UNIVERSITY
OF TRENTO

DIPARTIMENTO DI INGEGNERIA E SCIENZA DELL'INFORMAZIONE

38100 Povo – Trento (Italy), Via Sommarive 14
http://www.disi.unitn.it

Modeling and Analyzing Contextual Requirements

Raian Ali, Anders Franzen, Alberto Griggio, and Paolo Giorgini

April 2009

Technical Report # DISI-09-019

Modeling and Analyzing Contextual Requirements

Raian Ali, Anders Franzen, Alberto Griggio, and Paolo Giorgini

University of Trento - DISI, 38100, Povo, Trento, Italy
{raian.ali, franzen, alberto.griggio, paolo.giorgini}@disi.unitn.it

Abstract. The relation between contexts and requirements can be very complex to analyze. A context can motivate a requirement, a requirement can be satisfied only in a specific context, and a context can influence the quality of each possible alternative of satisfying a requirement. To capture and deeply understand this relation, we need to start from the reasons of a requirement, namely stakeholders goals, and analyze at this level the system variability with respect to the context. In this paper, we propose a goal-based approach to model and analyze contextual requirements. We adopt Tropos goal modeling framework where we introduce contextual variation points and provide a set of constructs to hierarchically analyze contexts. We also articulate a new problem, the *context interaction problem*; we study its influence on both monitoring and functional requirements, and we finally provide a SAT-based approach to deal with it. We show the process for creating our models and illustrate our approach on a museum-guide scenario.

1 Introduction

There is a continuous need for adaptive systems with a degree of autonomy to take decisions by themselves with the minimum intervention of humans. In other words, it becomes more and more important to reduce the human intervention in readjusting the system configuration, or even worst stopping, redesigning, updating and starting again the system. As a baseline, we need to identify during the analysis the parameters that may introduce the need of changing the behavior of the system at run-time, what alternatives the system may have to accommodate variations of such parameters, and how the system can decide among them.

Although the notion of context [1] has been adopted and currently considered in the software engineering literature as a main stimulus for changes of the system behavior, the relation between context and system requirements has not deeply investigated and it still leaves many open research problems. We believe that to fully understand the role of the context in the behavior of a system, context has to be considered along with stakeholders goals since the very early phases of the analysis.

Goal analysis (*i** [2], Tropos [3], and KAOS [4]), provides a way to analyze high level goals and then to discover and represent alternative sets of the tasks that can be adopted to achieve such goals. Goal models – a mainstream technique in requirements engineering – are used to represent the rationale of both humans and software systems, and help representing software design alternatives [5]. These features are also important for self-contextualizable software that must allow for alternatives and have a rationale to reflect users and software adaptation to the context in order to adopt one useful execution course [6].

In [7, 8] [1], we have introduced the self-contextualizable Tropos goal model where contexts are defined and associated to variation points of a goal model. The model, however, can become easily very complex and it may introduce redundancy and/or inconsistency problems in both functional and monitoring requirements. The relations between contexts, namely the implications, can cause what we call *context interaction problem*, that is analogous to and complex as the *feature interaction problem* [9].

In this paper, we build on and extend our previous work ([7, 8, 10, 11]) to provide an engineering framework for self-contextualizable requirements. We describe our contextual goal model that captures the relation between requirements and context and helps to identify the monitoring requirements together with the functional ones. We define the context interaction problem and argue about its generality for self-contextualizable systems. We show the importance of context interaction problem by studying its influence on functional and monitoring requirements, and provide a SAT-based automated reasoning technique to handle with it. We also show the methodological activities for building our set of models, and explain the whole approach on a running example of a system for assisting visitors of museums.

The paper is structured as follows: in Section 2, we explain our contextual goal model. In Section 3, we study the context interaction problem, and in Section 4, we provide an automated reasoning that treats the redundancy, triviality, and inconsistency in a context. In Section 5, we outline a set of activities that produce our proposed models. We discuss the related work in Section 6, and conclude our work in Section 7.

2 Contextual Goal Model

We believe that a contextualizability study has to start at the requirements level, where stakeholders goals are identified and analyzed, and where the system behaviors can differ according to the context. The idea is to integrate goal analysis, as an early requirements phase, and context analysis in order to systematically derive contextualized goal satisfaction alternatives. In turn, as we introduced in [7, 8], context itself needs to be modeled and analyzed. Thus, as shown in Fig. 1, goal analysis provides constructs to hierarchically analyze goals and discover alternative sets of tasks the system can execute to satisfy such goals, while context analysis provides constructs to hierarchically analyze context and discover alternative sets of facts the system has to monitor to verify a context and then act accordingly.

Fig. 2 represents a Tropos goal model for an information system that is intended to assist visitors of a museum and interact with them and assistance staff, mainly through their PDAs. The model represents the different alternatives of satisfying the main goal of the system of giving information to visitors about the pieces of art in the museum. To make it self-contextualizable, we need to explicitly represent the relation between the possible alternatives and the context. Contexts, that are labeled as $C1..C8$ on Fig. 2, can be related to the following variation points of the goal model:

1. *Or-decomposition*: is the basic goal variability relation where we can specify the context in which each alternative goal/task is adopted. E.g., to provide information

[1] The papers are to appear, you can get them from http://disi.unitn.it/∼ali/

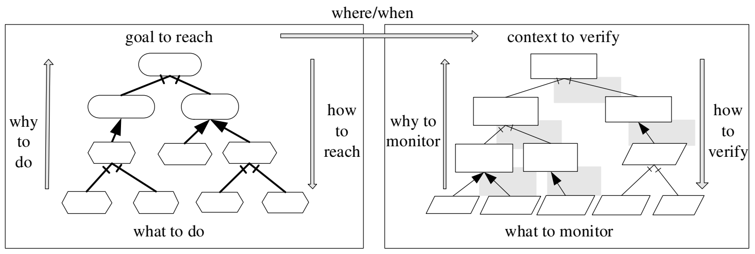

Fig. 1. The analogy between Goal and Context Analysis

about a piece of art, a visitor can be directed to a dedicated terminal when such terminal is free and close to the visitor and he/she is able to use and interact with it $(C2)$; the visitor's PDA can be alternatively used to show information when the piece of the art information are not so complicated, and the visitor has the ability and the knowledge to use PDAs $(C3)$, while getting information through an assistance staff requires that the visitor is not able to use PDA and not familiar with terminals, or that the visitor is classified as an important visitor $(C4)$.

