On Goal-based Variability Acquisition and Analysis

Sotirios Liaskos Alexei Lapouchnian Yijun Yu Eric Yu John Mylopoulos
Department of Computer Science
University of Toronto
10 King's College Road,
M5S 3G4, ON, Canada,
{liaskos, alexei, yijun, eric, jm}@cs.toronto.edu

Abstract

We introduce a variability-intensive approach to goal decomposition that is tailored to support requirements identification for highly customizable software. The approach is based on the semantic characterization of OR-decompositions of goals. We first show that each high-level goal can be associated with a set of concerns, in response to which, alternative refinements of the goal can be introduced. A text corpus relevant to the domain of discourse can be used to derive such variability concerns that are specific to the problem. In parallel, contextual facts that can vary while a goal is being fulfilled are modeled. Then, a high-variability goal model is constructed aiming at responding to the predefined variability concerns completely, while contextual factors are used to test whether it addresses all realistic background circumstances. We apply our approach in a study from the geriatric health care domain.

1. Introduction

Software customization enjoys an increasing interest in many computer science research communities. The omnipresence of software to support every aspect of modern daily living, including user's personal and leisure activities, calls for a shift of the adaptation effort from the user to the software system. Furthermore, an increasing demand for technologies that are inclusive for challenged user groups, such us the elderly or people with physical or cognitive impairments, implies that software designs must be highly adaptable to people with a wide diversity of backgrounds, goals and abilities.

From a requirements engineering standpoint, these challenges have been understood as a need for personal requirements processes, that are sensitive to characteristics of individual users and circumstances ([28]). The ability of a software system to support all configurations that may be needed to address a variety of user and context cases is understood as a necessary part of a personal and contextual framework for software development. In parallel, research in requirements for software product lines has long been exploring ways by which one can define core assets capable of serving as the basis for cost-effective derivation of products for individual users ([5]). In both directions, adaptation and reuse, the identification of the variability that the required software needs to exhibit is a central goal.

However, variability in a solution must reflect the variability of the problem. Thus, the identification of the latter can be viewed as an early requirements engineering problem. By considering variability at the level of stakeholders in terms of their goals, characteristics and contexts before we actually start sketching a solution, we increase the chances that the resulting software system will feature the appropriate set of variation points. Such a set must include those variation points that are needed to accommodate most known ways by which the stakeholders fulfill their goals. Yet it should exclude variation points that do not serve a purpose, but instead increase the cost and complexity of the system ([21]).

Goal models ([7, 4]) have been found to be effective for this purpose. They provide a natural way to identify variability at the early requirements phase, by allowing the capture of alternative ways by which stakeholders achieve their goals ([22, 17, 20]). This is achieved by viewing OR-decompositions of goals as a means to accommodate variability in stakeholder intentions. However, a better understanding of the meaning and potential origin of OR-decompositions of goals would shed more light on how goal models can influence the effectiveness of the variability acquisition result. Further, non-intentional variability, that is, variability that is not a result of stakeholders' intentions (e.g. time, weather, stakeholders' capabilities), must play a role in the analysis. The interplay between intentional and non-intentional variables of a problem and the role of the latter in shaping and constraining the former, although it provides a promising way to understand adaptation at a requirements level, it has thus far remained unexamined.

In this paper, we take a deep look into the types of variability that can be encoded in goal models and, based on our findings, we propose a *variability-intensive* process for decomposing and analyzing goals. This process aims at attaining completeness in the variability acquisition results, while allowing their representation in a concise manner. In addition, it allows *reasoning about alternatives* while taking into account the circumstances that hold in the context of attaining a goal. We structure the presentation as follows. We first discuss how the problem of variability identification has been tackled so far in the literature (Section 2). In Section 3, we construct a general set of semantic types of alternative refinements of goals. In Section 4, we further show how sets of semantic types that are custom-fit to particular domains can be constructed. We then discuss how non-intentional variability can be modeled (Section 5) and introduce a goal decomposition process, in which variability types guide the OR-decomposition of goals, while non-intentional variables are posed as selection conditions for the alternatives (Section 6). We conclude in Section 7.

2. Variability in Requirements Engineering

The problem of identifying and representing variability has been extensively investigated in the context of domain analysis ([24]). An essential part of domain analysis is the *commonality and variability analysis*, whereby the common and varying elements of a domain are identified. Most domain analysis methods point to potential sources of knowledge that can lead to the identification of variability (e.g. [12]) including examination of existing systems, consultation with domain experts, or even processes such as specially structured meetings ([30]). The result is most commonly formulated as a feature model ([18, 6, 19]), which represents admissible combinations of user-visible characteristics of the system-to-be in a concise hierarchical manner.

However, in order to be capable of representing many and diverse types of variability in a single view ([2]), feature models have deliberately relaxed semantics ([6]), which in turn makes them inappropriate for representing the structure, the behavior or other characteristics of the required system. Thus, researchers have been proposing the integration of feature modeling with requirements-based variability discovery methods. In [14], for example, use cases are proposed as a main way to capture variability while existing feature models give analysts a hint on how variability of past (good) systems was organized. The possibility of capturing variability in use cases is also discussed in [15], where variation points are introduced within use case diagrams. Customizable SCR representations have also been introduced ([10]), whereas in [29], parametric requirements documents are proposed.

Nevertheless, the above approaches define variability in terms of varying characteristics of the system-to-be, and not in terms of the *causes* of these variations, i.e. the varying characteristics of the problem, the stakeholders and their needs. Goal models have been found to be a potential way to shift the focus of variability discovery from the solution to the problem. In goal models it is possible to represent variability in stakeholder goals, through the notion of *OR-decomposition* of goals. When a parent goal is OR-decomposed into subgoals, fulfillment of any of the latter implies fulfillment of the former. In *i** ([31]) this is further understood as alternative *means* (subgoals) by which a certain *end* (parent goal) can be met. In GBRAM, on the other hand, alternative sub-goals appear as alternative responses to questions that arise from the generic type of goals ([1]).

