View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Repositorio Institucional de la Universidad de Alicante

Adding Semantic Modules to improve Goal-Oriented Analysis of Data Warehouses using I-star

Alejandro Maté[a,*], Juan Trujillo[a], Xavier Franch[b]

[a]*Lucentia Research Group, Department of Software and Computing Systems, University of Alicante, Carretera San Vicente del Raspeig s/n - 03690 San Vicente del Raspeig - Alicante, Spain*
[b]*BarcelonaTech, Universitat Politècnica de Catalunya, Calle Jordi Girona, 31 - 08034 Barcelona, Spain*

Abstract

The success rate of data warehouse (DW) development is improved by performing a requirements elicitation stage in which the users' needs are modeled. Currently, among the different proposals for modeling requirements, there is a special focus on goal-oriented models, and in particular on the i* framework. In order to adapt this framework for DW development, we previously developed a UML profile for DWs. However, as the general i* framework, the proposal lacks modularity. This has a specially negative impact for DW development, since DW requirement models tend to include a huge number of elements with crossed relationships between them. In turn, the readability of the models is decreased, harming their utility and increasing the error rate and development time. In this paper, we propose an extension of our i* profile for DWs considering the modularization of goals. We provide a set of guidelines in order to correctly apply our proposal. Furthermore, we have performed an experiment in order to assess the validity our proposal. The benefits of our proposal are an increase in the modularity and scalability of the models which, in turn, increases the error correction capability, and makes complex models easier to understand by DW developers and non expert users.

Keywords: Data Warehouses, modules, user requirements, i-star

1. Introduction

Organizations manage huge amounts of information, and wish to take informed decisions by using that information. Nowadays, there is an increasing importance of the Business Intelligence (BI) in the enterprise environment. In

---

*Corresponding author. Tel: +34 96 5909581 ext. 2737; fax: +34 96 5909326
Email addresses: `amate@dlsi.ua.es` (Alejandro Maté), `jtrujillo@dlsi.ua.es` (Juan Trujillo), `franch@essi.upc.edu` (Xavier Franch)

fact, the Gartner Group showed that, during the recent recession period, the BI market not only did not decrease, but instead it grew a 4% [1].

At the core of the BI, among other technologies, is the Data Warehouse (DW). DWs integrate several heterogeneous data sources in multidimensional structures (i.e. facts and dimensions) in support of the decision-making process [2, 3]. Therefore, the development of the DW is a complex process which must be carefully planned in order to meet user needs. This process can be even more complex, if we consider that requirements for the DW change as the organization's information needs change. For this reason, the modeling of user needs is a very important aspect of DWs, which can be accomplished by means of goal-oriented models. These models represent the users' intentions in a requirements model using goals and are easily understandable by users. Among the goal-oriented approaches, the i* framework [4], is currently one of the most widespread goal modeling frameworks. This framework has been applied for modeling organizations and system requirements among others.

However, due to the idiosyncrasy of DWs, a specialization of the i* framework was required, in order to correctly model the desired information goals. In our previous work, we presented the required specialization, along with our development methodology for DWs. In our proposal, we follow the Model Driven Architecture (MDA) [5], starting from a Computation Independent Model (CIM) layer where requirements are modeled. From this layer, the DW schema is derived into a Platform Independent Model (PIM) layer, reconciliated with the information present in the data sources, and derived into its implementation. Therefore, the CIM layer is crucial, since it acts as the starting point of the process.

Nevertheless, as pointed in [6], the i* framework lacks scalability due to the absence of modularity. Modularity is a well-known concept in software engineering. As far as the start of the 70s, modular programming became a hot topic and the benefits of splitting complexity using some well-defined criteria were subject of several seminal papers [7]. Afterwards, modularity spread over other life-cycle activities and artifacts, and became very popular especially in the context of system design, where the notion of decomposing a system into its parts offers several benefits like better flexibility, management and testability, to name a few. Since we are interested in modularity applied to specification models, it can be defined as the ability to decompose a large model into several sub-models, such that they independently have a well-defined meaning, and whose combination solves the original problem.

Since the work presented in [8] is a specialization of the original framework, it lacks modularity as well. As DW requirements models may become very complex, this hurts their readability and comprehension, becoming more difficult to correct and update as requirements change. We have experienced this drawback ourselves, as some of our real projects had over 16 goals, 15 tasks and 53 resources for a single actor. These models became huge for correction and communication with the users. Sometimes these models even included repeated DW elements in the same model with different structure, since designers forgot which elements were already defined. Therefore, it is important to improve this aspect in order to manage corrections and changes in DW requirements in an easier way.

In the short version of this paper [9] we proposed an extension of our i* profile [8], in order to adapt it and improve its modularity. In turn, this increases its manageability, as well as the comprehension capability of the user when dealing with complex models. With these modifications, the communication between users and developers is improved, leading to higher success rates. We also provided a set of guidelines to correctly apply the proposal. Moreover, we performed an experiment in order to assess the validity our proposal.

In addition, in this long improved version, we (i) include the definitions of the main goals (strategy, decision, and tactic) in which we classify the final user's needs, (ii) perform an ontology mapping between concepts in the i* framework and the DW context, and include *Decision*, *Information*, and *Hierarchy* modules, increasing the scalability of the models, (iii) include an extended case study in order to show the applicability and benefits of applying our proposal, as well as (iv) describe a second, deeper analysis of the results, reaching new conclusions to better define and organize the modules.

The rest of the paper is structured as follows. Section 2 presents the related work in this area. Section 3 presents our i* profile for DWs. Section 4 proposes the different types of modules for our i* profile. Section 5 presents an example of application and the experiment performed. Finally, Section 6 summarizes the conclusions and future work.

Basic knowledge of i* is assumed in the paper, see [4] and the i* wiki (http://istar.rwth-aachen.de) for a thorough presentation.