2. *Goal/Task activation*: depending on the context, an actor might decide to satisfy a goal / execute a task. E.g., to initiate the goal *"visitor gets informed about a piece of art"*, the system needs to verify if the visitor is interested in the piece, if the visitor does not already know about it, and if there is still time to explain about it $(C1)$.
3. *Means-end*: goals can be ultimately satisfied by means of specific executable processes (*tasks*). The adoptability of each task in means-end analysis might depend on the context. E.g., the visitor can be notified about the availability of information terminals through a voice message when he/she puts the headphones on, and is not talking to somebody or using his/her PDA for a call $(C5)$, while notifying him/her by SMS can be adopted in the opposite context $(\neg C5)$
4. *Actors dependency*: in some contexts, an actor might attain a goal / get a task executed by delegating it to another actor. E.g., the dependency between the two system actors for providing the information to a visitor through an assistance staff requires a staff that is close to and talks a language common to the visitor, and knows enough about the considered piece of art comparing to the visitor knowledge $(C6)$.
5. *And-decomposition*: a sub-goal/sub-task might (or might not) be needed in a certain context, that is some sub-goals / sub-tasks are not always mandatory to fulfil the top-level goal/task in an And-decomposition. E.g., guiding the assistance staff to the visitor place is not needed if the visitor stays around and can be seen directly by the assistance staff $(C8)$.
6. *Contribution to softgoals*: softgoals [2] are qualitative objectives for their satisfaction there is no clear cut criteria, and they can be contributed either positively or negatively by goals and tasks. The contributions to softgoals can vary from one context to another. We need to specify the relation between the context and the value of the contribution. E.g., giving the information in person is comfortable to the assistance staff if the visitor is in the same room as the assistant $(C7)$, while it is not comfortable when they are in different rooms.

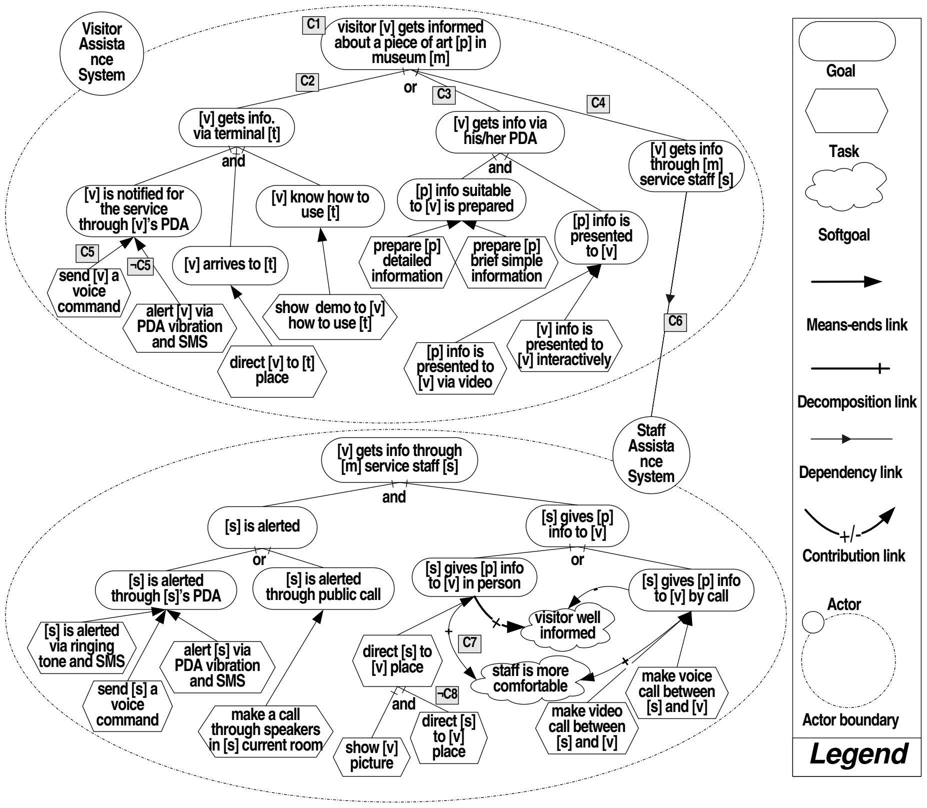
Visitor Assistance System
C1
visitor [v] gets informed about a piece of art [p] in museum [m]
C2
or
C3
C4
[v] gets info. via terminal [t]
[v] gets info via his/her PDA
[v] gets info through [m] service staff [s]
and
and
[v] is notified for the service through [v]'s PDA
[v] know how to use [t]
[p] info suitable to [v] is prepared
[p] info is presented to [v]
C5
¬C5
send [v] a voice command
alert [v] via PDA vibration and SMS
[v] arrives to [t]
direct [v] to [t] place
show demo to [v] how to use [t]
prepare [p] detailed information
prepare [p] brief simple information
[p] info is presented to [v] via video
[v] info is presented to [v] interactively
C6
Staff Assistance System
[v] gets info through [m] service staff [s]
and
[s] is alerted
[s] gives [p] info to [v]
or
or
[s] is alerted through [s]'s PDA
[s] is alerted through public call
[s] gives [p] info to [v] in person
[s] gives [p] info to [v] by call
[s] is alerted via ringing tone and SMS
send [s] a voice command
alert [s] via PDA vibration and SMS
make a call through speakers in [s] current room
visitor well informed
staff is more comfortable
C7
direct [s] to [v] place
and
¬C8
show [v] picture
direct [s] to [v] place
make video call between [s] and [v]
make voice call between [s] and [v]
Goal
Task
Softgoal
Means-ends link
Decomposition link
Dependency link
+/-
Contribution link
Actor
Actor boundary
Legend

Fig. 2. Tropos goal model for the museum assistance system

We need to analyze context to discover, represent, and agree on how it can be verified. Differently from the other research in context modeling (for a survey see [12]), we do not provide an ontology or a modeling language for representing context, but modeling constructs to hierarchically analyze context. Moreover and to clarify the domain of discourse, i.e. the elements of the environment under discussion, along goal and context analysis, we parameterize the goal and context models. Taking the parameterized root goal *"visitor[v] get informed about a piece of art [p]"*, the analysis of its activation context ($C1$) is shown in Fig. 3.

Here we explain the set of modeling constructs we provide to analyze a high level context and identify the atomic verifiable facts that give its truth value.

Definition 1 (Fact) *a boolean predicate specifying a current or a previous context, whose truth value can be computed objectively.*

The objective method to compute a fact truth value requires monitoring some characteristics and/or history of a set of relevant environment elements. Facts are graphically represented as parallelograms as in Fig. 3.

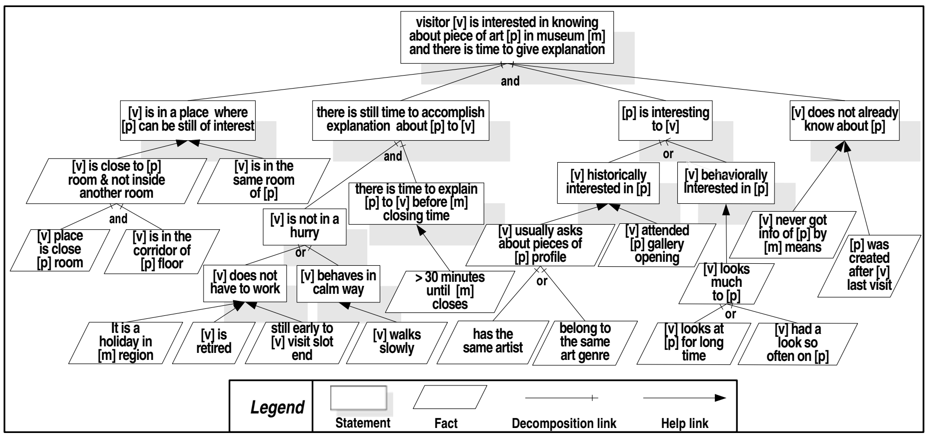
visitor [v] is interested in knowing about piece of art [p] in museum [m] and there is time to give explanation
and
[v] is in a place where [p] can be still of interest
there is still time to accomplish explanation about [p] to [v]
[p] is interesting to [v]
[v] does not already know about [p]
[v] is close to [p] room & not inside another room
[v] is in the same room of [p]
and
or
[v] historically interested in [p]
[v] behaviorally Interested in [p]
and
[v] place is close [p] room
[v] is in the corridor of [p] floor
[v] is not in a hurry
there is time to explain [p] to [v] before [m] closing time
[v] usually asks about pieces of [p] profile
[v] attended [p] gallery opening
[v] never got info of [p] by [m] means
[p] was created after [v] last visit
or
[v] does not have to work
[v] behaves in calm way
> 30 minutes until [m] closes
or
[v] looks much to [p]
or
It is a holiday in [m] region
[v] is retired
still early to [v] visit slot end
[v] walks slowly
has the same artist
belong to the same art genre
[v] looks at [p] for long time
[v] had a look so often on [p]
Legend
Statement
Fact
Decomposition link
Help link