But in [27] and [26], Rolland et. al. show that OR-decompositions can have more specific semantics. A semi-structured formulation of the goal description, adopted from [23], allows the analyst to define a number of aspects with respect to which variation may be possible. For example, alternative times, locations or beneficiaries for the fulfillment of a parent goal may each lead to an alternative subgoal. The authors go on to propose automatic generation of OR-subgoals by exhausting all possible combinations of alternatives of such aspects and pruning the result though identifying dependencies between these aspects. The dependencies assume that the relevance of a variability aspect depends on values given in other aspects. However, most inapplicable alternatives seem to be filtered out through informal examination of the result. As reported in [26], even after extensive pruning, the discovery procedure may lead to OR-decompositions with as many as 40 subgoals.

To see how variability occurs when analyzing a problem through goals, let us use an example from the health care domain, where we actually applied the process we describe later in the paper. The context of the application is a geriatric assessment unit, where elderly patients with moderate to severe health issues are hospitalized for an amount of time. The objective of the system-to-be is the monitoring of the patients' movement on the bed and around the unit so that nurses and doctors both maintain awareness of the patients' health condition and are appropriately alerted when their services are required. The latter is particularly needed for the nurses, who must administer a care plan defined by the doctors, and respond to a variety of events, including cases in which the patient is in danger.

An example of an event that needs such immediate response is when a patient with hypotension is trying to get off the bed by herself. She will probably fall down immediately after she is up. Thus, when an attempt is being made, somebody needs to `Be Notified` that the patient tried to get off the bed, and then rush and prevent it. But there are many ways by which this process can vary. For sensing that the patient is trying to get up, for example, one option is to install specially designed sensors that trigger the event automatically. Another option is to have a camera in the patient's room and a screen at the nursing station allowing manual monitoring and firing of the alarm. When it comes to the notification, there are more alternatives on whom to notify (the nurse that is assigned to the patient or any nurse that is close?), how to notify (using rooms' speakers or devices wearable by the nurses?), where to notify (among the rooms of the unit?), how intensively (loudly) to notify, etc.

Thus, the goal `Be Notified` comes with a list of variability issues (who?, where? etc.), which need to be tackled through decomposing the goal. But on what grounds can one assume a complete list of issues and how can the goal be decomposed to address all of them, while avoiding a combinatorial explosion problem? Further, non-intentional variables, such as how severe the patient's condition is or where her nurse currently is and what he is doing, obviously play a significant role in selecting the appropriate alternative for each of the above variability issues. But how can one systematically introduce them in the goal model, and then use them to reason about alternatives that are applicable in given circumstances? The rest of the paper addresses these questions.

The diagrammatic formalism we will use to represent goal models is a subset of the one introduced in [13] and, at the same time, expressively equivalent to Propositional Calculus. An example of such a model is seen in Figure 1. The backbone of the model is an AND/OR decomposition tree. Goals (the ovals in the figure) can be either satisfied or not satisfied. When a goal g is AND-decomposed into $g_1, \ldots, g_n$ then g is satisfied iff g_i are satisfied for all i. If g is OR-decomposed, it is satisfied iff there exists an i such that g_i is satisfied. In addition, several types of links are used to represent constraints among goals. Thus, given two goals g_1 and g_2 the links $g_1 \xrightarrow{++} g_2$ and $g_1 \xrightarrow{--} g_2$ show that when g_1 is satisfied then g_2 is or is not satisfied respectively. The link $g_1 \xleftrightarrow{++} g_2$ (respectively, $g_1 \xleftrightarrow{--} g_2$) is equivalent to having both $g_1 \xrightarrow{++} g_2$ ($g_1 \xrightarrow{--} g_2$) and $g_2 \xrightarrow{++} g_1$ ($g_2 \xrightarrow{--} g_1$) at the same time. Further, the link $g_1 \xrightarrow{pre} g_2$ indicates that g_2 cannot be satisfied unless g_1 is satisfied. Finally, when there is a need to represent more elaborate constraints we can use rectangles in which we can construct condition formulae using more than one goal. From each of such condition entities c we can then draw $c \xrightarrow{++} g$, $c \xrightarrow{--} g$ or $c \xrightarrow{pre} g$ links to one or more goals g of the model.

We can associate each goal with a propositional literal and represent the satisfaction of the root goal in terms of a propositional formula $G \equiv S_g \wedge C_g$. S_g represents the AND/OR structure in terms of leaf level literals. Each non-

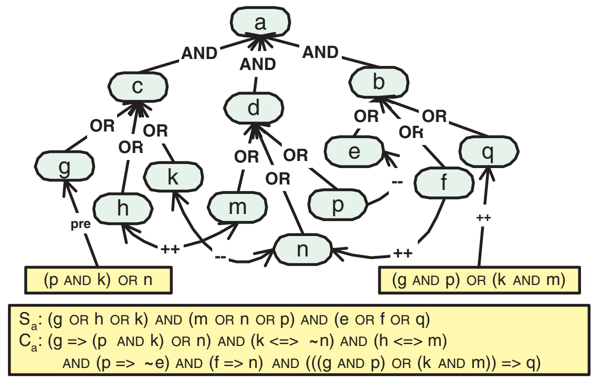

Figure 1. A goal model

leaf node is recursively replaced by the conjunction or disjunction of its children depending on whether the decomposition is AND or OR, respectively. C_g represents the additional constraint links. Each constraint link in the model results in a conjunct in the formula C_g as follows:

Link Type	Conjunct
$g_1 \xrightarrow{++} g_2$	$g_1 \Rightarrow g_2$
$g_1 \xrightarrow{--} g_2$	$g_1 \Rightarrow \neg g_2$
$g_1 \xrightarrow{pre} g_2$	$g_2 \Rightarrow g_1$
$g_1 \xleftrightarrow{++} g_2$	$g_2 \Leftrightarrow g_1$
$g_1 \xleftrightarrow{--} g_2$	$g_2 \Leftrightarrow \neg g_1$

In the first three cases, g_1 can be a condition formula instead of a single goal. In all cases, literals representing non-leaf nodes are replaced with clauses that contain only leaf nodes according to the AND/OR structure.

