JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2005**

Vol. 4 No. 5, July-August 2005

Aligning IT with the Changes using the Goal-Driven Development for UML and MDA

The Goal-Driven Development Process for increasing your business reactivity with UML and MDA

Birol Berkem, GOObiz / CNAM, Paris - France

Abstract

For the last few years, organisations have tried to develop their software systems with use case driven and object-oriented development processes. This practice brings some benefits by allowing them to concentrate their analysis and design efforts on the usage dimension of the system and on its architecture of domain objects.

However, modeling a system only with use case driven UML specifications - without focusing in the GOALS that have to be supported by the system – and its domain objects does not allow to organizations good levels of **reactivity** in **face of changes** (1).

Indeed, one of the main issues that we experienced about this lack of reactivity is that in use cases specifications, **actor/system interactions are often tightly coupled (mixed) with business goals!** In this situation, it is particularly difficult to reorganize use case descriptions in coherence when business rules evolve within these goals. On the other hand, implementing changes as a reaction to new requirements for time-to-market is still very much a challenge due to the **orthogonality between functional and object-oriented** representations of specifications and their **absence of traceability** toward the implementation layer.

Within such traceability issues, business experts, analysts, designers and developers ressort to "spaghetti oriented" development that renders the evolution of their system hazardous. In consequence, their business systems are very slow to react to changes!

In order to assist organizations for aligning IT with their changing business environment, a Goal-Driven Development Process should be considered to accompany the OMG's MDA (2). Using such a methodology, MDA users could enable their organisations in propagating changes they capture through their business processes till their IT applications, thus **synchronizing IT with their changing business rules,** as a result to **better capitalize on their business knowledge**, independently of technological changes.

Business Reactivity: *Agility* offered by a business system that ensures to the corresponding organization *swift and coherent adaptation* to the changes of its environment.

MDA® and UML® are registered trademarks of the Object Management Group (OMG) [Ref: www.omg.org].

(*) MDA and UML are both registered trademarks of the OMG (Object Management Group)

1 INTRODUCTION: ABOUT THE GAP BETWEEN BUSINESS AND APPLICATION LAYERS

One of the main factors that causes reactivity issues to IT systems in face of changes of the business rules is due to the lacks of traceability between business and application layers of the organisations. Such a gap causes important delays for adapting IT applications to changes!

As a direct impact of this gap, business analysts are not encouraged to formalize their business behaviors using appropriate business models: both business rules and their usage constraints of the applications are mixed inadvertently within application use cases.

This impacts negatively evolution of business rules and the validation process of use cases. As a result, use cases descriptions are often rendered very long, evolution of business rules they utilize become difficult!

In order to eliminate this gap, thus allowing organizations to align their IT with their changing business environment, changes have to be captured using identifiable goal[1] structures at the business layer. Their impacts have to be propagated first within the business layer for aligning underlying structures of the organisation and finally toward the application layer for synchronizing IT structures (applications, actors) with these changes.

To support such a propagation, specifications must be rendered visible and traceable by the use of goals within and between business and application layers.

The example below shows traditional requirement-gathering process by use cases in the business and application system layers. It also highlights needs for propagating changes within the business layer and toward the structures of the application system layer. An activity diagram could be used for illustrating actions that compose the Business Use Case and looking for potential use cases of the application layer. But an important problem arises: in business use cases descriptions, interactions with participants (actors, workers) are tightly coupled with business rules, so rules are hidden

[1] In the OMG's Business Motivation Model [OMG], a goal is defined as a statement about a state or condition of the enterprise to be brought or sustained through appropriate means. A goal amplifies a vision (an overall image of what the organization wants to be or become) - that is, it indicates what must be satisfied on a continuing basis to effectively attain the vision.

by use cases – Sales (Order-to-billing) in the example. As a result, business goals and underlying behaviours are not rendered visible in order to be aligned with the changes!

As a consequence of this “lack of visibility”, at the application layer, business behaviours cannot be traced to be invoked by the actors in order to synchronise applications with changing business rules.

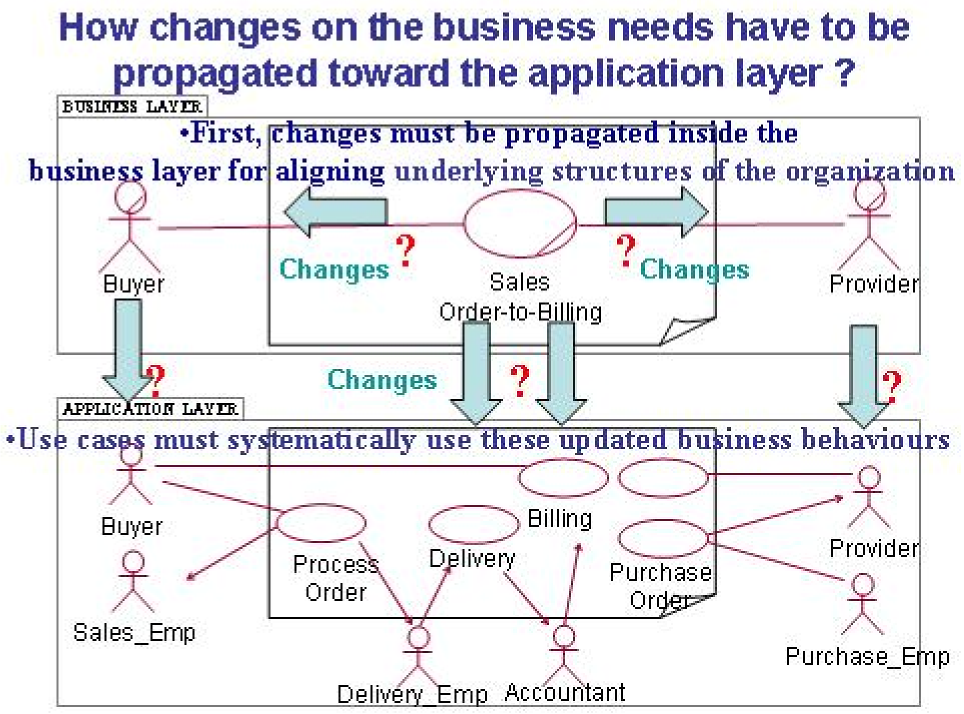

How changes on the business needs have to be propagated toward the application layer ?
BUSINESS LAYER
•First, changes must be propagated inside the business layer for aligning underlying structures of the organization
Buyer
Changes
?
Sales Order-to-Billing
?
Changes
Provider
Changes
?
?
?
APPLICATION LAYER
•Use cases must systematically use these updated business behaviours
Buyer
Sales_Emp
Process Order
Delivery
Billing
Purchase Order
Provider
Purchase_Emp
Delivery_Emp
Accountant