Fig. 3. The context analysis hierarchy for $C1$

Definition 2 (Statement) *a boolean predicate specifying a current or a previous context, whose truth value can not be computed objectively.*

Statement verification could not be objectively done because the system is not able to monitor and get all the data needed to compute the truth value of a statement, or because there could be no consensus about the way of knowing the truth value of a statement. To handle such problem we adopt a relaxed confirmation relation between facts, which are objectively computable by definition, and statements, in order to assign truth values to statements. We call this relation *"help"* and define it as following:

Definition 3 (Help) *Let f be a fact, s be a statement.* $help(f, s) \iff f \rightarrow s$

The relation $help$ is strongly subjective, since different stakeholders could define different $help$ relations for the same statement, e.g., one stakeholder could say $help(f_1, s) \wedge help(f_2, s)$, whereas another one could say $helps(f_2, s) \wedge help(f_3, s)$. Statements are graphically represented as shadowed rectangles, and the relation $help$ is graphically represented as a filled-end arrow between a fact and a related statement as in Fig.3.

Definition 4 (And-decomposition) *Let* $\{s, s_1, \ldots, s_n\},\ n \geq 2$ *be statements (facts).* $and\_decomposed(s, \{s_1, \ldots, s_n\}) \iff s_1 \wedge \ldots \wedge s_n \rightarrow s$

Definition 5 (Or-decomposition) *Let* $\{s, s_1, \ldots, s_n\},\ n \geq 2$ *be statements (facts).* $or\_decomposed(s, \{s_1, \ldots, s_n\}) \iff \forall i \in \{1, \ldots, n\}, s_i \rightarrow s$

Decomposition is graphically represented as a set of directed arrows from the sub-statements (sub-facts) to the decomposed statement (fact) and labeled by *And* or *Or* as in Fig. 3. Let us illustrate the above context analysis constructs by examples:

– *"the piece of art [p] artist [a] has lived in the visitor's [v] city of birth"* is a fact the system can verify through checking the profile of the artist and the city of birth of the registered visitor.

– *"the visitor [v] is interested in the piece of art [p]"* is a statement since the system can not objectively compute its truth value. This statement can be Or-decomposed into *"[v] is behaviorally interested in [p]"* and *"[v] is historically interested in [p]"* substatements. The system can get some evidence of the first substatement through the help of the fact *"[v] looks at [p] much"* that is Or-decomposed into the fact *"[v] looks at [p] for long time"*, and the fact *"[v] comes to [p] area and has a look at [p] so often"*. The second substatement can be verified through the fact *"[v] usually asks for pieces from the [p] art genre"* or the fact *"[v] usually asks for pieces made by the same [p] artist"*.
– *"visitor [v] does not already know about the piece of art [p]"* is a statement that can be verified through the fact *"[p] is created after the last visit of [v] to the museum"*, or the fact *"[v] never got information of [p] by any of the museum means"*. However both facts do not ensure that the visitor does not know about the piece (e.g., the system cannot verify if the visitor has been told about it by a friend or read an article about it somewhere).

As discussed in [1], context is the reification of the environment in which the system is supposed to operate. Each single fact and statement is a context, and our proposed reification hierarchy relates different subcontexts into one more abstract. Moreover, and considering that context is the reification of the environment, our context analysis is motivated by the need for constructs to analyze context to discover the relevant atomic data that represent that environment, i.e. the data the system has to monitor. For example, and taking the leaf facts of the statement *"piece of art [p] is interesting to visitor [v]"* $(ST1)$ of Fig. 3, the analyst could elicit the data conceptual model of Fig. 4, and we can look at facts and statements as views (reifications) over it.

Our proposed hierarchical context analysis helps to make a context (i) more understandable for the stakeholders, (ii) more modifiable as it is not given as one monolithic block, and (iii) more reusable as parts of the analysis hierarchy can be also used for other variation points or other stakeholders context specifications. Moreover, the analysis justifies *why* the monitoring system has to capture environmental data (like the data of Fig. 4), as such data are needed to verify leaf facts that propagatively confirm or disapprove the context needed to make a decision at the goal model. However, the context hierarchy, representing either direct or accumulated context, suffers of interaction problems that can make a context trivial, redundant, or even inconsistent as we are going to explain in the next section.

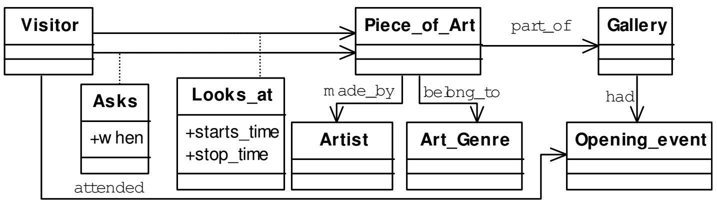

Fig. 4. The conceptual model of data needed to verify $ST1$ leaf facts

3 Context Interaction Problem

In this section, we articulate the problem of context interaction, show its importance, and argue about its generality and complexity. The context hierarchial analysis we proposed, is easily transformable into a propositional formula consists of the leaf facts as atomic predicates (variables). The relations, namely the implications, between these facts can make the context formula redundant, trivial, or even inconsistent. Here we give the definition of *redundant* context and its two extremes: the *trivial* and the *inconsistent*.