2. Related Work and Background

Scalability is probably the best-known and widely acknowledged problem of i*. It is fact that i* models quickly grow in size (see [10] for an illustrative example of large-scale model) making them rapidly difficult to manage. As argued by [11], the scalability problem is a direct consequence of the lack of mechanisms for modularization. In that work, the authors conducted an empirical study on different aspects related to i* as a modeling language, and it was concluded that modularity is not supported in i*, consequently we may say that scalability is not supported either. Since the core of the language has not evolved since then, the problem persists nowadays.

When dealing with scalability issues, other works have focused on i* modularity in general, like the one in [6]. However, these modules do not have meaningful semantics for being applied in DWs, which could favor the understandability of the modularization process. Therefore, before performing any kind of adaptation, a study of the target domain must be performed along with a mapping between the concepts. Then, the necessary modules should be defined accordingly to how the target domain is structured. In other areas, a similar approach has been applied successfully in order to improve the scalability of the models. For example, in [12] the authors introduce new elements in the notation of task models that summarize several elements in the diagrams, allowing the designers to manage their complexity while keeping the models meaningful.

Within the area of DW requirements modeling, initial works such as [13] propose to represent DW requirements by means of use case diagrams. Use cases divide DW requirements into an actor dependency diagram and several use case specifications. However, use case notation is difficult for users to understand. Therefore, more recent works focus on representing DW requirements in terms of goals, both i* based, such as [8, 14], and non-i* based [15]. The i* based approaches suffer from the lack of modularity intrinsic to the i* core, as they do not provide any mechanisms to control the complexity and size of the diagrams. Unfortunately, non-i* based models do not include any modularization elements either, thus the complexity and size of the diagrams is only determined by the complexity of user requirements themselves.

In our previous work, we developed a UML profile for modeling DWs at conceptual level [16], where the importance of packages was shown, in order to improve the modularity of DW conceptual models. The packages included were StarPackage, for differentiating cubes, DimensionPackage, for aggregating dimensions along with their hierarchies, and FactPackage, which included the associated fact. These packages allow the developer to analyze the model at different levels of detail, hiding those elements on which he has no interest, lowering the complexity of the model and increasing its readability. In turn, this aspect makes the developing of the schemata less error prone.

However, since the conceptual level is closer to developers than to users, we required models with a higher level of abstraction in the development process. Therefore, in order to improve the communication with the users, and increase the success rate of DW projects, we included a RE phase in our methodology [8]. In this RE phase, requirements are captured on a model by using a UML profile [8] based on the i* framework [4]. From these requirements, the conceptual model is automatically derived by means of Model Driven transformations [17], transforming the different Business Process, Contexts, and Measures associated with the goals at requirements level into Facts, Dimensions and Measures at the conceptual level.

Nevertheless, although our i* profile incorporated the necessary semantics and methodology, it lacks any kind of modularity. In turn, this hurts the communication with the users, since complex requirements models can become huge and difficult to read and understand. Now, in this work, we complement our approach, by improving the modularity and scalability of our i* profile. We include modules for the decision and information goals, as well as for hierarchies of contexts. By improving the modularity, the models are easier to read, which, in turn, reduces the error rate and increases user satisfaction.

3. i* Profile for DWs

Our i* profile, presented in [8], follows a Goal-Oriented Requirements Engineering (GORE) approach. GORE is concerned about modeling goals, thus obtaining user requirements by following a refinement process [18]. The i* modeling framework [4] provides mechanisms to represent actors, their dependencies, and structuring the business goals that organization pretends to achieve. This framework establishes two models: the strategic dependency (SD) model for describing the dependency relationships among various actors in an organizational context, and the *strategic rationale* (SR) model, used to describe actor interests and concerns, and how they might be addressed. From now on, we focus on the SR models to model goals and information requirements of decision makers.

The first step is aligning the ontology of i* with the target domain. In the DW domain, the requirements model specifies the informational needs of different stakeholders in order to support the decision-making process. This information is used to improve the performance of a business activity. In our observation, several types of goals which arise naturally during the design process.

- **Strategic goals**. They represent a desired change from a current situation into a future one. A strategic goal is always related to the main objectives of the business process (see below) that is being improved. Therefore, strategic goals always have an objective to be met, either clear, i.e. Sales increased, or fuzzy, i.e. Number of clients significantly increased. They are long-term goals that cause an immediate benefit for the organization when fulfilled.

- **Decision goals**. They operationalize strategic goals into appropriate actions by answering the question: "how can a strategic goal be achieved?". Decision goals represent decisions that make use of information in order to provide a benefit for the organization. Decisions can be described either in terms of objectives, i.e. Some kind of promotion offered, or in terms of tasks, i.e. Open new stores. The benefit obtained by a decision goal is directly related to the achievement strategic goals by means of the decision goal.

- **Information goals**. They identify the information required for a decision goal to be achieved by answering the question: "how can decision goals be achieved in terms of information required?". Information goals specify the necessary information to be gathered, typically by means of an analysis. Therefore, they can be defined in terms of goals, i.e. Customer purchases analysed, or in terms of the analysis process, i.e. Examine the stocks daily. The satisfaction of information goals allows decision makers to take decisions and fulfill decision goals.

These three types of goals have a decreasing level of abstraction, from strategic (most) to informational (less). In addition, there is a contextualization relationship among goals: decision goals only take place within the context of strategic goals, and informational goals only take place inside the context of decisional goals.