Figure 1: Business Use cases hide rules, so that applications cannot be aligned on changes captured at the business layer

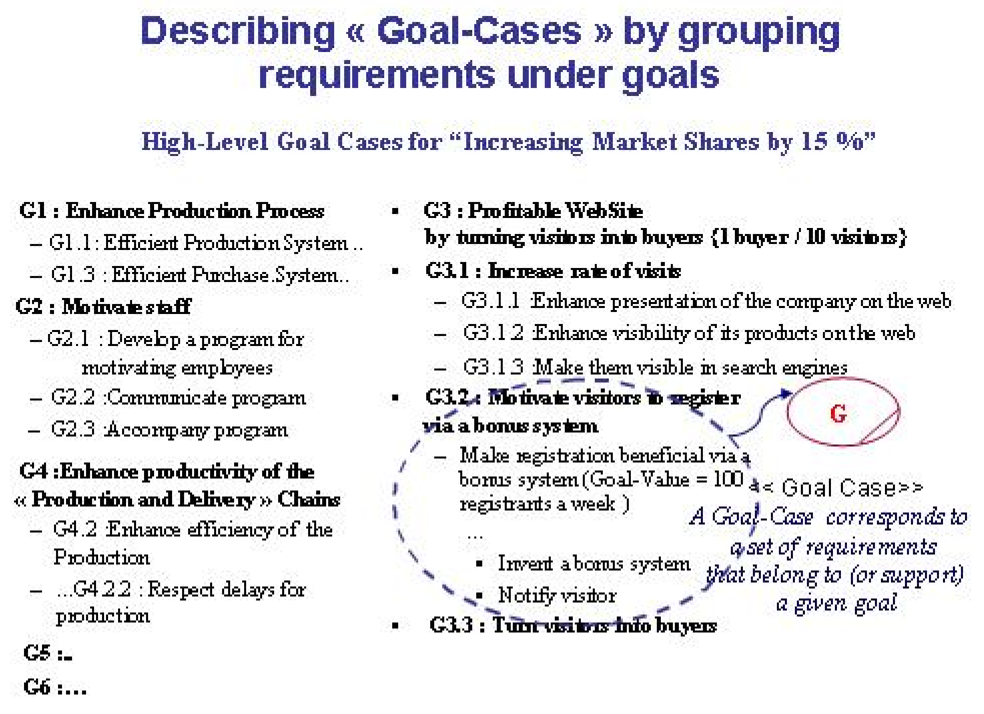
In order to propagate changes coherently within the business and application layers, specifications need to be identified and rendered executable as autonomous and traceable units on the basis of goals rather than domain objects for traceability reason! Thus, changes must be propagated first inside the business layer for aligning underlying structures of the organization. Then, use cases must invoke systematically these updated business behaviours. This is what we present in sections 2 and 3 below through the CIM[2] and PIM[3] viewpoints (levels) of the OMG’s MDA. Finally in section 4, we zoom in automating adaptation to changes through these levels.

[2]**CIM**: Computation Independent Business Model (a specification platform dedicated to the language of non technical business experts)
[3]**PIM**: Platform Independent Model (a specification platform dedicated to the language of the system analysts and designers) **PSM**: Platform Specific Model (a specification platform dedicated to the languages of the specialists of target technologic platforms)

2 CAPTURING CHANGES WITHIN THE BUSINESS LAYER AT THE CIM AND EARLY TESTING AT THE PIM

For assisting business analysts in capturing changes that arise in the business system and propagating them within the business layer of the organization in order to align its underlying structures, requirements have to be grouped using identifiable goal structures (goal cases) at the CIM level. A goal case is a set of requirements (behaviours for the system) that belong to (or support) a given goal. Goal cases are described by grouping courses of actions (behaviors of the system) that support the corresponding goal. For this reason, goal cases constitute the essence of use cases as they make sense to actor / system interactions that permit to realize a goal.

For example, the courses of actions presented in the figure below describe high-level business goals namely G1,.G6. Increase Rate of Visits (G3.1), Motivate Visitors to Register (G3.2) as well as Turning visitors into buyers (G3.3) correspond to underlying goal cases of the high-level goal case G3- Profitable Website by turning visitors into buyers.



Describing « Goal-Cases » by grouping requirements under goals

High-Level Goal Cases for "Increasing Market Shares by 15 %"

G1 : Enhance Production Process
- G1.1 : Efficient Production System ..
- G1.3 : Efficient Purchase System..

G2 : Motivate staff
- G2.1 : Develop a program for motivating employees
- G2.2 :Communicate program
- G2.3 :Accompany program

G4 :Enhance productivity of the « Production and Delivery » Chains
- G4.2 Enhance efficiency of the Production
- ...G4.2.2 : Respect delays for production

G5 :..

G6 :...

- **G3 : Profitable WebSite by turning visitors into buyers {1 buyer / 10 visitors}**
- **G3.1 : Increase rate of visits**
  - G3.1.1 Enhance presentation of the company on the web
  - G3.1.2 Enhance visibility of its products on the web
  - G3.1.3 Make them visible in search engines
- **G3.2 : Motivate visitors to register via a bonus system**
  - Make registration beneficial via a bonus system (Goal-Value = 100 registrants a week)
  - ...
    - Invent a bonus system
    - Notify visitor
- **G3.3 : Turn visitors into buyers**

G
<< Goal Case>>
A Goal-Case corresponds to a set of requirements that belong to (or support) a given goal

Figure 2: Goal Cases of the system and their description (means)

Underlying goal cases that support the high-level goal case G3- WebSite [Turn Visitor into buyers] are illustrated in the figure below with other ones like Present Company and its Products on the Web (G3.1.1) and (G3.1.2).