An alternative in a goal model such as the one in Figure 1 is a solution of the AND/OR tree, that is, a subgoal choice for each of the OR-nodes, that satisfies the constraints. Finding alternatives is understood as finding truth assignments that satisfy the resultant propositional formula G. Thus, in Figure 1, configuring the OR-decompositions to the children $\{g, n, f\}$ satisfies the root goal.

3. Variability Concerns for Goal Decomposition

Since goals express desired states of affairs, they are normally descriptions of something that needs to be true in the world, for example `Message is Sent` or `Light is ON`. However, semi-formal goal decomposition calls for a gradual shift of focus from the desired state of world that a goal describes, to the human or machine activities that can potentially satisfy the goal. Thus, in most of the cases, the above goals will be seen phrased as `Send a Message` or

Turn Light ON. In that spirit, a goal can be understood in terms of a generic activity, and its analysis as the process of specifying this activity better.

Consequently, when a high level goal is phrased, the generic activity it requires is necessarily vague and incomplete. For example consider the goal Send a Message. Who will Send a Message? To whom? When? Where? How fast? What message? Such questions describe concerns that call for alternative responses and, consequently, alternative refinements of the goal. A study of the possible types of such concerns can be greatly facilitated by looking at categorizations of semantic roles of sentence elements, as they are studied in Linguistics.

We use Fillmore's case system ([11]) as a basis for understanding language semantics in a requirements engineering context; though, here we focus on goals. According to Fillmore, a simple sentence consists of a verb and a set of noun phrases. Each noun phrase holds with the sentence a relationship of a particular semantic type. Linguistically, these types correspond to different *cases*. Fillmore proposes that there exists an essential set of such case types that fits in the case system of every known language. Each of these universal case types addresses a particular semantic concern associated with the verb of a sentence. Hence, they can be seen as a set of potential semantic slots that may or must be associated with each verb, and filled whenever the verb is used in a complete sentence. This way, given a verb, a *frame* feature can be defined, which is a set of such semantic slots (*frame elements*) that the verb *"evokes"*. For example the verb "open" is necessarily associated with an *objective* slot ("*what* opens/is opened?") but may also be associated with an *agentive* slot (to answer "*who* opens?") and an *instrumental* one (to answer "open *with what*?").

Considering the verb that describes the generic activity in a goal description, the discovery of alternative goal refinements can be driven by the frame that is associated with that verb and the corresponding elements. In this context, the frame elements can be seen as *variability concerns*, that is, types of questions whose alternative answers result in alternative refinements of the original goal. In the Send Message example, the agentive variability concern asks who sends the message. For each potential *response* to the variability concern (for example "the user", "the machine", "the administrator", "the user and her assistant together") an alternative refinement of the goal needs to be introduced. The collection of all concerns relevant to a goal is the *variability frame* evoked by the goal.

Based on Fillmore's idea of defining a universal set of frame elements, we can introduce a general set for variability concerns, to be used for the construction of variability frames for goals. The set we constructed includes most of the semantic types Fillmore proposes, but also draws information from adjunct classification schemes that are frequently discussed in grammar books (we used [25] and [16]) as well as from the way goals are formalized in the goal analysis literature. Thus:

Agentive (*A*) is the concern of the agent(s) whose activities will bring about the state of affairs implied by the goal description. Responses to the concern are typically actors or combinations of actors found in the domain, including the system(s)-to-be. For example $\{$*Machine, User alone, User Supported*$\}_A$ *to choose schedule*. Alternative responses to the agentive concern are essentially *alternative delegations* of a goal to actors (including the system-to-be).

Dative (*D*) is the concern of the agent(s) who will be affected by the generic activity implied by the goal. As above, responses to the concern are typically actors or combinations of actors found in the domain, including the system(s)-to-be. Examples are *Send a message to* $\{$*the admin, the user*$\}_D$, *Notify* $\{$*designated nurse, nurses at nursing station*$\}_D$

Objective (*O*) is the concern of the object(s) that is affected by the generic activity implied by the goal. Example are: *Send* $\{$*an e-mail message, a fax message*$\}_O$, *Print* $\{$*a full report, a summary*$\}_O$.

Factitive (*F*) is the concern of the object(s) or being(s) that is/are resulting from the activity or understood as part of the meaning of the verb. Examples: *Format Text* $\{$*bold, italic*$\}_F$ or *Turn light* $\{$*on, off*$\}_F$.

Process (*P*) is the concern that determines the instrument (*P.ins*) that is involved in the performance of the generic activity implied by the goal, as well as the means (*P.mea*) and the manner (*P.man*) by which the activity is performed. The subcategory *P.mea* is the concern to which "pure" *i** means-ends variability should be classified. Some examples are: *Pay* $\{$*by debit, by credit, by cash*$\}_{I.ins}$, *Meet new people* $\{$*by organizing activities, by participating in activities*$\}_{I.mea}$ or *Notify User* $\{$*loudly, subtly*$\}_{I.man}$.

Locational (*L*) is the concern about the spatial location(s) where the generic activity that is implied by the verb is supposed to take place. Example: *Send a message* $\{$*in the Car, on a Bus*$\}_L$.

Temporal (*T*) is the concern about the duration (*T.dur*) or frequency (*T.frq*) of the generic activity that is implied by the verb. For example: *Check for messages every* $\{$*hour, 10 mins*$\}_{T.frq}$, *Suspend Notifications for* $\{$*2 hours, 10 mins*$\}_{T.dur}$. Temporal location is dealt through the next concern.

Conditional (*C*) concerns refer to either alternative conditions under which the goal can be fulfilled (*C.con*) or alternative triggers of the generic activity associated with the goal (*C.tri*). For example: *Ship product only if* $\{$*order has arrived, payment has arrived*$\}_{C.con}$ or *Notify user* $\{$*when message arrives, in regular intervals*$\}_{C.tri}$.

Extent (*E*) variability concerns refer to alternative degrees by which the generic activity can be performed (excluding duration). For example: *Display the first* $\{$*10,20,10%*$\}_E$ *(of) records*.