Definition 6 (Redundant Context) *given the implications between its facts, a context is redundant iff some facts has no effect on its validity.*

Definition 7 (Trivial Context) *given the implications between its facts, a context is trivial iff it is always reduced to true.*

Definition 8 (Inconsistent Context) *given the implications between its facts, a context is inconsistent iff it is always reduced to false.*

Context redundancy makes the context representation more complex without justification and leads to a useless monitoring of facts that have no effect on its validity. Context redundancy motivates us to *optimize monitoring requirements*. Let us take the two facts $f1 =$ *"the visitor is inside a museum room"* and $f2 =$ *"there is enough light at the location of the visitor"*. Consider a context $C = f1 \wedge f1$; if the system is going to operate in a museum that its rooms are well illuminated then $f1 \rightarrow f2$ and C is reduced to $f1$ which means that there is no need to install sensors to capture the level of light in the museum. Alternatively, if $C = f1 \vee f2$ then C is reduced to $f2$ and there will be no need for installing a positioning system to decide if the visitor is inside a museum room. Some base reductions rules are shown in the following table:

Assured Implication (Assumptions)	$A \vee B \leftrightarrow$	$A \wedge B \leftrightarrow$
$A \rightarrow B$	B	A
$A \rightarrow \neg B$	$AxorB$	$false$
$A \leftrightarrow \neg B$	$true$	$false$

While the redundancy of context implies a redundancy in monitoring requirements; context inconsistency adds to that a redundancy in the software functionalities. Besides the uselessness of monitoring their facts, inconsistent contexts deny the adoptability of the software functionalities preconditioned by them. E.g., if a functionality is preconditioned by the context $C = f1 \wedge f2$, and if the museum rooms are not well illuminated for some decorating reasons or to conserve the pieces of art, i.e. $f1 \rightarrow \neg f2$, then such functionality is never adoptable since C is inconsistent.

After showing its influence, now we argue about the generality of the context interaction problem and that it is not tied to or caused by our context analysis and goal model. We expect any contextual system to monitor several pieces of information that could be also combined through logical relations to conform logical expressions. Assuring some implications between these pieces might reveal a problem of redundancy, triviality, or inconsistency. Let us take the following generic pseudo code that reflects, or can be part of, any contextual decision model (including our contextual goal model):

```
1: if $(A \vee B) \wedge C$ then
2:    if $D \wedge E$ then
3:       adopt alternative set of actions (action_set 1)
4:    else
5:       if $F \vee G$ then
6:          adopt another set of actions (action_set 2)
7:       end if
8:    end if
9: end if
```

The three contexts boolean abstractions we have here are $(A \vee B) \wedge C$, $D \wedge E$, and $F \vee G$, that involve monitoring the set of facts $S = \{A, B, C, D, E, F, G\}$. The model has two alternative set of actions *action_set 1* and *action_set 2* each fitting to a certain context. When we assure the implication $C \rightarrow A$ then $(A \vee B) \wedge C$ can be reduced to just C and therefore there will be no need to monitor A and B. If there is no implication between D and E then $D \wedge E$ alone is not redundant, but this does not mean that *action_set 1* is adoptable or D and E are not redundant; suppose we have the implication $C \rightarrow \neg E$, then the accumulated context at line 2, which is $C \wedge D \wedge E$, is inconsistent and reduced to *false*, and the *action_set 1* is not adoptable. In case we assure that $C \rightarrow \neg G$ then the accumulated context at line 5, $C \wedge (F \vee G)$, can be reduced to $C \wedge F$ which means that G is redundant and has no effect on the validity of the accumulated ontext.

The implications between facts can be *absolute* or *dependent* on the characteristics of the system environment. Absolute implications are applied wherever the system has to operate, e.g., $f3 \rightarrow \neg(f4 \wedge f5)$ where $f3 =$*"piece of art [p] is always exclusive to museum [m]"*, $f4 =$*"visitor is visiting the museum [m] for the first time"* and $f5$=*"visitor has seen [p] some date before today"*. Other implications depend on the nature of the environment the system is going to operate in, like the mentioned implication between the light level and being inside a museum room. Moreover, the environment itself assures that some contexts are always true or false, so we have to consider a special kind of dependent implications between the system environment and context analysis facts, i.e. $Env \rightarrow facts\_formula$. E.g., if the museum opens only in holidays, so the fact *"the day is holiday in museum region"* is always true.

Facts verification has costs; costs are those related to the facts verification process itself and to getting the data (e.g., the one of Fig. 4) needed to make the verification possible, such as installing physical equipments like sensors, inserting data by human operators, having enough storage, processing time and so on. When we have more than one possibility to reduce contexts, we select the one that minimizes the costs.

After the above explanation of the context interaction problem, we now explain the two main sub-problems that we need to face in order to solve it:

– *Optimizing monitoring and functional requirements*: checking and fixing the redundancy, triviality, and inconsistency of contexts lead to minimizing the costs of the system as it avoids us sensing, storing, processing data to verify facts that have no effect on any decision, and developing software functionalities that are preconditioned by inconsistent contexts until such inconsistencies are fixed.

Let us consider the contextual goal model of Fig. 5. Whenever the analyst defines a direct context at each variation point ($C1$, $C2$, $C3$, $C4$, $C5$, $C6$), the automated reasoning has to check if this direct context alone and if the accumulated context ($C1$, $C1 \wedge C2$, $C1 \wedge C3$, $C1 \wedge C4$, $C1 \wedge C5$, $C1 \wedge C3 \wedge C6$ respectively) are consistent and non trivial, and notify the analyst to fix any error before repeating the check and proceeding with the next contexts. However, this check has also to be done progressively for the accumulated context on the alternatives in the goal model to know if they can be adopted together, e.g., if $C1 \wedge C2 \wedge C5$ is inconsistent then the root goal satisfaction alternative $G5, G3, G8$ is never adopted.
After defining contexts at all of the variation points and passing the consistency and triviality check, we can start to optimize the monitoring requirements. Optimization takes the set of all contexts associated to the different root goal satisfaction alternatives and softgoals, i.e. $A1C$, $A2C$, $A3C$, $A4C$, and $SG1C$ and gives equivalent reduced contexts $A1C'$, $A2C'$, $A3C'$, $A4C'$, and $SG1C'$ that can be verified on a sub-set of facts with minimum monitoring costs. The analyst has to study the set of facts of the resulted formulas and elicit the data that the monitoring system has to obtain like the one we showed in Fig. 4.

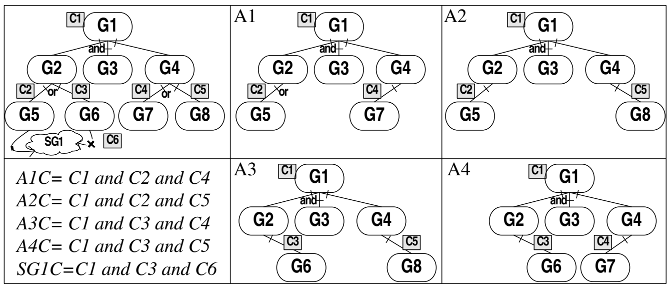

Fig. 5. The accumulated context for the root goal satisfaction alternatives and softgoals

The automated reduction has to minimize the total cost of monitoring all the reduced contexts facts, as doing it separately for each context of $A1C$, $A2C$, $A3C$, $A4C$, and $SG1C$ does not guarantee, in the general case, having the minimal total monitoring costs. The problem of optimizing a set of contexts together to reduce the overall cost is highly expensive as we explain later. In the next section, we provide an algorithm, based on SAT techniques and greedy algorithms, that takes a context formula together with the implications between facts (assumptions), checks its consistency and redundancy, and produces an equivalent formula with less costs.

– *Efficient specification of implications*: when the number of facts is big, it will be hard for the analyst to specify even the binary implications between facts. Moreover, the specified implications themselves might be wrong and inconsistent. Designing a supporting tool that helps the analyst to correctly specify, with a minimum number of interactions, the implications between facts is another main problem. We expect such tool to make a kind of facts analysis and asks the analyst to specify the relation where the probability of implication is high.