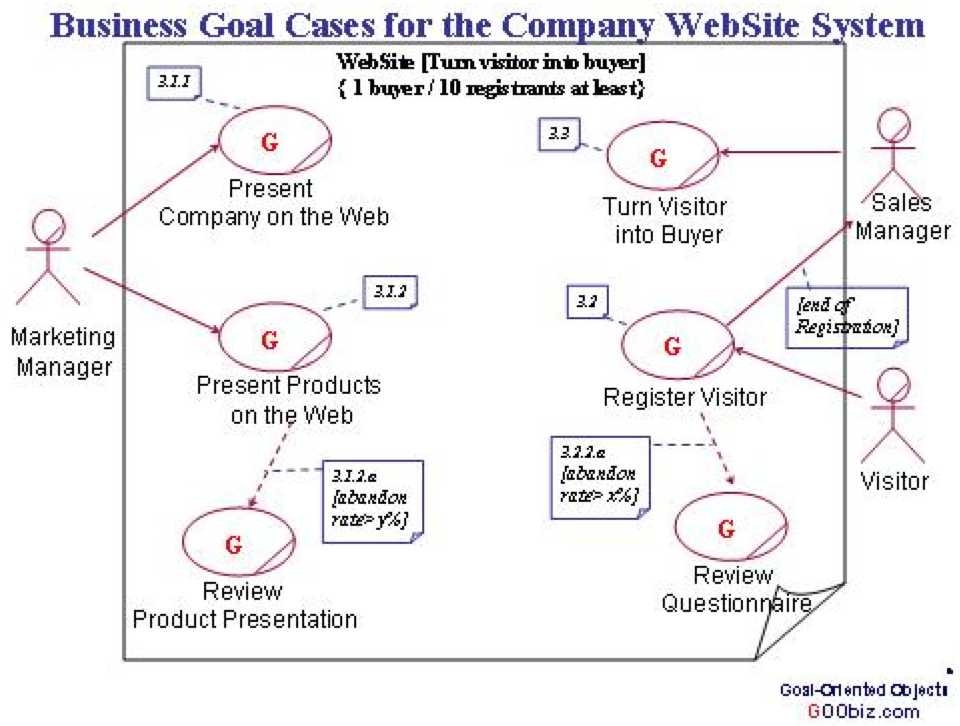


Figure 3: Underlying goal cases of the high-level goal case G3 – Turn visitors into buyers

At the CIM level of the MDA, business experts describe realizations of goal cases using business processes. In order to increase their reactivity by early testing correct understanding of these requirements, the figure 4 shows activities of the business process that realize the goal case Register Visitor (at the left) and transformation of these activities and their guard-conditions respectively as operations and pre-conditions of a Goal-Oriented Object (GOO) at the PIM of the MDA (at the right).

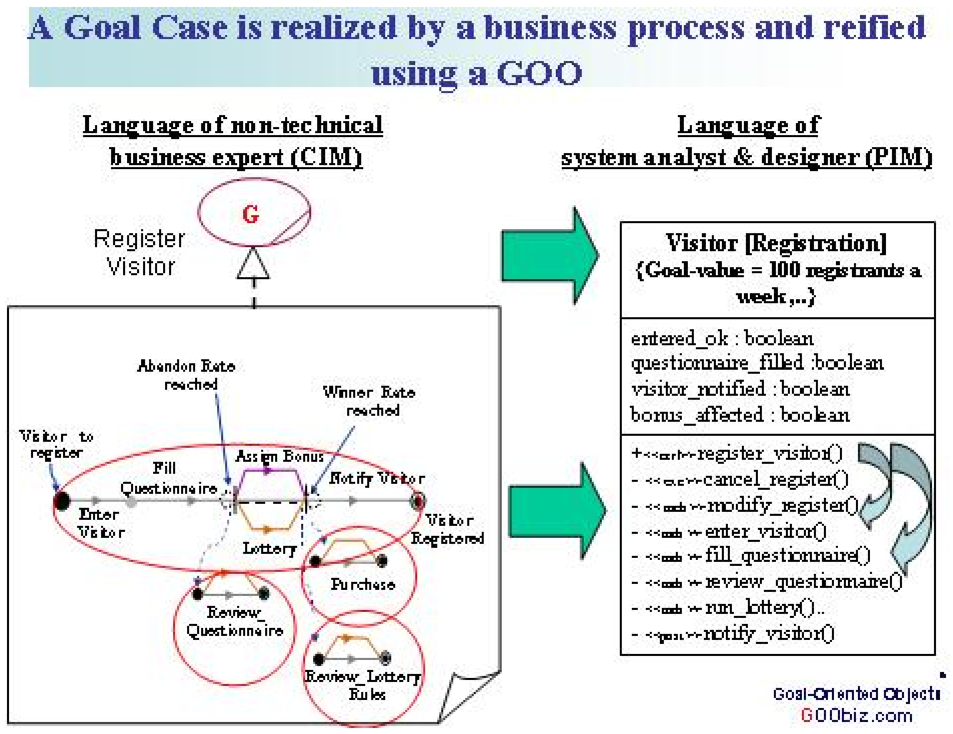


Figure 4: Realization of a goal case by a business process and transformation using a GOO at the PIM

In order to test these operations, a special operation that is called the *controller operation* of the GOO executes them according to their preconditions and depending on the state of the system which is expressed by the attribute values of the related GOO.

In such a way, Goal-Oriented Objects aim at assisting system analysts and designers by making identifiable units of specifications to test at the PIM, thus ensuring to them good levels of traceability between functional and object-oriented representations of specifications. Such a traceability aims at reducing effort for the maintenance of the system in face of changes.

GOOs also confer easy evolutions to complex system specifications by allowing refinement and traceability of these specifications toward lower refinement levels.

In this perspective, complex operations specified at a given refinement level are considered as **goal cases** that have to be realised at the immediate lower refinement level. For example, in the figure below operations *enter_visitor()* and *notify_visitor()* in *Visitor [Registration]* are refined using underlying GOOs *Visitor[Entry] and Visitor [Notification]*. These goal cases have to be realized by actor/system interactions that are necessary respectively for entering and notifying the visitor.