The set is by no means a template for structuring goal phrases, as in [26, 23], but a catalogue of categories that can help analysts understand the variability aspects of goals.

4. Constructing Problem-specific Variability Frames

Using a general set of variability concerns to characterize variability for arbitrary goals may come with certain drawbacks. Firstly, the concerns are not guaranteed to be equally intuitive for every goal for which they are to be used, due to the necessary generality they must demonstrate. Secondly, in order to fit a great number of cases, they are necessarily coarse-grained and may ignore certain variability aspects that arise when examining individual goals in detail. Maintaining catalogues of variability frames that are specific to goals and the problem domains they appear in is a way to cope with these issues.

In the area of semantic frames, the construction of a frame lexicon has proved possible in FrameNet ([3]). FrameNet introduces a lexical database for English that provides the meaning of words in terms of the semantic frame they evoke. The lexicon contains a large set of semantic frames that are to be used for this purpose. Each frame contains its own set of elements that are specific to the frame's semantics. Hence, if we are given a frame associated with a goal, by simply interpreting the frame elements as variability concerns, we can construct a customized variability frame for a particular goal.

The association of a goal with a semantic frame is again based on the verb of the goal phrase, but other key words of the phrase may play a role as well. In practice, multiple frames may be considered for a single goal. Consider for example a health organization posing the goal `Be Aware of Patient's Condition`. By consulting FrameNet, one discovers that there are two frames that are related to it, namely *awareness* and *becoming_aware*. Both contain elements that are very useful in identifying interesting variability concerns: the *cognizer* is the person who wants to be aware (e.g. {doctors, nurses, family}), the *phenomenon* is the situation of which the cognizer(s) wants to become aware (e.g. {a complication, a delirium, a fall}), *evidence* refers to the particular observations that allow the cognizer to become aware (e.g. {patient not responding, patient wandering in the unit}), while *state* is the state of the phenomenon when the cognizer becomes aware of it (e.g. for the case of a patient's fall: {is about to fall, is currently falling, has fallen}). Although none of these concerns belongs to the universal set we discussed earlier, they arguably do a better job in describing fine-grained variability aspects of the goal originally stated by the stakeholders.

But how can such a frame lexicon be built? The construction of FrameNet is based on the examination of a large corpus of English texts. Once several sentences in which a particular word is found are collected, they are subjected to annotation, which is, roughly, a classification of the phrases that are surrounding the word into frame elements. Associating the word with a particular frame (which may include devising a new frame or specializing an existing one) also requires human involvement and intuition. Due to the inherent labor-intensiveness of the endeavour, FrameNet is still under construction (2006). However, its development to date shows that given a corpus of *attested* sentences, it is possible to construct a lexicon of semantic frames.

In the context of requirements analysis, frames that are particular to goals can be constructed the same way. The motivation for doing so is not only the potential incompleteness of general purpose lexicons such as FrameNet, but mainly the construction of frame lexicons that are more informed with respect to particular problem domains. Such specialized frame lexicons can be constructed by referring to documents that describe the processes of the domain of interest or by examining evidence from past projects, particularly artifacts of early elicitation efforts (e.g. interview transcripts, reports etc). The analysts can then identify sentences in which words of interest appear, and annotate the surrounding phrases appropriately. Thus, the resulting frames are based on a corpus that is specific to a domain scoped by the analyst herself.

Consider for example that we want to decompose the goal `Schedule a Graduate Meeting` in the context of a graduate educational program. Suppose also that we are particularly referring to the meetings that relate with a student's progress in the graduate program (progress meetings, checkpoints etc). In the department of Computer Science at the University of Toronto, we found three documents that describe how this process should be performed, namely the Graduate Handbook issued by the department, the Graduate Calendar issued by the University-wide umbrella organization of all graduate programs, called the School of Graduate Studies (SGS), as well as a text providing Graduate Supervision Guidelines again issued by SGS. This collection of texts, which amounts to a total of about 18,000 words, was then subjected to searches of words that relate to the goal in question. We considered the words *meet* and *examination* (the latter being an alternative way to refer to several formal meetings) and looked in the context in which they appeared. We found a total of 41 and 116 sentences where these words occurred in their various forms, respectively. By examining the phrases that accompanied the words in each sentence we were able to collect a number of standard semantic elements that define such a meeting.

For instance, in the sentence "The departmental $\{$thesis$\}_M$ examination is open to {all students and faculty members of the department$\}_N$", discloses two elements that vary in examination meetings and namely the material that is central to the meeting, denoted with M (here it is a thesis but can also be a progress report, a literature review, etc.) as well as the "openness" of the meeting in terms of who is allowed to attend it, denoted with N. By working this way with all sentences, we collected 21 variability concerns, some of which are: *Purpose, Language, Frequency, Formality, Duration, Participants, Openness, Temporal Location in Year, Temporal Location in Graduate Program Timeline, Agent Responsible for Organization, Agents to be Notified*, and *Meeting Material*. We call our frame *Graduate_Meeting*. Each of its elements has a special meaning that is specific to the domain. For example alternative responses to the concern *Participants* can be {Core Committee, Core Committee with Student, Extended Committee with Student} whereas the concerns *Agents to be Notified* may be {Nobody, The Grad Office, The SGS}.

Potential semantic elements that are absent can be derived from frames that appear in FrameNet through frame *inheritance*. A frame may inherit all elements of another frame and introduce its own elements. Moreover, certain semantic elements may be *elaborated*, i.e. made more specific. Thus, *Graduate_Meeting* inherits elements from FrameNet's *Congregating*, and hence have frame elements currently absent from the former, such as the *Place* of the graduate meeting, be drawn from the latter. Elaboration is performed by changing the name of an inherited frame element to a more specific one. Thus, the element *Individuals* that is part of *Congregating*, is renamed into *Participants* for *Graduate_Meeting*, still leaving space for further elaboration.

