal to be possible. For example, before the goal `Certification granted` can be achieved, an employee must first complete a qualifying course (`Course completed`). The postcondition characterizes the state of the system once the goal is completed. Below we show how to specify goals using a goal schema model. Figure 1 shows the goal schema model for G_7 (`Available course slots announced`).

The two subgoals for G_7 in Figure 1 were identified by considering two maintenance goals: `Position`

Achievement Goals	Goal Obstacles	Scenarios
G_4 : Career portfolio made available	1. Career portfolio not made avail	1. Supervisor's concurrence not obtained
G_7 : Available course slots announced	1. Avail course slots not announced 2. No slots available 3. Empl course prefs not reviewed	2.a All courses closed (max capacity) 2.b Course cancelled (no slots available)
G_{15} : Submitted paperwork reviewed	1. Submitted paperwork not reviewed	1.a Paperwork submitted late 1.b Paperwork not complete 1.c Paperwork not received
G_{19} : Certification status improved	1. Certification status not improved 2. Certification not granted	1.a Certification status not updated 1.b Course not completed 1.c Empl fails course

Table 5: Goal Obstacles and Scenarios

Goal:	`Available course slots announced`
Type:	`Achievement`
Description:	HRD must announce the courses and the number of slots available so that qualified personnel can be identified, matched and notified.
Action:	`Announce course slots`
Agent:	`HRD`
Stakeholders:	`HRD, employee, TSD`
Obstacles:	`1. No slots available` `2. Employee prefs not available`
Scenarios:	`1.1 All courses closed (max capacity)` `1.2 Courses cancelled (no slots avail)`
Preconditions:	`1. Employee prefs ready` `2. IPs made available`
Postconditions:	`Employee and course slot matched`
Subgoals:	`1. Qualifying training slots announced` `2. Position training slots announced`

Figure 1: Goal Schema Model for G_7

training provided and **Qualifying training provided**. These goals specify that there are actually two different types of training courses offered to AFB employees: qualifying training and position training. Analysis of these two maintenance goals, coupled with the achievement goal G_7, led to the construction of two new achievement goals listed as subgoals in Figure 1. The action associated with goal G_7 is defined as follows:

```
action AnnounceCourseSlots
   agent:  HRD
   reads:  course.slots
   changes:
   assumes:course.slots available
           employee.prefs ready
           employee.IP made available
   result: course.slots announced
end AnnounceCourseSlots
```

In above specification, **action** is the action which should be taken to achieve goal G_7. To announce the available course slots, the environment must supply **course.slots**. Since this particular action merely involves the broadcasting of information (e.g. the available course slots) **course.slots** is not modified by the action. The **assumes** clause shows what the environment is responsible for ensuring before the action (**AnnounceCourseSlots**) occurs. We now discuss the lessons learned and provide practical recommendations.

4 Lessons Learned

The only valid test of a practical method is in its use on real projects. While the work reported here does not yet provide that level of validation, it does provide some insights. In this section we discuss our ideas about goal-based requirements analysis and evolution in the light of work reported above and discuss our plans for future research.

In order to examine the evolution of the CTTS goal set, we tracked the size of the goal set during each stage of our analysis. Figure 2 shows the results of our CTTS goal analysis. The ovals show the number of identified instances of achievement and maintenance goals, constraints, goal obstacles and scenarios. The numbers depict the evolution of the initial goals to a more refined and elaborated set of goals. Given 36 lines of text (470 words) in the CTTS textual description (Introduction and Process Scenario), we identified 21 goals. After our initial goal analysis, the size of the goal set had been reduced to 18. The flow chart goals were refined resulting in a reduction from 14 to 9. Once the duplicate goals were eliminated and the synonymous goals reconciled, the set of achievement goals was reduced in size again from 27 to 18. Finally, as a result of refinements during operationalization, the size of the achievement goal set actually increased by 1 to 19. Our analysis of the evolution of the CTTS goals yielded several 'lessons' which we now discuss.