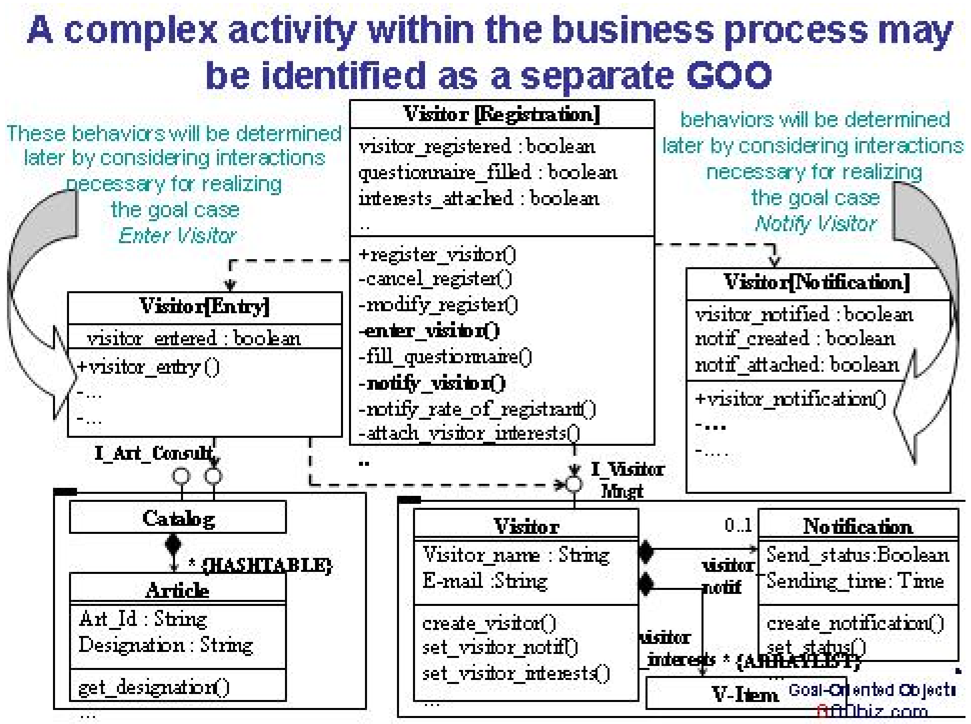


Figure 5: Refinement of complex activities of business processes using underlying GOOs at the PIM. Types of entity objects (the content) that are handled by the operations of the GOOs are illustrated in the bottom part of the figure.

Such PIM level components constitute an **evolutive embryon** of the **architectural backbone of the business processes** as the refinement process by goals permit to them to retain their validity independently of futur design decisions in the CIM level. So, such design decisions at the CIM couldn't alter this architecture when they are mapped at the PIM.

3 PROPAGATING CHANGES TOWARD THE APPLICATION LAYER

In order to align IT with the changes captured at the business layer, changes must be **propagated first** ***through appropriate business components of the business layer*** **and finally toward the application layer** for allowing actors to use updated software components according to their business responsibility.

As actors of the application layer may be represented at the business layer by business actors and workers, in order to assuring traceability of specifications between business and application layers, responsibilities of these participants must be discovered first at the business layer by considering their support to the interactions between such processes as well as in the realization of their activities.

The business process cartography illustrated in figure 6 presents an overview on the interactions between business processes. For example, the responsibility for turning visitor into buyer is assigned to Sales_Manager using the UML's object-in-state Sales_Mgr [turn to buyer] in order to ensure communication between the processes Register Visitor and Turn Visitor to buyer once visitor is entered.

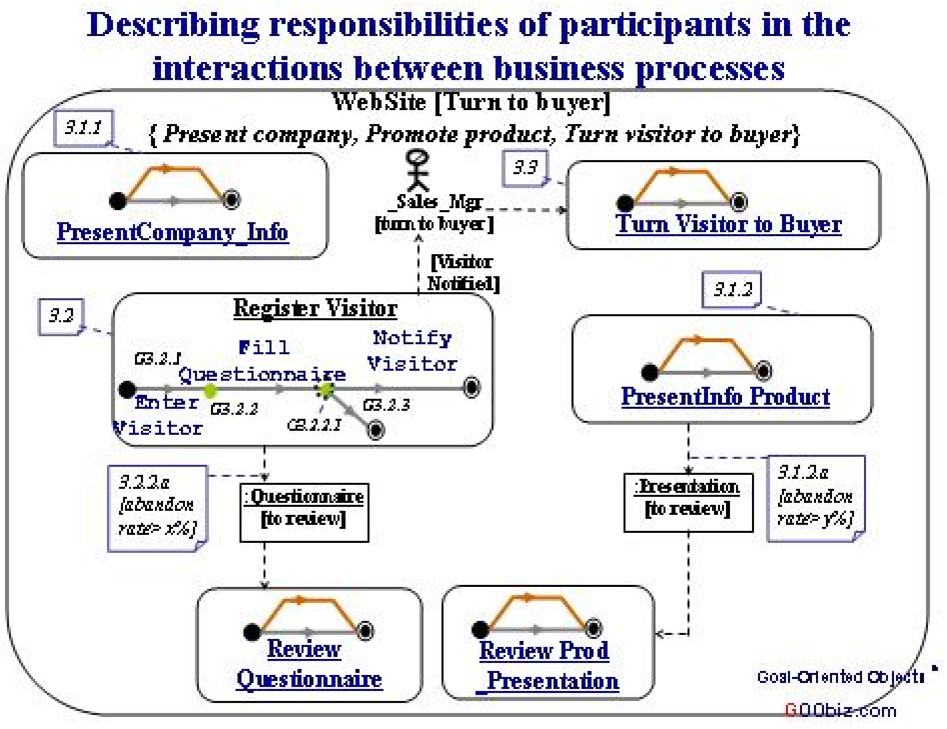

Describing responsibilities of participants in the interactions between business processes
WebSite [Turn to buyer]
{Present company, Promote product, Turn visitor to buyer}
3.1.1
PresentCompany_Info
3.3
_Sales_Mgr_ [turn to buyer]
Turn Visitor to Buyer
[Visitor Notified]
3.2
Register Visitor
G3.2.1
Fill Questionnaire
Notify Visitor
Enter Visitor
G3.2.2
G3.2.3
G3.2.2.1
3.1.2
PresentInfo Product
3.2.2a [abandon rate> x%]
:Questionnaire [to review]
:Presentation [to review]
3.1.2a [abandon rate> y%]
Review Questionnaire
Review Prod _Presentation
Goal-Oriented Objects
GOObiz.com

Figure 6: Business process cartography with responsibilities assigned to participants

The responsibility about *turning visitor into buyer* assigned to the *Sales_Mgr* in the figure 6 may be realized at the design level of the CIM by a couple of use case and goal case components where the use case component describes actor/system interactions whereas the goal case component describes behaviours of the system as illustrated in figure 7 below.