4 SAT-based Redundancy Elimination

In this section we describe our algorithm for determining whether a context is inconsistent or trivial, and for identifying redundant facts in a context. The algorithm is based on propositional satisfiability (SAT), and in particular on SAT-based techniques for the enumeration of all the models of a propositional formula. Before describing the algorithm, we recall some necessary definitions and notions from propositional logic.

Definition 9 (Model, Satisfiability, Equivalence) *Let φ be a propositional formula, and $V(\varphi)$ be the set of its atomic predicates. Let μ be a function $\mu : V(\varphi) \rightarrow \{0, 1\}$, and let ν be a function from propositional formulas to the set $\{0, 1\}$ defined as:* [2]

$$\nu(P) = \mu(P), P \in V(\varphi) \qquad \nu(\neg\phi) = 1 - \nu(\varphi) \qquad \nu(\phi \wedge \psi) = min(\nu(\phi), \nu(\psi))$$

μ is a model *for φ if $\nu(\varphi) = 1$. φ is* satisfiable *if it has at least one model,* unsatisfiable *otherwise.*

Two formulas ϕ and ψ are equivalent *if and only if they have the same models. A formula ϕ* entails *another formula ψ, denoted as $\phi \models \psi$, if all the models of ϕ are also models of ψ, but not vice versa.*

In what follows, we might denote a model μ as a *set of literals* μ_S, such that for each variable P, if $\mu(P) = 1$ then $P \in \mu_S$, and if $\mu(P) = 0$ then $\neg P \in \mu_S$. Analogously, we might denote μ as a *formula* μ_F which is the conjunction of the literals in μ_S.[3]

Example 1. Let φ be the formula $(P \vee Q) \wedge (R \vee \neg S) \wedge (S \vee P)$. Then $\mu := \{P, Q, \neg R, \neg S\}$ is a model for φ.

Definition 10 (Equivalence under assumptions) *Let ξ and φ be two formulas. Then a formula φ' s.t. $\xi \models \varphi \leftrightarrow \varphi'$ is said to be equivalent to φ under the assumption of ξ.*

Example 2. Let P and Q be predicates. Given the definition of equivalence under assumptions, $P \wedge Q$ is equivalent to P under the assumption $P \rightarrow Q$ since $P \rightarrow Q \models (P \wedge Q) \leftrightarrow P$. There are other formulas which will be equivalent, e.g. $P \wedge Q$ is trivially equivalent to itself.

By substituting every fact (a predicate) in a context with a fresh propositional variable (*fact variable*) we obtain the *boolean abstraction* of a context. In the same way, we can obtain the boolean abstraction of the assumptions which are known to be true in a context. Given the boolean abstraction for a context φ and the corresponding assumptions ξ we can express the problem of reducing redundancy of contexts as the problem of finding an equivalent context φ' which is equivalent to φ under the assumptions ξ.

In order to obtain such a φ', we exploit SAT solvers, and in particular techniques for generating all the models of a boolean formula. The pseudo-code of our algorithm is reported in Fig. 6. The algorithm enumerates the models of the boolean abstraction φ of the context, and for each such model μ it checks whether μ is compatible with the assumptions ξ (which express the known implications between facts). If μ is compatible

[2] We define ν only for the connectives $\neg, \wedge$ since they are enough to express all the others.

[3] Moreover, we shall drop the subscripts $_S$ and $_F$ when they are clear from the context.

```
Input: context $\varphi$, assumptions $\xi$
Output: reduced context $\varphi'$
 1: $\varphi' \leftarrow \bot$
 2: for all models $\mu$ of $\varphi$ do
 3:     if Is_Satisfiable($\mu \wedge \xi$) then
 4:         for all literals $l \in \mu$ do
 5:             $\mu' \leftarrow \mu \setminus \{l\}$
 6:             if not Is_Satisfiable($\mu' \wedge \xi \wedge \neg\varphi$) then
 7:                 $\mu \leftarrow \mu'$
 8:             end if
 9:         end for
10:         $\varphi' \leftarrow \varphi' \vee \mu$
11:     end if
12: end for
13: return $\varphi'$
```

Fig. 6. Pseudo-code of the context reduction algorithm.

with the assumptions, the algorithm tries to reduce μ by removing literals from it as long as it is still a model for φ *under the given assumptions*, that is, as long as $\mu \wedge \xi \models \varphi$, or in other words as long as $\mu \wedge \xi \wedge \neg\varphi$ is unsatisfiable. Then, the reduced context is given by taking the disjunction of all the reduced models that are compatible with the assumptions.

Theorem 1. *Let ξ and φ be two formulas, and let φ' be the result of applying the algorithm of Fig. 6 to φ and ξ. Then φ' is equivalent to φ under the assumptions ξ.*

Proof. We have to show that:

1. every model of φ that is compatible with ξ is also a model of φ'; and
2. for each model μ of φ', all its extensions to all the variables in $V(\varphi) \setminus V(\varphi')$ that are compatible with ξ are models of φ.

1. Let μ be a model of φ compatible with ξ (that is, $\mu \wedge \xi$ is satisfiable). Then, by construction (lines 4-10 of the algorithm) φ' contains a subset of μ as a disjunct. Therefore, μ is a model for φ'.
2. Let μ be a model of φ' compatible with ξ. Since φ' is a disjunction of conjunctions of literals, μ must be a superset of the set of literals σ in one of such conjunctions. We can assume w.l.o.g. that σ is the smallest such set, because clearly if μ satisfies $\psi \wedge l$, then μ satisfies also ψ. Moreover, the variables occurring in μ are a subset of the variables of φ. Consider any extension μ' of μ to all the variables of φ, such that μ' is compatible with ξ, and suppose that μ' is not a model for φ. Then μ' can be turned into a model for φ by flipping [4] some of the literals in $\mu' \setminus \sigma$, since by construction the literals in σ occur in a model of φ compatible with ξ (lines 3-10 of the algorithm). Let η be a minimal set of literals to flip to obtain a model μ'' of φ from μ'. By construction, $\mu'' \wedge \xi \wedge \neg\varphi$ is unsatisfiable, and for all the literals l in η,

[4] Flipping a literal here means replacing l with $\neg l$ or vice versa.

$(\mu'' \setminus \{l\}) \wedge \xi \wedge \neg\varphi$ is satisfiable. [5] But then, none of the literals in η would have been removed from μ'' by the algorithm (lines 4-9) when processing μ'' (which must have been processed since it is a model of φ), and so η must be a subset of σ, which is a contradiction. Therefore, μ' is a model for φ.