Multiple sources yield better goals

The goals for a desired system are not always clear at the outset. They must be extracted from diverse sources of information. It is unreasonable to expect to produce a complete goal set for a system from only one information source, but the combination of goals extracted from various information sources does produce a *more* complete set of goals especially if they incorporate the analysis of both the current and desired system. Maintenance goals are more likely to arise when exploring prescriptive organizational/policy level de-

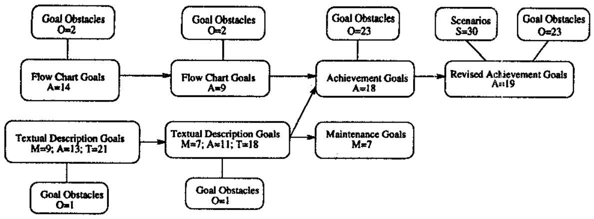

Figure 2: Evolution of CTTS Goal Set

scriptions of a desired system which tend to delineate the objectives of an organization. Consider, for example, that we extracted 9 maintenance goals from the CTTS prescriptive description and none from the process description. Achievement goals are more likely to arise when exploring process descriptions, which tend to contain more action words (verbs) than prescriptive descriptions. In our experience, it is easier to extract achievement goals from flow charts than from, for example, interview transcripts. Process descriptions in flow charts tend to be much more clear and succinct than a transcript of someone speaking with no organizational structure to minimize tangential comments.

Stakeholders provide insight into viewpoints

GBRAM allows analysts to share knowledge about the different stakeholders, their viewpoints and their responsibilities so that inconsistencies can be detected. Stakeholder identification allows us to consider when agents stand to gain or lose by the completion or prevention of each goal thereby facilitating the identification of conflicting viewpoints. However, stakeholders rarely give consideration to the actual goals of a system without some form of prompting. They are more likely to focus directly on operationalizations and actions they rely on a system to perform. Stakeholders tend to state goals descriptively [1].

Diversity of goal information gives a rich picture

System requirements information tends to be much more specific and more implementation-dependent than domain theory information. Domain theory information may appear more useful for identifying the high-level goals since the main concern is the overall problem not a proposed solution. But, more concrete goals and goal obstacles can be identified from system requirements (which primarily focus on actual functionality and system performance), ultimately leading to a better understanding of the system due to the availability of more concrete descriptions allowing us to more clearly envisage the desired system. Consider, for example, the different types of goals identified in the CTTS: maintenance goals extracted from policy level descriptions and achievement goals extracted from these same descriptions and process descriptions. When operationalizing an achievement goal, it is helpful to analyze the goals extracted from a different source since they enable us to clarify our understanding of the goal we seek to operationalize in order to specify it in more detail.

Categorizing goals suggests operationalizations

It is useful to differentiate among types of goals according to the target conditions of the goals. Given that high-level maintenance goals are often not reflected in operational strategies, it is useful to differentiate them from achievement goals. An efficient enterprise and use of information technology/systems depends on a close correspondence between the supporting systems and the enterprise goals and objectives. Maintenance goals tend to represent organizational goals while achievement goals tend to represent operational strategies. A close correspondence between both is essential.

As expected before our study, achievement goals best map to actions that occur in the system while maintenance goals tend to be nonfunctional (actions or constraints that prevent things from occurring). Maintenance goals are helpful when operationalizing achievement goals because they can point previously overlooked goals. In the CTTS, our analysis of maintenance goals resulted in refinements to the achievement goals since the availability of new relevant information at times provided additional descriptions of the goals yielding more meaningful operationalizations. By categorizing goals, we can also begin to bridge the communication gap between stakeholders and developers.

Constraints indicate requirements and point to new goals

Constraints provide additional information about requirements that must be met for a given goal to be completed and they provide insight into issues that must be considered when goal priorities change. For example, a constraint may indicate *when* a goal can be completed. Consider a constraint which specifies that a meeting must be scheduled on a specific day. If no room is available or no one can attend the meeting on that day, then clearly goal priorities must be reexamined. In the CTTS we identified new goals that we otherwise would have overlooked by examining the constraints. If an employee must take a course which qualifies him to advance to another level, then there must be a mechanism to identify the courses which the employee is qualified to take (see G_8 in Table 2).

Exceptions can be anticipated via goal obstacles

Goal obstacles are an effective mechanism to identify the possible ways for goals to fail and anticipate exception cases that must be handled by system operations. In the CTTS, obstacle analysis forced us to consider the reasons that could prevent an agent from achieving a goal. For example, consider G_{19} in Table 5 (`Certification status improved`). The two goal obstacles (`1. Certification status not improved` and `2. Certification not granted`) indicate the need to be prepared to handle such possible obstacles. If the certification is not improved, is it because the paperwork is tardy? If so, what goals must follow? If an employee fails a course, is he eligible to re-enroll in that course? Does he lose his ability to improve his certification status? Another benefit of obstacle analysis is the ability to more easily identify scenarios to determine why a goal could be blocked and when the relationship between goals and scenarios deserves further research since obstacles indicate which scenarios, if elaborated, would ensure coverage of exception cases.