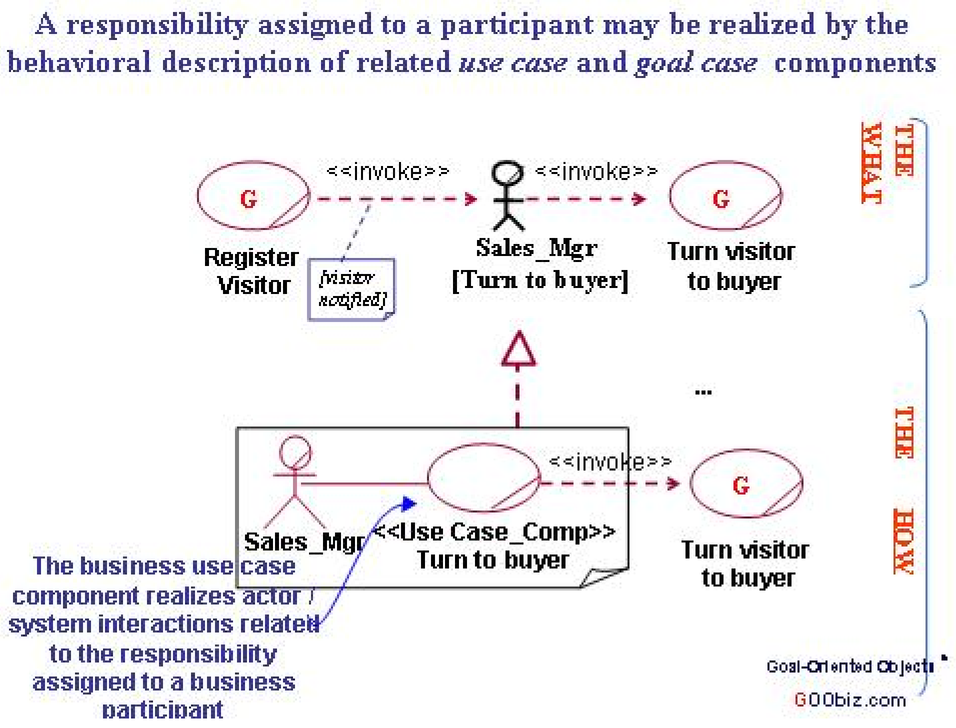

A responsibility assigned to a participant may be realized by the behavioral description of related *use case* and *goal case* components
<<invoke>>
<<invoke>>
G
Register Visitor
[visitor notified]
Sales_Mgr [Turn to buyer]
G
Turn visitor to buyer
THE WHAT
...
<<invoke>>
Sales_Mgr
<<Use Case_Comp>> Turn to buyer
G
Turn visitor to buyer
THE HOW
The business use case component realizes actor / system interactions related to the responsibility assigned to a business participant
Goal-Oriented Object
GOObiz.com

Figure 7: The realization relationship between analysis and design level descriptions at the CIM

Similarly, activities of business processes are realized by underlying actions at the immediate refinement level.

The figure 8 shows behavioral description of use case and goal case components that realize the activity Enter Visitor of the process Register Visitor described in the process cartography of the figure 6.

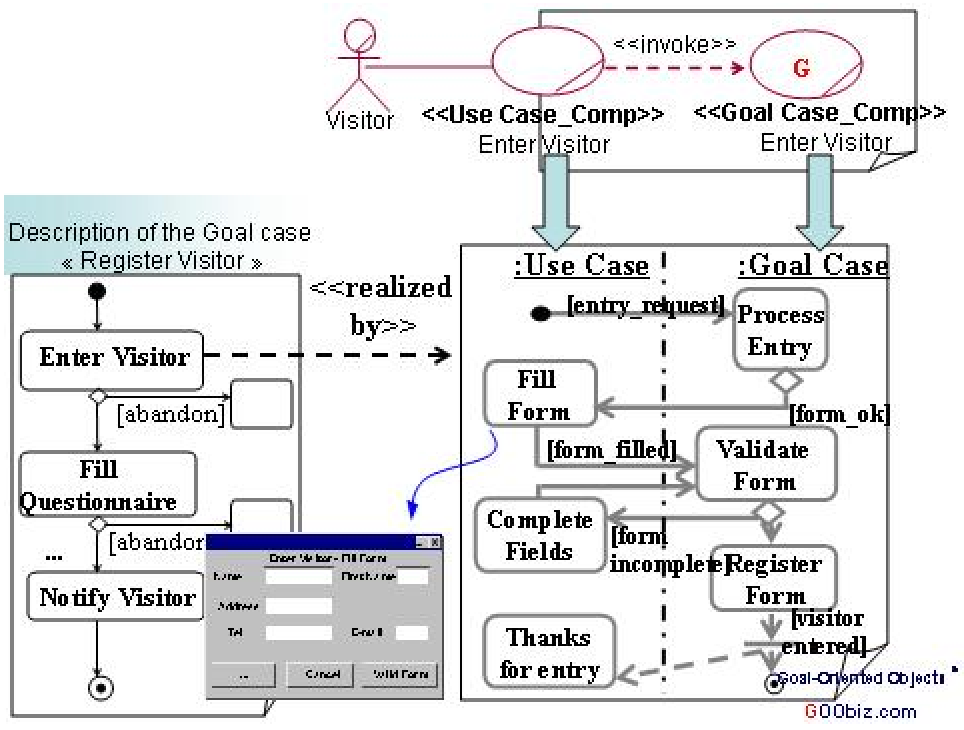


Figure 8: Description of actor / system interactions for realizing the activity *Enter Visito*

In order to plug these behaviours into the architectural backbone of the business process components at the PIM, goal cases and use cases components are reified using corresponding types of GOO. Operations and their pre-conditions for the use case and goal case components are discovered from the activity diagram as illustrated in the figure 9 below.