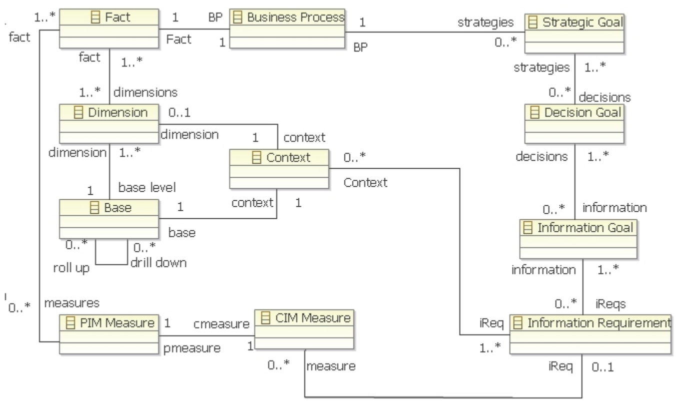


Figure 1: DW domain metamodel

Along with these goals, the DW domain includes several key concepts related to the multidimensional level of DWs [19]. The description of these concepts, presented in Figure 1, is as follows:

- **Business Processes**. Represent an activity that the user wishes to improve by means of strategies. These business processes have associated a series of performance indicators, represented as measures of the business process. Sometimes business processes can be described in terms of the goal pursured by the activity, i.e. Contracts agreed, or in terms of the activity itself, i.e. Make sales.

- **Information Requirements**. Represent the necessary information in order to achieve an information goal. They are always considered in terms of information gathering tasks. Information requirements are decomposed into context and measures, that represent the information to be gathered by the information requirement.

- **Contexts**. Describe the necessary additional data in order to analyze a given business process. They represent information about entities involved in the business processes of the organization, i.e. Department or Customer, and can be grouped into hiearchies, i.e. Customers within the same city.

- **CIM measures**. Represent indicators of performance of a business process. They provide quantitative information that can be assessed by decision makers in order to evaluate if business processes are performing as expected. PIM measures are the multidimensional counterpart of indicators.

- **Bases**. Represent the multidimensional counterpart of contexts. They describe the levels of aggregation within a dimension of the data warehouse.

Table 1: Alignment of the DW concepts with the i* framework

DW	i* concept	Example
Strategic goal	Goal	Sales increased
	Softgoal	Number of clients significantly increased
Decision goal	Goal	Some kind of promotion offered
	Task	Open new stores
Information goal	Goal	Customer purchases analysed
	Task	Examine stocks daily
Business Process	Goal Task	Contracts agreed Make sales
Information Requirement	Task	Record task assignments and durations
Context	Resource	Market; Department
Measure	Resource	Discount; Income generated

- **Dimensions**. Represent a context of analysis to analyze a fact, and are formed by sets of hierarchies. Each hierarchy can have one or more groups of bases, forming classification and generalization hierarchies in the multidimensional model and defining the structure of the data warehouse.
- **Facts**. Represent the multidimensional counterpart of the business process which wants to be improved.

After having presented the target domain concepts, we proceed to map the DW concepts with the i* ontology, as shown in Table 1. As can be perceived, not all the elements are aligned. Specifically, the concepts of dimension, fact, base, and PIM Measures, are not considered part of the requirements engineering process, thus they are left aside as external elements. Moreover, some DW concepts can be mapped into more than one i* intentional type depending on the level of abstraction and the cut criterion chosen.

Once we have defined and mapped the concepts in our i* profile, we will describe them through an example, shown in Figure 2. In this example, we start the requirements analysis from a business process (BP), related to the decision-maker. The BP, which is the center of the analysis, models an activity of interest for the decision-maker. In this case the activity is to *Make Contracts*, and has associated a series of strategic goals, aimed to improve the business performance. Strategic goals represent the highest level of abstraction. They are thought as changes from a current situation into a better one in terms of business process objectives. In our case, the strategic goals associated with the BP are *Cost of contracts minimized* and *Quality of workers increased.* Other examples of strategic goals would be *Increase sales*, *Increase number of customers*, *Decrease cost*, etc. Their fulfillment causes an immediate benefit for the organization.

In order to achieve these strategic goals, there are a series of decision goals that must be met. Decision goals represent the medium level of abstraction in our SR models. They try to answer the question "how can a strategic goal be

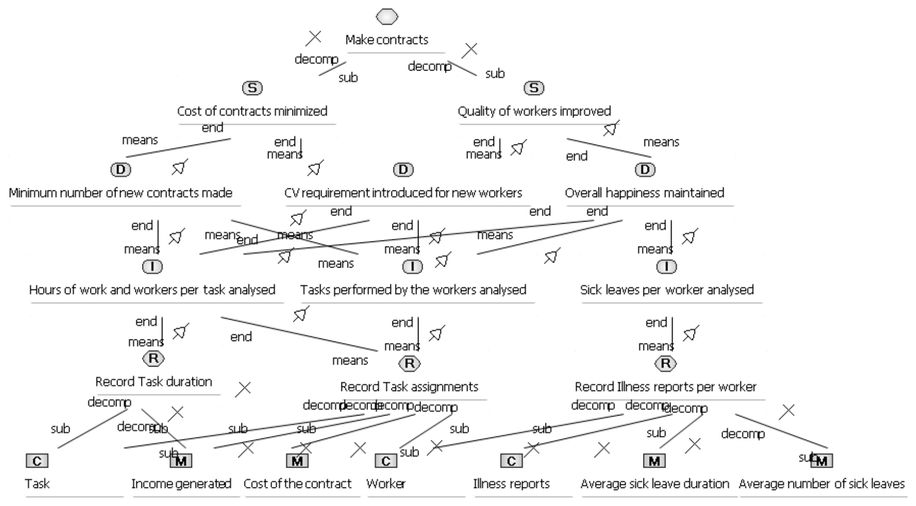

Make contracts
Cost of contracts minimized
Quality of workers improved
Minimum number of new contracts made
CV requirement introduced for new workers
Overall happiness maintained
Hours of work and workers per task analysed
Tasks performed by the workers analysed
Sick leaves per worker analysed
Record Task duration
Record Task assignments
Record Illness reports per worker
Task
Income generated
Cost of the contract
Worker
Illness reports
Average sick leave duration
Average number of sick leaves

Figure 2: Example of the current monolithic CIM representation

achieved?", and they aim to take the appropriate actions to fulfill a strategic goal. They are related to strategic goals by intentional means-end relationships. In our example, in order to achieve *Cost of contracts minimized*, it has been decided that it is necessary to have the *Minimum number of new contracts made* as well as have a *CV requirement introduced for new workers*, in order to achieve the strategic goal. However, decision goals can affect more than one strategic goal. In our case, the last decision goal is related with *Quality of workers increased* strategic goal as well, since the CV affects the quality of the new workers being employed. Other examples of decision goals would be *Determine some kind of promotion* or *Open new stores*. Their fulfillment only causes a benefit for the organization if it helps to reach strategic goals, since decision goals only take place within the context of strategic goals.