5. Background Variability

By the term *background variability* we refer to facts about the domain of discourse that unintentionally vary in the context where the fulfillment of a goal is attempted. For example, a user may want to `Send a Message` while being at a particular place, a particular time, doing something specific or being capable of doing certain things such as hearing or speaking in a particular language. Such facts are circumstances under which the goal will need to be achieved. Hence, they constitute factors that may influence both the identification of new alternatives when an OR-decomposition is attempted and their selection thereafter. Thus, the goal `Fire a Loud Audio Notification` may presume that there are no people sleeping around the agent to be notified (e.g. in a hospital at night). Conversely, knowledge of the possibility that there might be a case of an agent that needs to receive a notification while being around people who sleep, calls for the identification of additional alternatives that can bypass these constraints.

Our experience showed that background variability can be effectively identified by focusing on three basic entity types in the domain of discourse: *agents* (e.g. Nurse, System-to-Be, Unit Administrator), *locations* (understood here as a synonymous of "contexts", e.g. Graduate Lounge, Street, Nursing Station) or *objects* (e.g. the Line at the Bank, an Incoming Message, a Driver's License). Background variability is then formulated in terms of attributes of each of these agents, locations and objects or relations among them. Of course, the analyst will focus on these attributes and relations that may vary. More specifically:

Agent Characteristics, refer to varying properties of agents (including the system to be) such as their location, *isAtLocation(Agent, {home, office, bus})*, their skills/capabilites, *hasCapability(Agent, {hear, access_sensitive_data, access_the_internet})*, their current business, *isDoing(Agent,{driving, meeting})*.

Location Characteristics, are attributes of the location where a goal may need to be satisfied, for example the local time *isTime(Location,{morning, winter, january, friday_evening})*, the levels of noise, *hasNoize(Location, none, low, extreme)*, or the temperature, *hasMinTemperature(Location, {T>30, T<-10})*. Note that mobile locations (e.g. buses, cars) can be treated as stable ones with varying characteristics.

Object Characteristics, refer to objects of the domain and varying attributes thereof, or simple global facts and parameters that cannot be classified otherwise. Examples of this category are highly domain dependent: *messageSize(Message, {X>50kB, X>1MB})*, *customersInLine(Line, {C>4, C>20})*, *isAtLocation(Printer, {lab, supply room, computer room})* or *totalEnrolments(Course, {C>300, C<20})*.

The relations may refer to either short term circumstances of the agents, locations and objects they involve (e.g. current level of noise somewhere) or long term conditions (e.g. a user's physical challenge).

6. Variability-Intensive Decomposition and Analysis

We now present an example process for variability-intensive reduction of goals and a method for reasoning about the resulting high-variability model. The process is based on the identification of an initial set of variability concerns, followed by a one-concern-at-a-time goal decomposition approach in order to form the AND/OR tree. Meanwhile, background variables are set as selection conditions for each of the introduced goals allowing reasoning about the role of background circumstances in the selection of alternatives. We illustrate the process using the `Be Notified` example we introduced in Section 2.

Identification of relevant variability concerns and background variables. Before starting the decomposition of the goal `Be Notified` the analyst will consider variability concerns that are relevant to that goal. Further, for each relevant variability concern, an initial domain of options is identified. In the example (we are using the general set of concerns): an agentive concern is relevant as to who will generate and send the notification ($\{$human observer, a system$\}_A$), a dative concern poses the question of who is notified, a process concern calls for alternative notification modes ($\{$open-audio, headphone-audio, vibration$\}_{P.ins}$) and manners ($\{$intensive, normal, subtle$\}_{P.man}$), a locational concern asks where the notification is send ($\{$nursing station, meeting room, room1, ...$\}_L$), a factitive concern questions the content of the notification ($\{$distinctive sound, a voice message explaining situation$\}_F$), a conditional-trigger concern deals with alternative options on the condition that needs to be true for the notification to be fired ($\{$trying to get up, sitting on bed$\}_{C.tri}$). This collection of concerns constitutes the variability frame of the goal.

At the same time, interesting relations for representing background variability are considered. As discussed earlier, these represent characteristics of agents, locations, as well as miscellaneous facts or objects whose characteristics may vary. In our case, agents are different types of nurses in relation to the patient ({The Assigned Nurse, The Closest Nurse}) and the locations are the rooms of the unit ({Patient's Room, Meeting Room, Nursing Station}). Each nurse can be in a room. The nurse may be busy attending to a patient, doing paperwork, having a break, or she may simply be available. The time can be night, day, or afternoon-nap time (associating time with location is not useful here). Certain other facts may influence the identification/selection of alternatives such as the severity of the patient's condition, the degree of belief by which we assume she is trying to get off the bed, the number of nurses available in the unit etc.

Such background facts are identified with respect to a particular variability concern, otherwise they are irrelevant. Thus, the time of the day (day vs. night) is important for deciding the loudness of a notification, whereas the age of a nurse does not influence such a decision. Figure 2 shows some variability concerns associated with the root goal `Be Notified` as well as the related background facts.

Concern-driven decomposition. Once an initial set of relevant concerns and their domains are collected, decomposition of the goal can follow.

Initially, for the root goal, each concern is by default labeled *unresolved*. When a goal is AND-decomposed, every variability concern relevant to the parent goal is inherited by at least one of the AND-subgoals. Some concerns of the parent goal, however, may be irrelevant for

`Be Notified`
Agentive: (who will decide and fire the notification)
{'The System', 'The Nursing Station's Nurses'}
depends on: -
Dative: (who will receive the notification)
{'The Closest Nurse', 'The Assigned Nurse'}
depends on: -
Locational: (where must the notification be sent)
{'Nursing Station', 'Lounge', 'Meeting Room', 'PatRoom1', ...}
depends on: location of notification subject
Process – Manner: (how intense the notification will be)
{'Intense', 'Normal', 'Subtle'}
depends on: patients condition, time of day, process-instrument
Process – Instrumental: (what is the notification mode)
{'Open-Audio', 'Headphone-Audio', 'Vibration'}
depends on: time of day, locational.
Condition-Trigger: (which cases are sufficient for triggering a notification)
{'Patient is almost up', 'Patient tries to get up'}
depends on: patient's condition, presence of caregivers in the room.