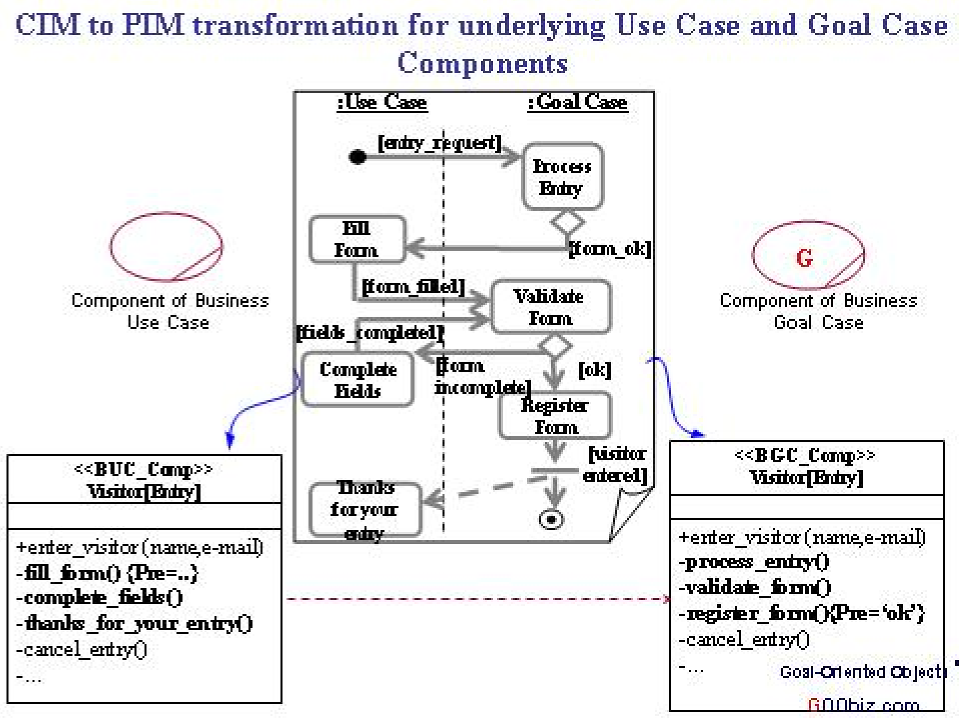


Figure 9: Operations and their pre-conditions for the use case and goal case components at the PIM are discovered from CIM level specifications provided by the activity diagram

The component diagram of the figure 10 shows the architectural backbone of the business components where the previous goal cases and use cases components are plugged.

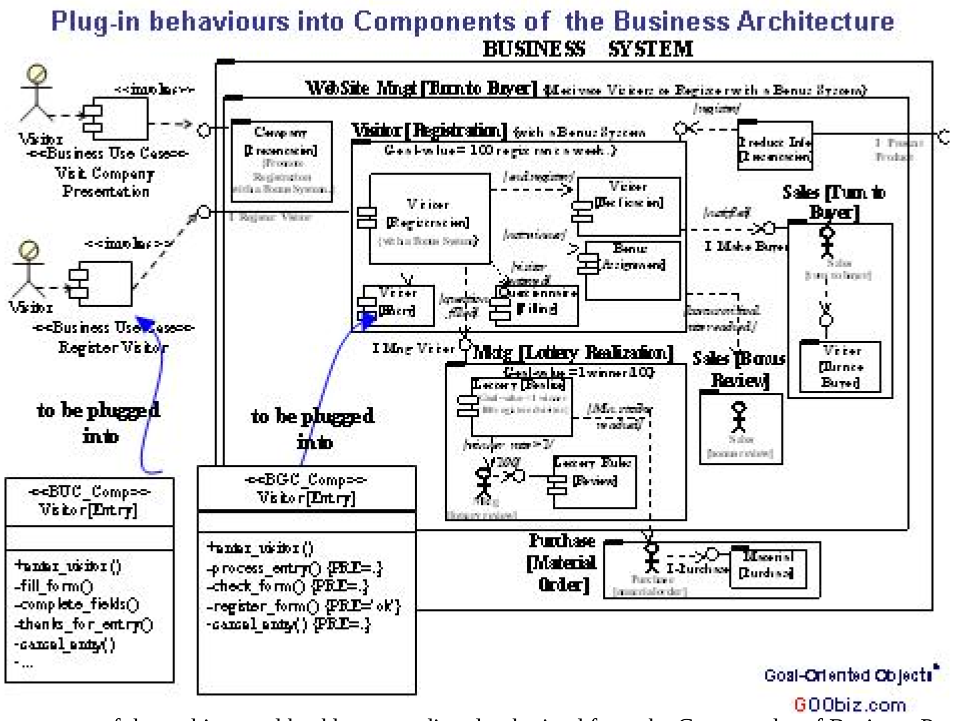


Figure 10: Components of the architectural backbone are directly obtained from the Cartography of Business Processes (figure 6). Based on actor/system interactions, related use case and goal case behaviors are plugged into their appropriate parent components

Separate description of the *goals of the system* from actor/system interactions ensures for use cases the ability to focus only on actor / system interactions for realizing required goals. This way, goal cases descriptions can evolve independently of use cases and are able to guide and govern use cases to let actors of the application layer use the system for realizing “the right business goals”!

Using business behaviours at the application layer

At the application layer, according to responsibilities assigned to the participants of business processes, actors of the application system may invoke business behaviours depending on their application choices, but in respect to the constraints that have been specified at the business layer.

Application choices of the actors of the application system are specified using *application goal cases* that are constituted by a set of requirements that belong (or support) the goal of an actor. These choices may be applied before, during or at the end of the execution of business goal cases. For example, during his registration on the website system, the visitor may decide to look for available promotions or take a look on products of his interest.

Thus, depending on the needs, an actor may require from an application use case to invoke:

Scenario 1- One or more business processes of his/her choices without any complementary application process

Scenario 2- An application process before or after using a business process

Scenario 3- An application process during the execution of a business process

In scenario 1, the application use case of the actor invokes behaviors defined in the corresponding business use case(s). This is materialized by the "invocation" relationship between use cases in the figure 11.