□

Example 3. Let the context be $\varphi = (P \wedge Q) \vee R$, and we wish to reduce this formula under the assumption $\xi = (P \rightarrow \neg Q) \wedge (P \rightarrow R)$

To obtain a reduced context we can enumerate all models of φ and reduce given the assumption in this way:

1. We set up the algorithm by setting $\varphi' \leftarrow \bot$
2. φ is satisfiable, and the first model returned is e.g. $\mu = \{\neg P, \neg Q, R\}$
   (a) $\neg P \wedge \neg Q \wedge R \wedge ((P \rightarrow \neg Q) \wedge (P \rightarrow R))$ is satisfiable (line 3), so the model is compatible with the assumptions.
   (b) Since $R \wedge (P \rightarrow \neg Q) \wedge (P \rightarrow R) \wedge \neg((P \wedge Q) \vee R)$ is unsatisfiable, both $\neg P$ and $\neg Q$ are redundant in this model, so they are removed from μ' in lines 4-9 of the algorithm.
   (c) Update $\varphi' \leftarrow R$
3. the second model of φ returned is e.g. $\mu = \{P, Q, \neg R\}$
   (a) As $P \wedge Q \wedge \neg R \wedge ((P \rightarrow \neg Q) \wedge (P \rightarrow R))$ is unsatisfiable (line 3), the model is not compatible with the assumptions, so we skip lines 4-10.
4. the other two models returned are $\mu = \{P, \neg Q, R\}$ and $\mu = \{\neg P, Q, R\}$. As above, they can be reduced to $\{R\}$ only, since $R \wedge (P \rightarrow \neg Q) \wedge (P \rightarrow R) \wedge \neg((P \wedge Q) \vee R)$ is unsatisfiable (line 6).

The resulting reduced context becomes $\varphi' = R$, and we have found that P and Q are redundant for this context.

We remark that the above algorithm can be used also to detect inconsistent or trivial contexts: in the former case, none of the models will be compatible with ξ, so φ' will be always equal to $\bot$; in the latter case, $\xi \wedge \neg\varphi$ will be always unsatisfiable, so in the loop of lines 8-10 all the literals would be removed from μ, which will therefore be reduced to $\top$. However, for efficiency reasons it might be preferable to check for inconsistency and triviality before entering the main loop of lines 2-12, by checking the unsatisfiability of the formulas $\xi \wedge \varphi$ and $\xi \wedge \neg\varphi$ respectively.

Efficiency of the algorithm The algorithm enumerates all models, and in the worst case there are an exponential number of them. For each model, we solve a number of SAT problems. So in the worst case, we need to solve an exponential number of NP-complete problems.

Despite this, the cost of the calls to a SAT procedure can be greatly reduced by using an incremental SAT solver such as MiniSat [13]. The call on line 3 will use the same formula ξ in every iteration of the outer loop, only varying the model μ. In this

[5] Because $((\mu'' \setminus \{l\}) \cup \{\neg l\}) \wedge \xi \not\models \varphi$, so $((\mu'' \setminus \{l\}) \cup \{\neg l\}) \wedge \xi \wedge \neg\varphi$ is satisfiable, and so also $(\mu'' \setminus \{l\}) \wedge \xi \wedge \neg\varphi$ is satisfiable.

case, one single solver instance containing ξ can be reused from one iteration to the next. In the same way, the call on line 6 uses the same formula $\xi \wedge \neg\phi$ in each call, only varying the model μ. A single SAT solver instance can be reused for all these calls. The advantage of using an incremental SAT solver for each of these three cases is that everything learnt from the formulas in one iteration of the outer loop can be reused for all following iterations and will not have to be rediscovered. Lastly, enumerating all models can be done with an efficient algorithm for the all-SAT problem.

Further optimizations are possible. E.g. the number of iterations in the loop enumerating all models on line 2 can be reduced by *blocking clauses* gained from the reduction. We can conjunct the negation of the reduced model μ computed on lines 4–9 to the formula ϕ after each iteration. In example 3 above, this improvement would remove the two last iterations.

4.1 Greedy Strategies for Cost Reduction

In the problem of reducing contexts, we wish to remove redundant facts from the context. This corresponds to producing a formula φ' with less variables than φ. In fact, we want to reduce the *cost* of monitoring facts in a context. If we associate a cost (a real number) to each fact variable in the boolean abstraction of a context, our aim is that of finding a φ' such that the sum of the costs of the variables occurring in φ' is smaller than the sum of the costs of the variables occurring in φ.

As presented, our context reduction algorithm does not take costs into account. One simple possibility to make it aware of costs is to apply some *greedy* strategies when determining the order in which variables are eliminated from the current model (line 4 of Fig. 6). For example, one strategy could be to sort the variables in the model μ according to their cost, to try to eliminate more expensive variables first. A more sophisticated strategy could also consider whether a variable already occurs in the current φ' constructed so far, to try to keep the set of variables $V(\varphi')$ as small as possible.

Example 4. Consider the following context formula φ and assumptions ξ:

$$\varphi = ((\neg P \vee \neg R) \wedge (\neg Q \vee \neg S)) \vee (\neg P \wedge S)$$
$$\xi = (P \leftrightarrow Q) \wedge (R \leftrightarrow S) \wedge (S \rightarrow Q)$$

Suppose that the cost of P is 3, that of Q is 1, that of R is 5 and that of S is 4. Moreover, suppose that the first model found by the algorithm of Fig. 6 that is compatible with ξ is $\mu_1 = \{P, Q, \neg R, \neg S\}$.

If the algorithm does not consider costs, μ_1 might get reduced in lines 4-9 to $\{\neg R\}$. Therefore, after the first iteration of the loop of lines 2-12, $\varphi' = \neg R$. The only other model of φ that is compatible with ξ is $\mu_2 = \{\neg P, \neg Q, \neg R, \neg S\}$. In this case, μ_2 might get reduced to $\{\neg P\}$, and thus the resulting reduced context φ' would be $\neg P \vee \neg R$, whose cost is 8.

However, if costs are considered, in the process of reducing μ_1 and μ_2 the algorithm would try to eliminate first the more expensive variables, resulting in the reduced models $\{\neg S\}$ and $\{\neg Q\}$ respectively. Therefore, in this case the reduced context φ' would be $\neg S \vee \neg Q$, whose cost is 5.

```
Input: context $\varphi$, assumptions $\xi$
Output: reduced context $\varphi'$
 1: $\varphi' \leftarrow \varphi$
 2: for all subsets $S$ of variables $V(\varphi)$ do
 3:     for all formulas $\psi(S)$ over $S$ do
 4:         if $\xi \models \psi(S) \leftrightarrow \varphi$ then
 5:             if cost of $\psi(S)$ is lower than cost of $\varphi'$ then
 6:                 $\varphi' \leftarrow \psi(S)$
 7:                 break
 8:             end if
 9:         end if
10:     end for
11: end for
12: return $\varphi'$
```

Fig. 7. Naive algorithm for finding the context with minimum cost.

Finally, if the algorithm takes into account also the presence of variables in the current φ', in the process of reducing μ_2 the order in which literals are processed in the loop of lines 4-9 would be $\neg R, \neg P, \neg Q, \neg S$, as S is already in φ' (because of μ_1). With this order, also μ_2 would be reduced to $\{\neg S\}$, and so in this case the final φ' would be $\neg S$, whose cost is only 4.

Efficiency of the algorithm The algorithm has the same complexity as the algorithm without costs, since we are only modifying the order in which we try to eliminate variables. We can therefore expect similar performance.

4.2 Finding an Optimal Solution and Reducing Multiple Dependent Contexts

The algorithm of Fig. 6 (and its greedy variant) computes *one* reduction for the input context formula, but it does not find (in general) the reduction with minimum cost. Clearly, finding such optimal context wrt. costs would be very desirable. However, solving this problem is far from trivial. A naive algorithm/solution for it is shown in Fig.7.