Figure 2. Initial Variability Concerns

some sub-goals. Thus, if the `Be Notified` goal is AND-decomposed into `Sense Event and Trigger Notification` and `Receive Notification` the former inherits the agentive (*A*) and condition-trigger concerns (*C.tri*), while the latter inherits, among others, the dative (*D*) and process (*P*) concerns. All concerns are inherited with their label.

When a goal is OR-decomposed, exactly one variability concern relevant to the goal is addressed, while the rest are automatically inherited. A variability concern is addressed by partitioning its domain and assigning each partition to one of the OR-subgoals. If the partition contains only one domain element, then the concern can be labeled *resolved* at this subgoal and is not inherited in further decompositions of the goal. If not, the concern is labeled *addressed* and further inherited by subgoals. Figure 3 shows the process in detail. In Figure 4, the variability concerns that are relevant to the goal `Be Notified` appear in a rectangular annotation close to it.

Multi-faceted OR-Decomposition. The analyst can organize the order by which variability is addressed in the decomposition model in two ways: vertically or horizontally. The former suggests addressing each concern at a separate level of the decomposition sub-tree. For example, for each response for the agentive concern, a subgoal is introduced, each of which is decomposed with respect to the locational concern, having each of the resulting subgoals decomposed with respect to a process concern, and so on. This way of decomposing the goal results in a rather impractically large goal model, as each leaf tends to represent a unique combination of values of each variability concern.

In practice, however, variability concerns are orthogonal with respect to the sets of alternative options they introduce. Thus, the set of options for where a notification is to be heard, and the set of options for how loud the notification is heard, do not depend on each other (although a selection in one certainly influences the selection in the

CONCERN-DRIVEN DECOMPOSITION

. Let $V(g)$ be the set of variability concerns that are relevant to a goal g.
. Let $res(c,g)$ be the label of the goal g with respect to variability concern c; $res(c, g)$ takes values from the set { RESOLVED, ADDRESSED, UNRESOLVED }.
. Let $dom(c,g)$ be the domain of alternative values the concern c can take in goal g and $|dom(c,g)|$ its cardinality.

1. Let g_p be the goal to be decomposed.
2. If g_p is a root goal, set $res(c,g_p)$:= UNRESOLVED for all c in $V(g_p)$.
3. Decompose the goal into its subgoals $g_1, \ldots, g_n$ as follows:
 If g_p is AND-decomposed:
  . depending on the decomposition purpose, set $V(g_i)$ to be any subset of $V(g_p)$ – {concerns c such that $res(c,g_p)$ == RESOLVED } that is relevant to g_i. Every concern in $V(g_p)$ must appear in some $V(g_i)$. For all c in $V(g_i)$, set $res(c,g_i)$:= $res(c,g_p)$.
- If g_p is OR-decomposed in response to a variability concern v then:
  . for each g_i, set $V(g_i) = V(g_p)$ – {c such that $res(c,g_p)$ == RESOLVED }
  . $dom(v,g_p)$ is partitioned into n non-empty $dom(v,g_i)$, i=1... n.
  . for each g_i, **if** $|dom(v,g_i)|$ == 1 then set $res(v,g_i)$:= RESOLVED ,
    else $res(v,g_i)$:= ADDRESSED.
  . for each g_i, for every concern c in $V(g_i)$ for which c != v, set $res(c,g_i)$:= $res(c,g_p)$
4. Run the resolution assessment algorithm.
5. For each subgoal repeat the process from (1) until all variability concerns of the root are resolved.

Figure 3. Concern-driven Decomposition

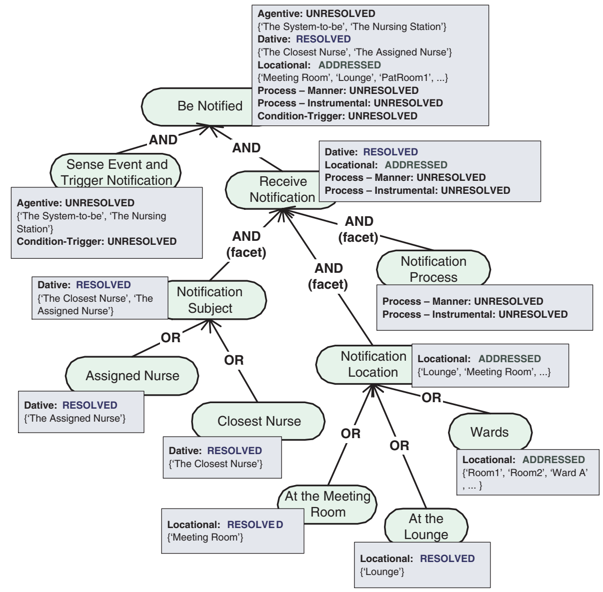

Figure 4. Decomposition for `Be Notified`

other). Horizontal organization of variability in goal models allows variability concerns that demonstrate such orthogonality, to be decomposed in parallel i.e. at the same level of the decomposition tree. This can be achieved by simply AND-decomposing the goal into subgoals for each of which only part of the variability concerns of the parent goal becomes relevant. Such an AND-decomposition deviates from its usual meaning (see for example [8] for an extensive discussion on AND-decomposition types) since it is part of what we would call a *multi-faceted* OR-decomposition, because it allows the representation of refinements of a goal from multiple points of view. In Figure 4, `Receive Notification` is analyzed through a multi-faceted OR-decomposition. The AND-decomposition is annotated appropriately, to show its special function.

As we will see below, selection dependencies among different facets will very likely exist and are treated by establishing lateral links between sub-goals, via e.g. formulating the selection conditions appropriately.

Variability resolution assessment. At any stage of the decomposition process, we can assess which of the concerns that were originally thought as relevant have been actually resolved through the decomposition. This can be done by propagating the concern resolution labels from the leaf level nodes towards the root.