As with the strategic goals, the decision goals can be achieved by having the necessary information available. This required information is modeled by means of the informational goals. Information goals represent the lowest level of abstraction. They try to answer the question: "how can decision goals be achieved in terms of information required?", and they are related to the information required by a decision goal to be achieved. In our example, the information required is *Hours of work and workers per task analysed* and *Tasks performed by the workers analysed* for each decision goal, whereas the information about *Sick leaves per worker analysed* affects only the *Overall happiness maintained* decision goal. Other examples of information goals are *Analyze customer purchases* or *Examine stocks*. Their fulfillment helps to achieve decision goals and they only happen within the context of decision goals.

Finally, informational goals are achieved by means of information requirements. In our case, we need to *Record Task duration*, *Record Task assignments*, and *Record Illness reports per worker* in order to gather the required informa-

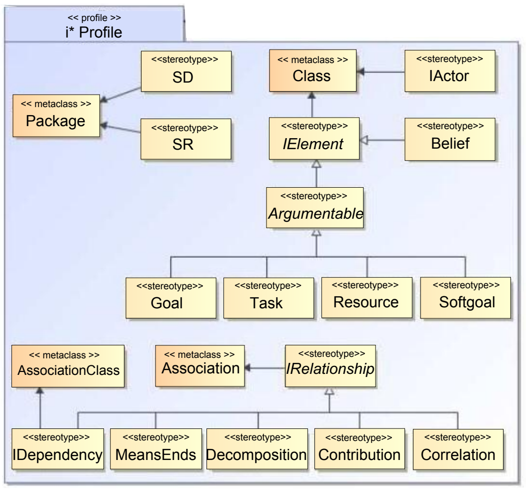


Figure 3: Original i* elements

tion. Each of these requirements is decomposed into contexts and measures. The example includes the *Task, Worker*, and *Illness report* contexts as well as the *Income generated, Average sick leave duration* and *Average number of sick leaves* measures, which determine the performance of the business process.

As has been shown in the example, a lower-level goal can be a part of different higher-level goals. This process is repeated at all levels, leading to highly interrelated elements in the model, making difficult to comprehend the business strategy in huge models. In order to solve this issue, we propose a series of modules, packaging all the elements related to a given higher-level goal on each module.

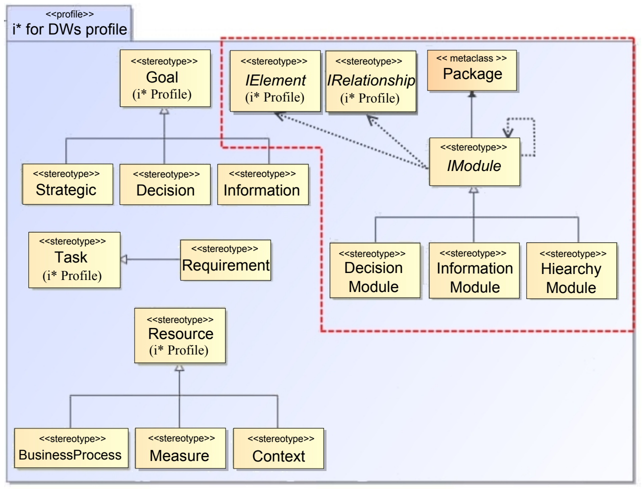
4. Definition of Modules and Guidelines

In this section, we will present the extension to our i* profile for DWs, by defining the proposed modules and the extended metaclasses. Furthermore, we will also present some guidelines to the application of the modules proposed.

4.1. Definition of Modules

First, we will define our proposed modules, in order to manage the complexity of the goal models. The modules which we will define are strongly related to the concepts identified in the DW domain. Therefore, each module has a specific semantic associated adapted for the DWs. We have not included a module for strategic goals since typically there is only a few of them.

- **Decision modules** include the elements related to a given *decision goal.* They can include *decision goals, information goals, requirements, contexts, measures, other decision modules, information modules,* and *hierar-*



<<profile>>
i* for DWs profile
<<stereotype>> Goal (i* Profile)
<<stereotype>> Strategic
<<stereotype>> Decision
<<stereotype>> Information
<<stereotype>> Task (i* Profile)
<<stereotype>> Requirement
<<stereotype>> Resource (i* Profile)
<<stereotype>> BusinessProcess
<<stereotype>> Measure
<<stereotype>> Context
<<stereotype>> IElement (i* Profile)
<<stereotype>> IRelationship (i* Profile)
<< metaclass >> Package
<<stereotype>> IModule
<<stereotype>> Decision Module
<<stereotype>> Information Module
<<stereotype>> Hiearchy Module

Figure 4: i* profile with modules extension for DW

chy modules. They contain all the necessary information to take a given decision, which helps achieving a *strategic goal*.

- **Information modules** include the elements related to a given information goal. They can include *information goals, requirements, contexts, measures, other information modules,* and *hierarchy modules*. They aggregate all the information which is necessary to satisfy a given information goal.

- **Hierarchy modules** include the elements which constitute a hierarchy. They are formed by the different contexts which represent the different levels of aggregation of a dimension. They can only include *contexts*. These modules help with the reusability of the dimensions at the requirements level, and hide the complexity of hierarchies when it is unnecessary.