This algorithm works by enumerating up to *all* the formulas $\psi(S)$ that are equivalent to the context formula φ under the assumptions ξ, and picking the one with minimum cost. Such exhaustive enumeration is prohibitively costly: the outer loop of lines 2-10 is executed $2^{|V(\varphi)|}$ times, and, since the number of different boolean formulas over k variables is 2^{2^k}, the inner loop of lines 3-9 is executed up to $2^{2^{|S|}}$ times. Moreover, checking whether $\psi(S)$ and φ are equivalent under the assumptions ξ (line 4) is an NP-complete problem. Therefore, the naive algorithm would require to solve up to $\sum_{S \in 2^{V(\varphi)}} 2^{2^{|S|}}$ NP-complete problems.

In practice, the algorithm can be improved by performing a branch-and-bound search [14] (on the sum of costs of the variables) instead of enumerating all the subsets of variables, thus avoiding to enumerate (and check) formulas over variables whose cost is known to be higher than the best solution found so far. However, in the worst case the complexity would not improve.

The situation is even more complex if mutual dependencies among different contexts are taken into account when searching for the globally-optimal solution. As we observed in Section 3, in general reducing each context individually does not lead to a globally-optimal monitoring cost, and thus it would be desirable to reduce each context by taking the others into account. However, this would be very costly. A naive solution would be to collect all the possible reductions for each individual context (computed with the algorithm of Fig. 7), and then try all the possible combinations to pick the best one: if $\Phi_1 \ldots \Phi_n$ are the sets of all the reductions of the contexts $\varphi_1 \ldots \varphi_n$ under assumptions $\xi_1 \ldots \xi_n$, this would mean to try all the $\prod_{i=1}^{n} |\Phi_i|$ possible combinations.

5 Modeling Activities

We outline now the activities needed to create our proposed set of models:

– **Goal analysis**: in this activity, a context independent goal model is constructed. The purpose of goal analysis is to define goals and the interdependencies between system actors and to refine goals and elicit alternative sets of tasks that are executed in order to satisfy the high level goals. Moreover, in this activity the quality of each satisfaction alternative is studied and represented through its contribution to a set of non-functional requirements (softgoals).
– **Deciding contextual variation points**: we have already defined six context dependent variation points in Tropos goal model; variation points are the places where the relation between the goal and context models has to be specified. In this activity, the context dependent variation points of the goal model have to be decided, i.e. a decision if the point is context free or context dependent has to be taken.
– **Parameterizing goal model**: Tropos goal model does not provide constructs to specify the domain of discourse of goals, softgoals, and tasks. Since we need this specification to explicitly define what is common, and what is not, along different goals, softgoals, tasks, statement and/or facts, we propose the use of parameters to define and keep track of the domain of discourse of goal and context analysis. Deciding parameters is expected to get feedback from the analysis of the context and to be, therefore, iterative.
– **Context analysis**: in this activity, we need to top-down scan the goal model and define the context hierarchy at each contextual variation point. Checking the consistency and triviality can be done on the direct and accumulated contexts immediately, while optimization needs knowing the overall set of contexts to decide the less expensive set of facts that is sufficient to validate all contexts as we explained. This activity involves the analyst to specify the implications between leaf facts and running iteratively the automated reasoning we already explained. We still need supporting tools and more efficient reasoning for this activity, especially for specifying the implications, and optimizing the overall costs and not only the cost for each context aside as we provided.
– **Simulation and refinement** in this activity, we need to show the system alternatives in different contexts in order to agree with stakeholders on and to better understand the resulted contextual goal model. Given the set of all true leaf facts and

given the prioritarization over the softgoals, we just need to evaluate contexts to instantiate a traditional goal model, and then rank the resulted possible alternatives according to their contributions to the prioritized softgoals as already done in [15]. As an example from the goal model of Fig. 2, suppose that the current context allows for both of the alternatives for satisfying the goal *"assistance staff [s] gives piece of art [p] info to visitor [v]"*, and suppose that the context $C7$ is not valid (i.e. assistance staff is not in the same room as the visitor), and therefore the contextual contribution to the softgoal *"staff is more comfortable"* is negative. If the softgoal *"visitor is well informed"* is more preferred than the softgoal *"staff is more comfortable"*, then the alternative *"[s] gives [p] info to [v] by call"* will be excluded, while *"[s] gives [p] info to [v] in person"* is excluded in the opposite preference.

– **Extracting conceptual data model**: in this activity, and after having the final agreed upon contextual goal model, and the set of reduced and consistent context formulas, the analyst has to extract the conceptual data model by analyzing the facts of all of these context formulas. We need to keep the link between each fact and the data model fragment needed to verify it to allow for tracking the changes in the data model that result from any modification of the contextual goal model.

6 Related Work

The research in context modeling, (e.g., [16]), concerns finding modeling constructs to represent software and user context, but there is still a gap between the context model and software behavior model, i.e. between context and its use. We tried to reduce such gaps at the goal level and allow for answering questions like: "*how do we decide the relevant context?*", "*why do we need context?*" and "*how does context influence software and user behavior adaptation?*". Salifu et al. [17] investigate the use of problem descriptions to represent and analyze variability in context-aware software; the work recognizes the link between software requirements and context information as a basic step in designing context aware systems.

Software variability modeling, mainly feature models [18, 19], concerns modeling a variety of possible configurations of the software functionalities to allow for a systematic way of tailoring a product upon stakeholders choices, but there is still a gap between each functionality and the context where this functionality can or has to be adopted, the problem we tried to solve at the goal level. Furthermore, our work is in line, and has the potential to be integrated, with the work in [20] and the FARE method proposed in [21] that show possible ways to integrate features with domain goals and knowledge to help for eliciting and justifying features.

Requirements monitoring is about insertion of a code into a running system to gather information, mainly about the computational performance, and reason if the running system is always meeting its design objectives, and reconcile the system behavior to them if a deviation occurs [6]. The objective is to have more robust, maintainable, and self-evolving systems. In [22], a GORE (goal-oriented requirements engineer) framework KAOS [4] was integrated with an event-monitoring system (FLEA [23]) to provide an architecture that enables the runtime automated reconciliation between system goals and system behavior with respect to a priori anticipated or evolving changes of the system environment. Differently, we propose model-driven framework that concerns an earlier stage, i.e. requirements, with the focus on identifying requirements together with context, and eliciting the monitoring data.

Customizing goal models to fit to user skills and preferences was studied in [15, 24]. The selection between goal satisfaction alternatives is based on one dimension of context, i.e. user skills, related to the atomic goals (executable tasks) of the goal hierarchy, and on user preferences which are expressed over softgoals. Lapouchnian et al. [25] propose techniques to design autonomic software based on an extended goal modeling framework, but the relation with the context is not focused on. Liaskos et al [26], study the variability modeling under the requirements engineering perspective and propose a classification of the intentional variability when Or-decomposing a goal. We focused on context variability, i.e. the unintentional variability, which influences the applicability and appropriateness of each goal satisfaction alternative. Reasoning with Tropos goal model has been already studied in [27]; adding context to goal models creates the need to integrate between reasoning with context and that with the goal model.