The propagation algorithm can be seen in Figure 5. If a goal is either OR- or AND-decomposed, a variability concern related to it is labeled *resolved*, when the concern is labeled *resolved* in all the subgoals that inherit it. If there exists at least one subgoal where the concern is labeled *unresolved* or *addressed* then the parent goal is appropriately (see Figure 5) labeled *unresolved* or *addressed*. At the end of the process, each of the variability concerns of the root goal are labeled as *resolved, addressed* or *unresolved*. In Figure 4, next to each concern, one of the three labels (unresolved, addressed, resolved) is used to describe the resolution assessment at the current (early) stage of the decomposition.

RESOLUTION ASSESSMENT ALGORITHM

INPUT: A concern-driven goal decomposition tree.
OUTPUT: A set of labels indicating the state of variability resolution of the root goal.
1. Consider the set G of all intermediate goals whose children are all leafs.
2. For each such goal g in G, let $g_1, \ldots, g_n$ be its children and $V(g)$ the set of variability concerns relevant to g.
3. Let $res(c,g)$ be the label of a goal g with respect to concern c, with domain {RESOLVED, ADDRESSED, UNRESOLVED}.
4. For each concern c in $V(g)$, update its label as follows:
  if there exists a g_i such that $res(c,g_i)$ == UNRESOLVED then
    $res(c,g)$:= UNRESOLVED.
  else if there exists g_i such that $res(c,g_i)$ == ADDRESSED then
    $res(c,g)$:= ADDRESSED.
  else /* i.e. no UNRESOLVED or ADDRESSED labels amongst children */
    $res(c,g)$:= RESOLVED
5. Prune the leafs of the tree and repeat from (1) until the set G is set to empty.

Figure 5. Resolution Assessment

Background Facts as Selection Criteria. Apart from addressing variability concerns, predicates describing background circumstances are set as selection conditions for alternative OR-subgoals. The concern that is addressed by the decomposition will indicate the predicate(s) to be used as a selection condition, by consulting the respective dependencies identified in the beginning of the process (Figure 2). Using these predicates, the analyst can construct formulae that are set as preconditions for the selection of a sub-goal as part of an alternative solution of the AND/OR subtree.

In practice, we can construct useful selection conditions as propositional clauses in which background predicates are treated as propositions. Such clauses are then set as conditions, exactly as we discussed in Section 2. For example, consider the case of a decomposition where the *P.ins* concern is addressed for the goal `Notify Designated Nurse` and one of the OR-subgoals for the concern is `Receive Open Audio Notification`, i.e. an audio signal through some sort of a speaker. Whether this goal can be selected as part of an alternative, depends on the response to the locational concern *L*, as well as the time of the day. In our case, it is OK to send an audio notification through the room speaker, provided that this room is not the patients' ward at the time when everybody is sleeping there (i.e. at night). More formally, this would be: $\neg$*'Wards'* $\vee \neg isTime($*night*$)$, where *'Wards'* and *isTime(night)* represent a goal in the goal model (see Figure 4) and a background fact, respectively. In Figure 6, selection conditions have been added to a few of the OR-subgoals.

In Practice. For our nursing study, we developed goal models with 124 distinct goals that made use of 36 different background predicates. The goal models include 31 distinct OR-decompositions of which 10 are of type *A*, 6 are *P*, 5 are *L*, and the remaining of type *C*, *F*, *E* and *T*. The application clearly showed that following a concern-driven decomposition process, as opposed to using traditional OR-decompositions, allows the discovery of variability aspects that would otherwise remain hidden. Dependencies among alternative responses to variability concerns were also found to be a very common element of the resulting diagrams. The use of condition clauses proved effective for representing such dependencies, though a formalism based on pure predicate logic allowing functions and variables may be more convenient.

Reasoning about the resulting goal models and the background circumstances, is essentially a satisfiability (SAT) problem for the corresponding propositional formula. The goal of alternatives' analysis is the identification of circumstances under which no alternative is appropriate for the fulfillment of a goal. In such cases the analyst needs to introduce new alternatives that are not constrained by such circumstances. In Figure 6, for example, the existence of the subgoal *'Headphone'* is the result of the requirement to have a notification sent even under circumstances in which *'Wards'* $\wedge$ *isTime(night)* is true.

We used Prolog to construct a tool that reads an AND/OR structure with $\xrightarrow{pre}$ constraints as well as a set of facts that describe background circumstances and outputs alternatives that satisfy the constraints. The tool goes through all alternatives of the tree and tests their applicability in the given conditions. Thus, the analyst can collect all background predicates that have been used in conditions, consider truth assignments for them that reflect realistic circumstances and run the procedure to see which alternatives can be considered. The satisfying alternatives can be further subjected to qualitative analysis and prioritization in the spirit of [13, 17]. Although our reasoning tool is rather naive in terms of computational efficiency, it preformed reasonably well with our goal models. Specifically, it can test one million of alternatives in about 40 sec on a PC with a 2GHz P-4 CPU and 768Mb RAM; our models demonstrated a maximum of 4 million alternatives. Recent advances in SAT and #SAT-solving (e.g. [9]) give us confidence that most practical goal models can be analyzed without any performance issues.

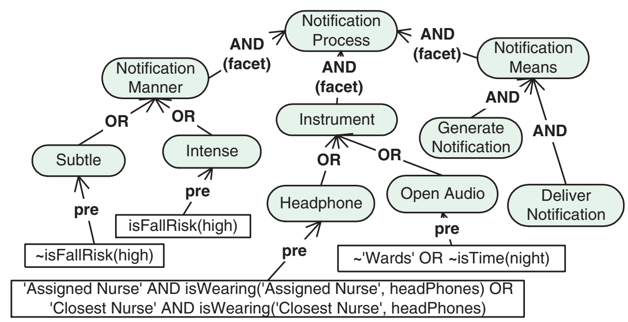

Figure 6. Selection Conditions

7. Conclusions

We illustrated how goal models can be used to capture variability in the early requirements phase before variation points of the system-to-be are defined. Frames of variability concerns can be constructed from textual matter and used in a concern-driven decomposition process, which also allows background non-intentional circumstances to play a role in assessing the result.