These modules are shown in Figure 4 together with the rest of the elements in the i* for DWs profile. The modules defined are loosely coupled with the core i* elements, shown in Figure 3, and extend from the *Package* element. Moreover, they include an intermediate element, *iModule*, in order to help with the definition of OCL constraints that guarantee their correct application in CASE tools. After having defined the modules, we will present a set of guidelines to apply them while minimizing the drawbacks.

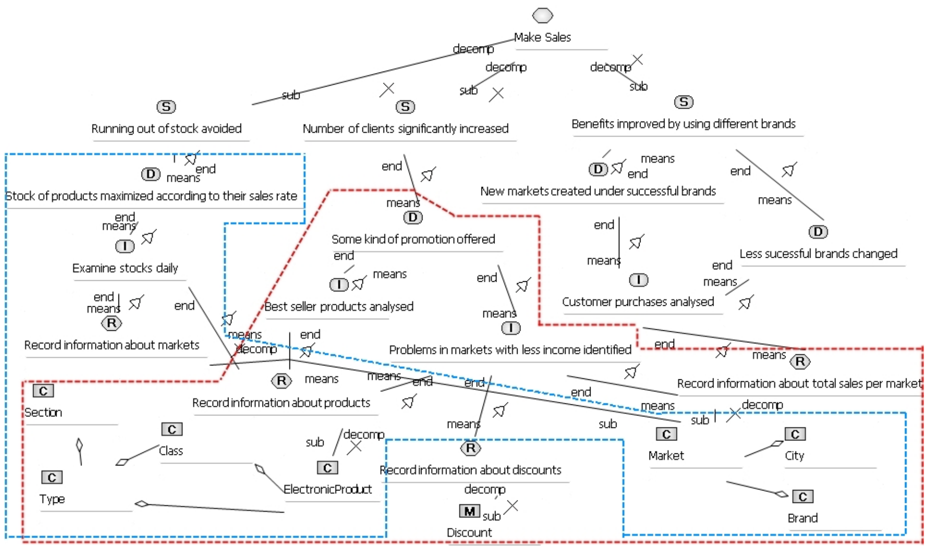
4.2. Guidelines

In this section, we will give some guidelines to use the provided modules, in order to maximize their benefits. It is not mandatory to package every element, although it is recommended for the sake of uniformity and to provide different abstraction levels, which results in a more intuitive approach (G1). However, if some parts of the goal tree have a low complexity, it might not be necessary to group them in a separate package (G2). For each package created, there should be a single root element, corresponding to the type of package, which acts as a connection for higher level elements. This element should have no dependencies to other elements inside the same package (G3). The name of the package should be the same as the root element, in order to help with the identification of the corresponding packaged subtree (G4). For each decision goal a *Decision module* should be created (G5). Inside a decision module there should be an *Information module* for each information goal that supports the decision goal (G6). If included in a CASE tool, elements should not be repeated, but instead imported from packages where they were first defined (G7). *Information modules* should include all the elements related to the information goal, importing elements where necessary, and always including a *Hierarchy module* for each different hierarchy of contexts present (G8). These *Hierarchy modules* represent the lowest level of abstraction in the strategic rationale, and should be always separated from the goal tree, in order to hide the details of the hierarchies of contexts unless they are necessary (G9).

4.3. Improvements in Scalability

In order to demonstrate the improvements in scalability obtained with the introduction of modules it is first necessary to define the concept of scalability. Scalability is a term often used intuitively, but with no clear definition. When used in the context of software engineering notations and diagrams [20], it usually refers to "the property of reducing or increasing the scope of methods, processes, and management according to the problem size [...] Inherent in this idea is that software engineering techniques should provide good mechanisms for partitioning, composition, and visibility control. It includes the ability to scale the notation to particular problem needs, contractual requirements, or even to budgetary and business goals and objectives." In practice, scalability problems arise typically when models become too large to be handled adequately, as shown in the previous sections.

In the particular case of i*, inherited by i* for DWs, the lack of modularization elements limits the capability of the designer to partition the model, control the visibility of elements, and scale the notation to particular needs, i.e. decision maker's goals vs. data warehouse structural requirements. The modules proposed in this section enable the designer to (i) perform partitioning, dividing a single diagram into multiple ones, thus reducing the visual complexity of each individual sub-diagram, (ii) control the visibility of unnecessary elements, by hiding lower abstraction goals and their structure by means of packages, and (iii) manage the scope of the diagrams, by separating decision maker's goals from data warehouse structural requirements (contexts and measures) derived from these goals.



Make Sales
decomp
decomp
decomp
sub
sub
sub
Running out of stock avoided
Number of clients significantly increased
Benefits improved by using different brands
Stock of products maximized according to their sales rate
New markets created under successful brands
Some kind of promotion offered
Less sucessful brands changed
Examine stocks daily
Best seller products analysed
Customer purchases analysed
Record information about markets
Problems in markets with less income identified
Record information about products
Record information about total sales per market
Section
Class
ElectronicProduct
Record information about discounts
Market
City
Type
Discount
Brand
means
end

Figure 5: Part of the requirements model for Sales analysis with the scope of the decision goal *Some promotion offered* marked in red, and the scope of *Stock of products* in blue

5. Example of Application and Experiment Results

In this section we will present the application of our proposal to an example, as well as the results of two experiments performed in order to analyze how users and developers perceive the modularized models.

5.1. Example of Application

The following example presents a simpler goal tree, as opposed to the one presented in Section 3, whereas the contexts and hierarchies are better defined at requirements level than previously, and the scope of each element may be hard to identify. In this case, the contexts can be aggregated at different levels of detail, presenting market and electronic product contexts as the lowest level, which can be aggregated up to state and section levels.