7 Conclusions and Future Work

In this paper, we have proposed a goal-oriented framework for modeling and analyzing contextual requirements. We adopted Tropos goal analysis to identify requirements and proposed the association between its variation points and context. In turn, context is defined through a hierarchial analysis that elicits alternative sets of facts the system has to verify on monitorable data so as to confirm the high level contexts. Analyzing facts will also lead to identify the data conceptual model the monitoring system has to instantiate to enable facts verification. We have also shown a set of activities that are followed to build our proposed set of models.

The direct and accumulated contexts at the contextual goal model might be easily redundant, which implies a redundancy in the monitoring requirements, or even inconsistent which furthermore denies the adoptability of the software functionalities preconditioned by them. We have articulated the context interaction problem, that is analogous to and complex as feature interaction problem [9], showed its effects and main difficulties, and proposed a SAT-based reasoning to check the consistency, triviality, and producing an equivalent context with reduced cost.

Extending and optimizing the reasoning we have proposed and integrating it with the previous work in reasoning on traditional Tropos goal model in [27] is a goal we will try to reach. We also want to develop a supporting tool for our framework that assists the analysts for building correctly our proposed models and simulating the system behavior. Moreover, we will work on complex case studies in order to better validate the proposed framework and refine it and make it more feasible.

Acknowledgement

This work has been partially funded by EU Commission, through the SERENITY, and COMPAS projects, and by the PRIN program of MIUR under the MEnSA project. We would also like to thank Prof. Jaelson Brelaz de Castro, Prof. Roberto Sebastiani, and Prof. Alex Borgida for the valuable discussion we had about this work.

References

1. Finkelstein, A., and Savigni, A.: A framework for requirements engineering for context-aware services. In: Proceedings of the 1st Int. Workshop on From Software Requirements to Architectures (STRAW), 2001
2. Yu, E.: Modelling strategic relationships for process reengineering. Ph.D. Thesis, University of Toronto (1995)
3. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents andMulti-Agent Systems 8(3) (2004) 203-236
4. Dardenne, A., Van Lamsweerde, A., Fickas, S.: Goal-directed requirements acquisition. Science of computer programming 20(1-2) (1993) 350
5. Mylopoulos, J., Chung, L., Yu, E. From object-oriented to goal-oriented requirements analysis. Commun. ACM, ACM, 1999, 42, 31-37
6. Fickas, S., Feather, M.: Requirements monitoring in dynamic environments. In: Proceedings of the Second IEEE International Symposium on Requirements Engineering, IEEE Computer Society Washington, DC, USA (1995) 140
7. Ali, R., Dalpiaz, F., Giorgini, P.: Goal-Based Self-Contextualization. In Proc. of the Forum of the 21st International Conference on Advanced Information Systems (CAiSE09 - Forum).
8. Ali, R., Dalpiaz, F., Giorgini, P.: A Goal Modeling Framework for Self-Contextualizable Software. In the 14th Intl. Conf. on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD09). LNBIP 29-0326, pp. 326–338. Springer (2009).
9. Cameron, E.J., Griffeth, N., Lin, Y.-J., Nilson, M.E., Schnure, W.K., Velthuijsen, H.: ”A feature-interaction benchmark for IN and beyond,” Communications Magazine, IEEE , vol.31, no.3, pp.64-69, Mar 1993
10. Ali, R., Dalpiaz, F., Giorgini, P.: Location-based variability for mobile information systems. In: Bellahs‘ene, Z., Leonard,M. (eds.) CAiSE 2008. LNCS, vol. 5074, pp. 575-578. Springer, Heidelberg (2008)
11. Ali, R., Dalpiaz, F., Giorgini, P.: Location-based software modeling and analysis: Tropos-based approach. In: Li, Q., Spaccapietra, S., Yu, E., Olive, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 169-182. Springer, Heidelberg (2008)
12. Strang, T., Linnhoff-Popien, C. A context modeling survey. Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp, 2004
13. Eén, N., Sörensson, N.: An Extensible SAT-solver. In Giunchiglia, E., Tacchella, A., eds.: Theory and Applications of Satisfiability Testing, 6th International Conference, SAT 2003. Volume 2919 of Lecture Notes in Computer Science., Springer (2004) 502–508
14. Hillier, F. S., Lieberman, G. J.: Introduction to Operations Research. McGraw-Hill: Boston MA; 8th. (International) Edition, 2005.
15. Hui, B., Liaskos, S., Mylopoulos, J.: Requirements analysis for customizable software goals-skills- preferences framework. In: RE, IEEE Computer Society (2003) 117-126
16. Henricksen, K., Indulska, J.: A software engineering framework for context-aware pervasive computing. In: Proc. Second IEEE Intl. Conference on Pervasive Computing and Communications (PerCom04). (2004) 77
17. Salifu, M., Yu, Y., Nuseibeh, B. Specifying Monitoring and Switching Problems in Context Proc. 15th Intl. Conference on Requirements Engineering (RE’07), 2007, 211-220
18. Pohl, K., Bockle, G., van der Linden, F.: Software Product Line Engineering: Foundations, Principles, and Techniques. Springer (2005)
19. Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E., Huh, M.: Form: A feature-oriented reuse method with domain-specific reference architectures. Ann. Softw. Eng. 5 (1998) 143168

20. Yu, Y., do Prado Leite, J.C.S., Lapouchnian, A., Mylopoulos, J.: Configuring features with stakeholder goals. In: SAC 08: Proceedings of the 2008 ACM symposium on Applied computing, New York, NY, USA, ACM (2008) 645649
21. Ramachandran, M., Allen, P.: Commonality and variability analysis in industrial practice for product line improvement. Software Process: Improvement and Practice 10(1) (2005) 3140
22. Feather, M. S., Fickas, S., Lamsweerde, A. V., Ponsard, C. Reconciling System Requirements and Runtime Behavior. Proceedings of the 9th international workshop on Software specification and design IWSSD '98, IEEE Computer Society, 1998, 50
23. Cohen, D., Feather, M. S., Narayanaswamy, K., Fickas, S. S. Automatic monitoring of software requirements ICSE '97: Proceedings of the 19th international conference on Software engineering, ACM, 1997, 602-603
24. Liaskos, S., McIlraith, S., Mylopoulos, J.: Representing and reasoning with preference requirements using goals. Technical report, University of Toronto (2006).
25. Lapouchnian, A., Yu, Y., Liaskos, S., Mylopoulos, J.: Requirements-driven design of autonomic application software. In: Proc. 2006 conference of the Center for Advanced Studies on Collaborative research (CASCON 06), ACM (2006) 7
26. Liaskos, S., Lapouchnian, A., Yu, Y., Yu, E., Mylopoulos, J.: On goal-based variability acquisition and analysis. In: Proc. 14th IEEE Intl. Requirements Engineering Conference (RE06). (2006) 76-85
27. Giorgini, P.,Mylopoulos, J., Nicchiarelli, E., Sebastiani, R.: Reasoning with goal models. In Spaccapietra, S., March, S.T., Kambayashi, Y., eds.: ER 2002. Volume 2503 of Lecture Notes in Computer Science., Springer (2002) 167181