In scenario 2, the application use case invokes additional behaviors from **application goal cases** before or after invoking those from business use case(s).

Finally, in scenario 3, the application use case and goal case specialize respectively behaviors defined in the business use case and business goal case **according to their constraints of pre-condition and post-condition that have to be respected**. In other terms, these application components are "conform" to their corresponding business components.

The figure below illustrates an overview on these scenarios.

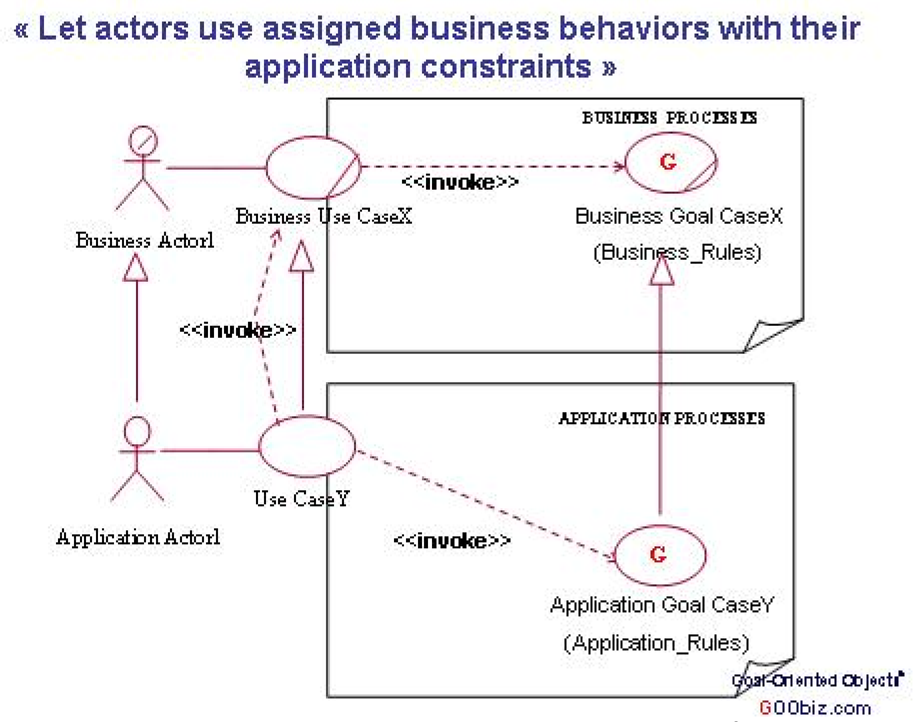


Figure 11: Behaviors stored in business goal case components are invoked by the actors of the application layer depending on their choices

The next section presents a summary of the Goal-Driven Development Process that illustrates how to bridge the related UML artifacts to confer good levels of reactivity to the resulting system using the CIM, PIM and PSM levels of the MDA.

4 STEPS OF THE GOAL-DRIVEN DEVELOPMENT PROCESS FOR AUTOMATING ADAPTATION TO CHANGES THROUGH THE MDA'S CIM/PIM/PSM LEVELS

Solutions described above do help analysts and designers in rendering their system and software specifications identifiable, flexible, executable and traceable, based on the requirements of non-technical business experts.

A development process is necessary for using these solutions inside the Goal-Driven Development Process where specifications of the business experts are captured first at the CIM being grouped by business goals (step 1). Realization of these specifications are then formalized using the UML's activity diagram at the CIM (step 2). These may be transformed at this stage for getting a preview of the architectural backbone of the system at the PIM. At the next step responsibilities of the business participants (actors and workers) while realizing activities of the process are specified at the CIM (step 4) and the resulting PIM level components are plugged to the architecture (step 5). Finally, in the last step of the process, application choices of the actors are integrated in the architecture for allowing them to invoke business behaviours via their application use cases (step 6). An overview of these steps are described schematically in the figure below.

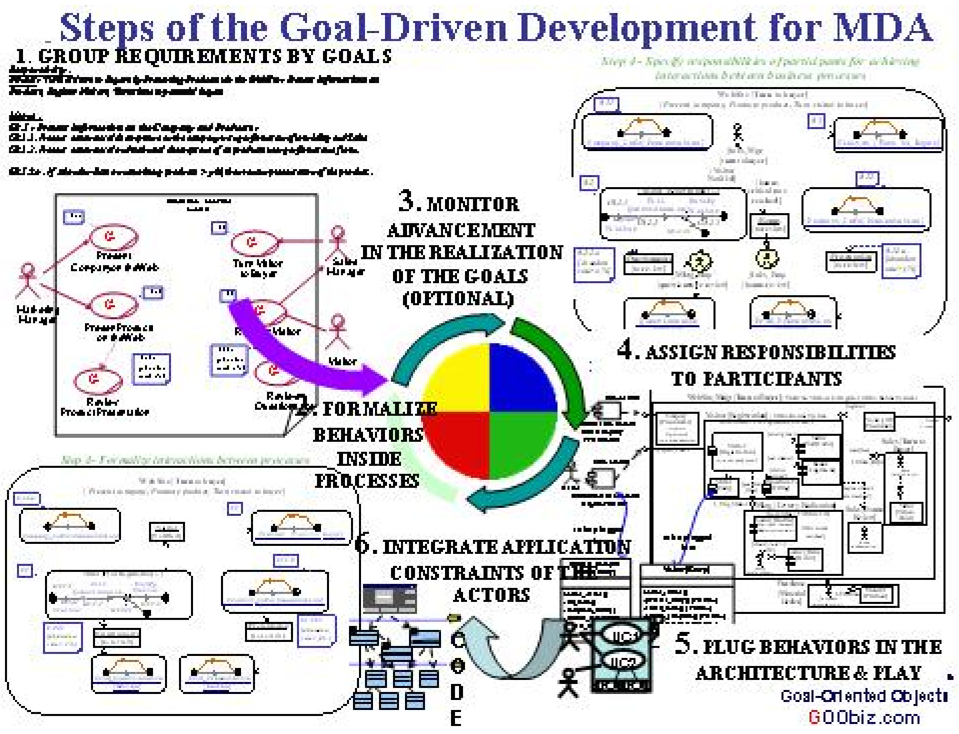

Steps of the Goal-Driven Development for MDA
1. GROUP REQUIREMENTS BY GOALS
2. FORMALIZE BEHAVIORS INSIDE PROCESSES
3. MONITOR ADVANCEMENT IN THE REALIZATION OF THE GOALS (OPTIONAL)
4. ASSIGN RESPONSIBILITIES TO PARTICIPANTS
5. PLUG BEHAVIORS IN THE ARCHITECTURE & PLAY
6. INTEGRATE APPLICATION CONSTRAINTS OF THE ACTORS
CODE
Goal-Oriented Objects
GOObiz.com

Figure 12: An overview of the UML diagrams partitioned between the CIM, PIM and PSM levels of the MDA with the bridges required by the Goal-Driven Development Process

The Goal-Driven Development Framework shows distribution of these artefacts and bridges through the CIM/PIM/PSM levels of the MDA along the model transformation.