This example can be modularized using the proposed packages, decreasing its complexity and providing different levels of detail. In this sense, the application of modules results in a first level providing an overview of the strategies related to the business process and their corresponding decisions. In this case, the goal tree presents the different decision packages as its leaves, which are further detailed in their corresponding models. Figure 6 presents the previous business process with 3 related strategies and the corresponding 3 decision packages.

For each decision, we have a different package which includes their related information goals and presents the intermediate level of detail. The elements corresponding to each information goal are also modularized in their own packages, which are the leaves of the decision models.

Finally, for each information goal, we have a package which includes their related information requirements, presenting the corresponding hierarchy packages and measures. Figure 7 presents a information goal with 3 different information requirements, which account for a total of 2 measures and 2 hierarchies.

In this model, hierarchies are included at CIM level, but are separated into their own packages. This hides the lowest level of detail, which acts as a bridge between the requirements and conceptual models, whenever it is necessary, allowing us to focus on the modeling of the goals and their related elements.

5.2. Experiment Results

We have performed two experiments, with participants ranging from non-expert people to DW designers and experts on i* modeling, in order to evaluate the impact of our proposal. These experiments are part of a family of experiments for assessing the validity and impact of the proposal, following the same methodology as in [21]. There were two rounds of experiments. The first round presented two examples, one smaller (see Figure 2, *Example 1*) and one bigger (see figure 5, *Example 2*). Both examples were presented in generic i* notation with our own stereotypes, in order to make the questionnaires more accessible for all the participants. Monolithic models were presented in a single sheet, whereas modularized models were presented in multiple sheets. Both base examples were small in comparison with real project models, presenting fewer goals and contexts in order to make them manageable. These examples were presented in the four combinations:

- Questionnaire 1.A: Example 1 without modules, Example 2 with modules.
- Questionnaire 1.B: Example 2 with modules, Example 1 without modules.
- Questionnaire 2.A: Example 2 without modules, Example 1 with modules.
- Questionnaire 2.B: Example 1 with modules, Example 2 without modules.

A total of 28 participants filled the questionnaire. Each participant was given one kind of questionnaire and they were asked to identify and mark a series of elements on each model (which were the same for both modularized

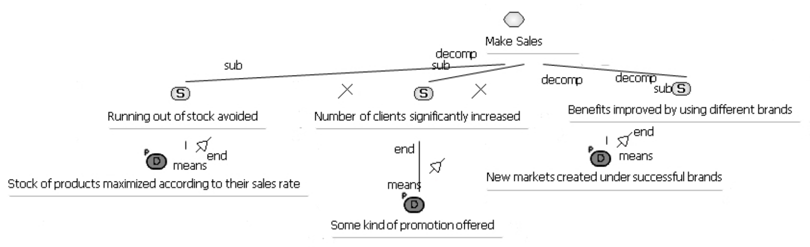

Make Sales
decomp
sub
sub
decomp
decomp
sub
S
Running out of stock avoided
S
Number of clients significantly increased
S
Benefits improved by using different brands
end
means
D
Stock of products maximized according to their sales rate
end
means
D
Some kind of promotion offered
end
means
D
New markets created under successful brands

Figure 6: Strategy level for Sales analysis

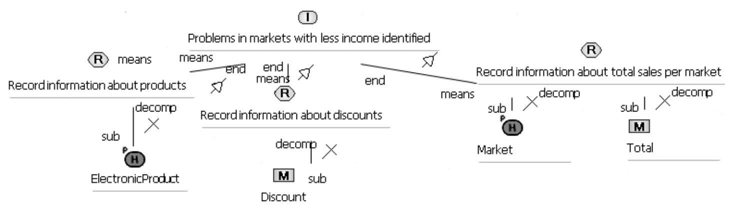

Problems in markets with less income identified
means
Record information about products
means
end
end
means
Record information about discounts
end
means
Record information about total sales per market
decomp
sub
ElectronicProduct
decomp
sub
Discount
sub
decomp
Market
sub
decomp
Total

Figure 7: Information level for Sales analysis

and monolithic versions of the same example). The participants were not able to modify their answer once they finished a question. After completing the identification tasks on each model they were asked to give scores for a series of characteristics of the model, ranging from 0 to 3. Finally, after having finished identifying elements in both models, they were asked abstract questions about how they would add a new element at different levels (decision goal, information goal and a set of contexts), while not referencing any example. The group of participants was formed by 14 self-evaluated beginners, 8 participants with some experience and 6 participants were advanced users/experts. Regarding the accumulated experience, 17 participants had less than 1 year of experience in the i* framework, 8 participants had between 1 and 5 years of experience and 3 participants had over 5 years of experience. Given this sample, the hypothesis for our experiment are:

Null hypothesis, H_{0_1}: There is no statistically significant correlation between the modularization of models and the time required for different tasks and the characteristics perceived.

Hypothesis H_{1_1}: $\neg H_{0_1}$

The independent variables in the experiment are those whose effects should be evaluated. In our experiment, this variable corresponds to how the model is structured (modularized or monolithic). On the other hand, dependent variables for the experiment are the understandability and manageability of the models, evaluated accordingly to the time necessary to perform different tasks on the models. As there was no statistically significant correlation between the structure of the models and most tasks, we did not calculate further values like efficiency and effectiveness. Therefore, the results will be discussed in terms of trends. The experiment results for the first round are shown in Table 2. Time is measured in seconds.

In order to obtain the results, in every step, first, outliers were identified and filtered. Then, the second step was to perform a variance analysis of the data (ANOVA), in order to identify significant differences between the models. After the first step, 27 questionnaires were left, which were used for the statistical analysis. The significance analysis ($\rho < 0.05$) revealed that the reading time for the Sales model was significantly different (inferior) than when built in a monolithic way. The only other significant difference perceived was the scalability of both examples, which had a notably increase.