From here, we plan to investigate the notion of frame inheritance and its role in constructing domain-specific hierarchies of variability frames. On the other hand, a stronger characterization and representation framework for non-intentional variability should be possible. But is there also a stronger relationship between the two types of variability (intentional and non-intentional), and how is this expressed? Further, what other reasoning procedures can help analysts claim completeness in variability acquisition? Last but not least, how does the characterization of variability at the requirements influences the way it is implemented in the software itself? There is certainly room for more study.

Acknowledgments. We would like to thank prof. Graeme Hirst and prof. Fahiem Bacchus for their invaluable advice on different aspects of this work, as well as Mehrdad Sabetzadeh for his very useful comments on earlier versions of this paper. Funding for this work was provided by the Precarn/IRIS project titled “Intelligent system requirements for cognitively impaired individuals”.

References

[1] A. I. Antón and C. Potts. The use of goals to surface requirements for evolving systems. In *Proceedings of the 20th International Conference on Software Engineering*, 1998.

[2] F. Bachmann, M. Goedicke, J. Leite, R. Nord, K. Pohl, B. Ramesh, and A. Vilbig. A Meta-model for Representing Variability in Product Family Development. In *Proceedings of the 5th International Workshop on Software Product-Family Engineering (PFE5)*, Siena, Italy, 2003.

[3] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The Berkeley FrameNet project. In *Proceedings of the 17th International Conference on Computational Linguistics*, 1998.

[4] J. Castro, M. Kolp, and J. Mylopoulos. Towards requirements-driven information systems engineering: the Tropos project. *Information Systems*, 27(6):365–389, 2002.

[5] P. Clements and L. Northrop. *Software Product Lines*. Addison-Wesley, 2001.

[6] K. Czarnecki and U. W. Eisenecker. *Generative Programming - Methods, Tools, and Applications*. Addison-Wesley, June 2000.

[7] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.

[8] R. Darimont and A. van Lamsweerde. Formal refinement patterns for goal-driven requirements elaboration. In *Proceedings of the 4th ACM SIGSOFT symposium on Foundations of Software Engineering*, pages 179–190, New York, NY, USA, 1996. ACM Press.

[9] N. Eén and N. Sörensson. An extensible SAT-solver. In *Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, 2003.

[10] S. R. Faulk. Product-line requirements specification (PRS): An approach and case study. In *Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE'01)*, 2001.

[11] C. J. Fillmore. The case for case. *In Bach and Harms (Ed.): Universals in Linguistic Theory. New York: Holt, Rinehart, and Winston*, pages 1–88, 1968.

[12] W. Frakes, R. Prieto-Diaz, and C. Fox. Dare: Domain analysis and reuse environment. *Annals of Software Engineering*, 5:125–141, 1998.

[13] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani. Reasoning with goal models. In *Proceedings of the 21st International Conference on Conceptual Modeling (ER'02)*. Springer-Verlag, 2002.

[14] M. L. Griss, J. Favaro, and M. d' Alessandro. Integrating feature modeling with the RSEB. In *Proceedings of the 5th International Conference on Software Reuse (ICSR '98)*, page 76, 1998.

[15] G. Halmans and K. Pohl. Communicating the variability of a software-product family to customers. *Software and System Modeling*, 2(1):15–36, 2003.

[16] R. D. Huddleston and G. K. Pullum. *The Cambridge grammar of the English language*. Cambridge University Press, 2002.

[17] B. Hui, S. Liaskos, and J. Mylopoulos. Requirements analysis for customizable software: A goals-skills-preferences framework. In *Proceedings of the 11th IEEE International Requirements Engineering Conference (RE'03)*, 2003.

[18] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. Feature-oriented domain analysis (FODA) feasibility study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, November 1990.

[19] K. C. Kang, S. Kim, J. Lee, K. Kim, E. Shin, and M. Huh. FORM: a feature-oriented reuse method with domain-specific reference architectures. *Annals of Software Engineering*, 5:143–168, 1998.

[20] S. Liaskos, A. Lapouchnian, Y. Wang, Y. Yu, and S. Easterbrook. Configuring common personal software: a requirements-driven approach. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering (RE'05)*, 2005.

[21] J. McGrenere. Are we all in the same “bloat”? In *Proceedings of Graphics Interface*, pages 187–196, Montreal, Canada, September 2000.

[22] J. Mylopoulos, L. Chung, S. Liao, H. Wang, and E. Yu. Exploring alternatives during requirements analysis. *IEEE Software*, 18(1):92–96, 2001.

[23] N. Prat. Goal formalization and classification for requirements engineering. In *Proceedings of the 3rd International Workshop on Requirements Engineering: Foundation for Software Quality*, Barcelona, Spain, 1997.

[24] R. Prieto-Díaz. Domain analysis: an introduction. *SIGSOFT Software Engineering Notes*, 15(2):47–54, 1990.

[25] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Grammar of contemporary English*. Longman, 1972.

[26] C. Rolland, G. Grosz, and R. Kla. Experience with goal-scenario coupling in requirements engineering. In *Proceedings of the 4th IEEE International Symposium on Requirements Engineering (RE'99)*, page 74, Washington, DC, USA, 1999. IEEE Computer Society.

[27] C. Rolland, C. Souveyet, and C. B. Achour. Guiding goal modeling using scenarios. *IEEE Transactions on Software Engineering*, 24(12):1055–1071, 1998.

[28] A. Sutcliffe, S. Fickas, and M. Sohlberg. Personal and contextual requirements engineering. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering (RE'05)*, Paris, France, September 2005.

[29] W. Tracz. DSSA (domain-specific software architecture): pedagogical example. *SIGSOFT Software Engineering Notes*, 20(3):49–62, 1995.

[30] D. M. Weiss and C. T. R. Lai. *Software Product-Line Engineering. A Family-Based Software Development Process.* Addison-Wesley, July 1999.

[31] E. S. K. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE Int. Symposium on Requirements Engineering (RE'97)*, Washington D.C., USA, January 1997.