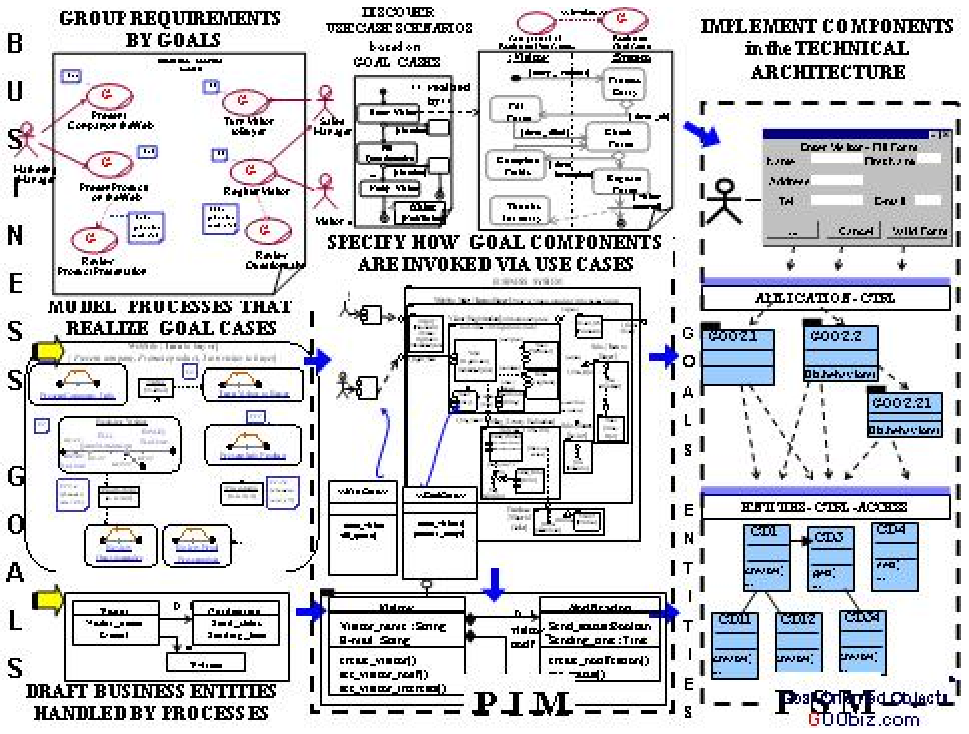

GROUP REQUIREMENTS BY GOALS
DISCOVER USE CASE SCENARIOS based on GOAL CASES
IMPLEMENT COMPONENTS in the TECHNICAL ARCHITECTURE
SPECIFY HOW GOAL COMPONENTS ARE INVOKED VIA USE CASES
MODEL PROCESSES THAT REALIZE GOAL CASES
DRAFT BUSINESS ENTITIES HANDLED BY PROCESSES
BUSINESS GOALS
GOALS
ENTITIES
APPLICATION - CTRL
ENTITIES - CTRL - ACCESS
PIM
PSM
GOOBiz.com

Figure 13: An overview of the UML diagrams partitioned between the CIM, PIM and PSM levels of the MDA with the bridges required by the Goal-Driven Development Process

The development framework allows traceability of specifications from requirement analysis (at the CIM) till implementation of related business and application behaviors on the PIM. On the basis of generic architectural layers of the PIM, it does also assist transformation of these PIM level specifications into appropriate components of the Platform Specific Model (PSM) *such as Servlets/JSPs, Session and Entity Beans in J2EE, WebServices, etc...*

5 CONCLUSION

Solutions presented above as part of the Goal-Driven Development Process aim at aligning IT applications with the business needs in order to guide actors of the application systems to use the system according to its business goals whatever changes that may arise on these goals, their supporting strategies as well as on the business processes that realize them.

In this direction, these solutions represent key factors for increasing reactivity of business systems developed with UML in the sprit of the OMG's Model Driven Architecture (MDA).

As a support to these solutions, the methodological framework of the Goal-Driven Development Process acts as a catalysor by ensuring **traceability between related UML artifacts through the CIM, PIM, and PSM levels of the MDA,** contributing so directly to the reactivity of the resulting system in face of changes.

REFERENCES

[Kend98] Elizabeth Kendall, 3 Goals and Roles http://www.ibissoft/se/events/-oocontr/kendall.htm

[OMG] The business governance in a volatile world. Business Motivation Model http://www.businessrulesgroup.org/second_paper/BRG-BRMM.pdf

[MDA] http://www.omg.org/cgi-bin/doc?mda-guide

[Berk05] Birol Berkem, Goal Driven Development for MDA –OMG's Technical Meeting in Athens http://www.omg.org/cgi-bin/doc?omg/2005-04-05, April 2005

[Berk03] Birol Berkem, "*How to increase your business reactivity with UML/MDA*", in Journal of Object Technology, vol. 2, no. 6, November-December 2003 http://www.jot.fm/issues/issue_2003_11/article4

[Berk99] Birol Berkem, Traceability Management with UML - Journal Of Object Oriented Programming (JOOP), September 1999

[Goobiz] How to increase your business reactivity with UML and MDA? (Patterns and processes) http://www.goobiz.com/GOObizWP/GOObizWP.htm

About the author

Birol Berkem is a consultant and trainer in the areas of business and application system analysis and design with the object technology. He is the author of the Goal-Driven Development Patterns and Frameworks for the traceability of specifications with MDA/UML. His e-mail address is birol.berkem@goobiz.com