Scenarios play a major role in uncovering issues

Scenarios offer a natural and concrete way to describe the circumstances in which a goal may fail or be blocked. They let us discover new goals and consider alternative mappings from goals to operations. In the CTTS, the use of scenarios allowed us to uncover hidden goals and obstacles and identify several circumstances leading to the occurrence a goal obstacle. By considering the obstacle `Submitted paperwork not reviewed` we uncovered issues that may otherwise have gone overlooked. For example, if paperwork is not complete, the employee must be notified and asked to resubmit their paperwork or risk losing their ability to improve their certification status. When operationalizing goals, it is important to identify postconditions for each goal. Analyzing scenarios in the CTTS enabled us to identify the different *possible* postconditions of various behaviors and goals. Our experiences demonstrate that scenarios are useful for uncovering and elaborating requirements, checking for completeness and conflicts, and communicating with stakeholders.

Future Work

Our findings have prompted us to strengthen our understanding of the relationship between goals, obstacles and scenarios. We seek to further develop and validate our strategies to identify and construct goals. The stability of goals is also interesting. If we can verify that goals are more stable and manageable than other types of system information, then goals would clearly be an excellent target for initial requirements acquisition. Our future work will concentrate in part on these elements of our goal-based approach. The results will influence our plans for building tools to support this process. We are currently prototyping tools that will allow us to support the process of identifying and capturing goals, responsible agents, stakeholders, constraints, goal obstacles and scenarios. To broaden our practical experience with the GBRAM, we are applying it to electronic commerce applications. In these studies we are seeking to answer how goals are used to identify and refine system requirements, and how the method's strategies are used in reengineering efforts involving a team of analysts.

Acknowledgements

The author would like to thank NTT for supporting this research, Darcy Painter for providing the CTTS case study, as well as, Peter Freeman, Mike McCracken, Colin Potts, Kenji Takahashi and Allison Elliott for their discussions concerning drafts of this paper.

References

[1] Antón, A.I., W.M. McCracken & C. Potts. "Goal Decomposition and Scenario Analysis in Business Process Reengineering," *Advanced Information System Engineering: 6th Int. Conf., CAiSE '94 Proc.*, Utrecht, The Netherlands, 6-10 June 1994, pp. 94-104, 1994.

[2] Coleman, D., P. Arnold, S. Bodoff, C. Dollin, H. Gilchrist, F. Hayes & P. Jeremaes, *Object-oriented Development : The Fusion Method*, Prentice Hall, 1994.

[3] A. M. Davis, *Software Requirements: Objects, Functions, & States*, Prentice-Hall, 1993.

[4] Dardenne, A., A. van Lamsweerde & S. Fickas, "Goal-directed Requirements Acquisition," *Science of Computer Programming*, 20(1-2), pp. 3-50, April 1993.

[5] Jacobson, I. *Object-Oriented Software Engineering: A Use-Case Driven Approach*, Addison-Wesley, 1992.

[6] Nuseibeh, B., J. Kramer & A. Finkelstein. "A Framework for Expressing the Relationships Between Multiple Views in Requirements Specifications," *IEEE TSE*, 20(10), Oct 1994, pp. 760-773.

[7] Potts, C. and K. Takahashi, "An Active Hypertext Model for System Requirements," *Proc. 7th Int. Workshop on Software Specification and Design*, Redondo Beach, CA, 6-7 Dec 1993, pp. 62-68.

[8] Potts, C., K. Takahashi and A.I. Antón, "Inquiry-Based Requirements Analysis," *IEEE Software*, 11(2), pp. 21-32, March 1994.

[9] Ross, D.T. and K.E. Schoman Jr., "Structured Analysis for Requirements Definition," *IEEE Transactions on Software Engineering*, 3(1), pp. 6-15, January 1977.

[10] Rubin, K. S. & A. Goldberg. "Object Behavior Analysis," *Communications of the ACM*, 35(9), Sept 1992, pp. 48-62.

[11] Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy, & W. Lorensen. *Object-Oriented Modeling and Design*, Prentice-Hall, 1991.

[12] Yu, E. and J. Mylopoulos, "Using Goals, Rules and Methods to Support Reasoning in Business Process Reengineering," *Proc. 27th Hawaii Int. Conf. System Sciences*, Maui, Hawaii, January 4-7, Vol. IV, pp. 234-243, 1994.