Table 2: Tasks performed (left) and independent (top) variables for experiment 1

	Monolithic	Modularized	ρ
Avg. reading time Sales	299.31	210.31	0.037
Identif. task 1 Sales	190.08	278.62	0.074
Identif. task 2 Sales	190.94	165.08	0.396
Avg. reading time Contracts	162.73	181.33	0.576
Identif. task 1 Contracts	150.07	211.5	0.112
Identif. task 2 Contracts	124.33	161.00	0.096
Avg. errors per questionnaire Sales	0.82	0.47	0.247
Avg. errors per questionnaire Contracts	0.33	0.36	0.906
Readability score Sales	2	1,93	0.826
Scalability score Sales	1,41	2,26	0.016
Comprehension score Sales	1,5	1,87	0.229
Modifiability score Sales	1,5	2,06	0.079
Readability score Contracts	2,27	2,33	0.803
Scalability score Contracts	1,67	2,41	0.011
Comprehension score Contracts	2,13	2,05	0.857
Modifiability score Contracts	1,73	2,17	0.128

However, we perceive an increase in time spent in order to identify and mark elements. The identification tasks required to identify and mark all elements related to a decision goal (task 1) and only the lowest level (contexts and measures) elements related to another goal. This increase in time can be due to marking a higher number of elements in 4 different sheets, as opposed to a single sheet in the monolithic model. Nevertheless, the number of wrong answered questions notably diminished in the Sales example when it was modularized. This is specially relevant, since participants identifying elements in the modularized example had to correctly identify the detail level of a package in the next sheet, whereas those in the monolithic example did not suffer from this drawback. Even though, participants identifying elements in the monolithic Sales model systematically forgot different contexts and measures.

Finally, when asked about how they would structure the models, most of the participants chose to organize them in a modularized fashion. Out of 27 participants 17 chose to package the decision goals and their related elements, 16 packaged the information goals, and 19 chose to package a new hierarchy and include it inside another package wherever it was necessary. It is noteworthy that, although we also analyzed the results according to the participants' expertise and years of experience on the i* framework, no significant correlation nor trend was found that differentiates both groups. This suggests that the addition of modules may affect participants similarly without regards to their experience.

After the first round, we performed a second round with 21 participants, including modification tasks over existing models, as well as the creation of a new model. The examples were the same as in the previous round, and they were presented in the same fashion. The group of participants in this second

Table 3: Tasks performed (left) and independent (top) variables for experiment 2

	Monolithic	Modularized	ρ
Modif. task 1 Sales	202	154,27	0.327
Modif. task 2 Sales	223,6	290	0.217
Modif. task Contracts	128,73	197,6	0.002
Avg. Time drawing	1306,67	1891,44	0.019
Avg. Time/element	50,10	44,34	0.809
Avg. number of elements	25,67	42,89	0.000
Avg. unique non package elements	25,67	27,67	0.021

round was formed by 9 self-evaluated beginners, 5 participants that had some experience, and 5 participants who were advanced users/experts. Furthermore 2 participants did not provide details about their background. As in the previous case, no effect of the participants' expertise on the results was found. The results are shown in Table 3.

After the first step, 4 questionnaires were excluded for the modification tasks, leaving a total of 17 questionnaires. However, the statistical analysis did not show any significant differences in the modification tasks. As previously, we can also perceive a decrease in time spent when the model is bigger and we perform small modifications on a single module (task 1), whereas there is an increase when we require information from multiple modules (task 2).

Finally, the creation of a new model from the scratch had a sample of 15 questionnaires, with a significant correlation between the structure of the model created and every result. Time spent was notably superior for models created with a modularized approach while time spent per element drawn was inferior when the model was modularized. Most importantly, the average number of elements identified from the text which described the model was superior when modules were applied, as opposed to the monolithic structure. Additionally, some monolithic models (filtered in the outliers analysis), presented repeated elements, which should not be created, and tend to increase in number as the model gets bigger.

A second, more detailed review of the results revealed an interesting result. Table 2 shows the number of questions incorrectly answered (average number of errors), either because of a single mistake or because of multiple mistakes. When analyzing the exact nature of errors, we found out that participants using monolithic models mixed different branches of the goal tree and overlooked several elements. On the other hand, participants using the modularized models overlooked sistematically overlooked measures when they were requested to identify hierarchies and measures. This is a significant result since, currently, information modules include both goals and DW elements. When participants tried to locate DW elements, they kept going deeper in the diagram until they found only DW elements, i.e. hierarchy modules. Thus, they systematically forgot measures in the answer.

According to these results, we intend to redefine the modules, substituting information modules with information requirement modules, that are more focused on DW elements. This way, we expect participants to have an easier time identifying where DW elements and user goals are located in the diagram. Furthermore, this redefinition also reduces the amount of diagrams required to interact with the users in order to validate the goals within the goal tree.

6. Conclusions and Future Work

Traditionally, i* models lack any modularity, suffering from scalability and readability issues. Regardless its widespread adoption, the i* framework is still facing several open issues. For instance, [22] mentions as directions of further research: clear definition of the i* language core, proposal of modeling methodologies and analysis techniques, and proposal of modularity constructs. Lack of modularity has been reported to harm model scalability and readability.

Therefore, although the profile presented in [8] is adapted for the semantics present in DWs, it suffers from the same issues as the original framework, since it provides no additional modularity. In turn, when real project models become huge, they turn from a useful tool for communicating with the user into a burden which requires too much work to correct, use and modify. Therefore, an improve in modularity is required in order to maintain the quality of the requirement analysis for DWs, while maintaining the specific semantics for them.

In this work, we have presented a proposal for applying modules, specially designed for DWs. We have defined our proposal, and provided some guidelines on how to correctly apply it. We also have shown an example of application and we have performed an experiment, with our proposal, including participants ranging from new users to experts on i* modeling. The results show a significant increase the scalability of the models, as well as a reduced error rate when identifying the scope of an element present in the model, while helping to create richer goal models. We also perceived an increase in the time necessary to perform different tasks over the models, which may be reduced if these tasks were performed with a CASE tool. Finally, experiment results show that most people tend to group elements in packages at different levels of abstraction, as opposed to adding them in a global schema, which may be helpful in the communication of models.

According to the new findings obtained from a deeper analysis, it would be positive to redefine the modules by substituting the intermediate level (information) by a lower abstraction level that completely separates user goals from information itself (information requirement). This way, both DW designers and users can focus strictly on analyzing the validity of the current set of goals or the current information considered to make decisions.

Finally, an interesting research direction is to analyze how the set of modules could be further extended to consider advanced types of data warehouses, such as distributed spatio-temporal DWs and stream DWs [23, 24, 25, 26]. In these cases, the special nature of the information stored should be considered, allowing not only to better package their requirements, but also improving the maintainability and change management of the data warehouse structure.

Acknowledgments.

This work has been partially supported by the ProS-Req (TIN2010-19130-C02-01) and by the MESOLAP (TIN2010-14860) and SERENIDAD (PEII-11-0327-7035) projects from the Spanish Ministry of Education and the Junta de Comunidades de Castilla La Mancha respectively. Alejandro Maté is funded by the Generalitat Valenciana under an ACIF grant (ACIF/2010/298).

[1] G. Group, Gartner Group BI Revenue Analysis 2009, `http://www.gartner.com/it/page.jsp?id=1357514`.

[2] W. Inmon, Building the data warehouse, Wiley-India, 2009.

[3] R. Kimball, The data warehouse toolkit, Wiley-India, 2009.

[4] E. S.-K. Yu, Modelling strategic relationships for process reengineering, Ph.D. thesis, Toronto, Ont., Canada, Canada (1995).

[5] A. Kleppe, J. Warmer, W. Bast, MDA explained: the model driven architecture: practice and promise, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[6] X. Franch, Incorporating Modules into the i* Framework, in: CAiSE, Vol. 6051 of LNCS, Springer Berlin, 2010, pp. 439–454.

[7] D. L. Parnas, On the criteria to be used in decomposing systems into modules, Communications of the ACM 15 (12) (1972) 1053–1058.

[8] J.-N. Mazón, J. Pardillo, J. Trujillo, A model-driven goal-oriented requirement engineering approach for data warehouses, ER'07, Springer-Verlag, 2007, pp. 255–264.

[9] A. Maté, J. Trujillo, X. Franch, A modularization proposal for goal-oriented analysis of data warehouses using i-star, in: Proceedings of the 30th international conference on Conceptual modeling, ER'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 421–428.

[10] J. Lockerbie, N. A. Maiden, A. Dotan, V. Lichtner, Using i* to support a summative evaluation, in: 4th International Workshop on iStar, 2010, pp. 586–589.

[11] H. Estrada, A. M. Rebollar, O. Pastor, J. Mylopoulos, An empirical evaluation of the i* framework in a model-based software generation environment, in: Advanced Information Systems Engineering, Springer, 2006, pp. 513–527.

[12] C. Martinie, P. Palanque, M. Winckler, Structuring and composition mechanisms to address scalability issues in task models, in: Human-Computer Interaction–INTERACT 2011, Springer, 2011, pp. 589–609.

[13] F. R. S. Paim, J. F. B. de Castro, Dwarf: An approach for requirements definition and management of data warehouse systems, in: Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International, IEEE, 2003, pp. 75–84.

[14] P. Giorgini, S. Rizzi, M. Garzetti, Grand: A goal-oriented approach to requirement analysis in data warehouses, Decision Support Systems 45 (1) (2008) 4–21.

[15] N. Prakash, A. Gosain, An approach to engineering the requirements of data warehouses, Requirements Engineering 13 (1) (2008) 49–72.

[16] S. Luján-Mora, J. Trujillo, I.-Y. Song, A UML profile for multidimensional modeling in data warehouses, DKE 59 (3) (2006) 725–769.

[17] J.-N. Mazón, J. Trujillo, An MDA approach for the development of data warehouses, DSS 45 (1) (2008) 41–58.

[18] E. Kavakli, Goal-oriented requirements engineering: A unifying framework, Requirements Engineering 6 (4) (2002) 237–251.

[19] P. Giorgini, S. Rizzi, M. Garzetti, Goal-oriented requirement analysis for data warehouse design, in: Proceedings of the 8th ACM international workshop on Data warehousing and OLAP, ACM, 2005, pp. 47–56.

[20] M. Laitinen, M. Fayad, R. P. Ward, The problem with scalability, Communications of the ACM 43 (9) (2000) 105–107.

[21] M. Serrano, J. Trujillo, C. Calero, M. Piattini, Metrics for data warehouse conceptual models understandability, Information and Software Technology 49 (8) (2007) 851–870.

[22] X. Franch, The i framework: The way ahead, in: Research Challenges in Information Science (RCIS), 2012 Sixth International Conference on, IEEE, 2012, pp. 1–3.

[23] M. Gorawski, M. Gorawski, Modified r-mvb tree and btv algorithm used in a distributed spatio-temporal data warehouse, in: Parallel Processing and Applied Mathematics, Springer, 2008, pp. 199–208.

[24] M. Gorawski, Extended cascaded star schema and ecolap operations for spatial data warehouse, in: Intelligent Data Engineering and Automated Learning-IDEAL 2009, Springer, 2009, pp. 251–259.

[25] M. Gorawski, R. Malczok, Indexing spatial objects in stream data warehouse, in: Advances in Intelligent Information and Database Systems, Springer, 2010, pp. 53–65.

[26] M. Gorawski, P. Jureczek, Regions of interest in trajectory data warehouse, in: Intelligent Information and Database Systems, Springer, 2010, pp. 74–81.