Formal specification and modeling of a project-oriented fractal company using situation calculus

Mercedes Canvesio[1], Ernesto Martinez[2]

[1]CIDISI-UTN, Lavaisse 610, Santa Fe 3000, Argentina
[2]INGAR (CONICET-UTN), Avellaneda 3657, Santa Fe, S3002 GJC, Argentina
mcanaves@frsf.utn.edu.ar ecmarti@conicet-santafe.gob.ar

Abstract. *In response to global competition, small and medium enterprises (SMEs) can gain a competitive edge from virtual networking. New information and communication technology further extended these opportunities but also raise organizational challenges. For SMEs integration to be fully realized, an information/management system describing roles, functions, tasks, objectives, goals of all actors and resources involved is required for enterprise networking as a whole. A novel approach to formal modeling of a fractal management system for an integrated enterprise using projects is proposed. As a guideline in information system design, an axiomatic, representation of the dynamics of the project-oriented fractal company is given using situation calculus. A prototype of the ProFCo architecture using the Enterprise Project Management Solution of Microsoft Project 2007 will be presented*

1. Introduction

Intense competition in the manufacturing and service sectors due to globalization, product customization, variations in demand patterns and rapid technological development demands new enterprise models. To survive, companies must increase product portfolio, reduce time-to-market, shorten product-life cycles and at the same time maintain good product quality and reduce investment cost. Competitive threats are much worse for small and medium enterprises (SMEs) which are reengineering their production and management systems to compete successfully. SMEs have to organize themselves in effective production networks to achieve a higher degree of flexibility, agility and low costs to cope with the increasing rate of change and complexity of a highly competitive environment. To address competitive threats and concentrate on their core competences and strengths networking is the alternative of choice for each individual SME survival and prosperity [Canavesio et al, 2007; Basole et al, 2011]. Objectives of a SMEs network include increased agility in responding to competitive threats, a more comprehensive pool of skills and resources, economy of scale and product portfolio diversification.

To grasp all the benefits of virtual enterprise networking is mandatory to define an integrated company model to influence by design the behavior of each SME and relationships thereof. To solve this problem, [Canavesio et al, 2007] proposed an integrated company model revolving around the concept of a project-oriented fractal company for SMEs networking. In this model, the fractal management unit is modeled as a project. Each project is seen as an autonomous, temporary entity within the enterprise network, in which different types of expertise are combined to achieve a concrete goal (e.g., in product development, satisfying resource usages as scheduled, etc.). The underlying idea in the project-based fractal company is establishing client-server (temporal) relationships between project and resource managers in an open market economy.

This work presents a formal model of the project-based fractal company for SMEs networking. Situation calculus is used to represent the plethora of relationships involving roles, actors, goals, projects, resources, etc, at different levels of abstraction over time. The formal enterprise model is a set of axioms which provides the detailed specification of the evolution of situations in the project-oriented management system resulting from actions taken by actors and events allows answering queries for what, who, when, where and how related to project, tasks and resources involved in client-server relationships. The set of axioms is expressed in ECLiPSe Prolog for reasoning and making inferences about the situation dynamics. In the last section of this paper, a prototype of the project-oriented fractal management model using the Enterprise Project Management Solution of Microsoft Project 2007 will be presented.

2. Project-based fractal enterprise model

The fractal company [Warnecke, 1993] is a conceptual enterprise model that aims to achieve a high degree of flexibility to react and adapt quickly to environmental changes using decentralized and autonomous organizational units known as fractals. A fractal is defined as a structure that describes an identical pattern that replicates itself at different abstraction levels in a recursive way. In the project-based fractal company, a fractal management unit is modeled as a project [Canavesio et al, 2007]. Thus, each project fractal is seen as an autonomous, self-optimizing, self-learning and goal-driven entity, in which different types of expertise are combined to achieve a concrete goal or deliverable (e.g. completing an order, discovery of a new drug for cancer treatment, introducing a new product, satisfying resource usages as scheduled, etc.). A fractal approach is used in order to allow enterprises to be able to self-adjust to the changes in the environment [Canavesio et al, 2007; Ramanathan, 2005; Kirikova, 2009; Bider et al, 2012]. Using projects at all abstraction levels is both a generator of efficiency and absorber of complexity to embed a highly adaptive and responsive organizational design that can balance each project internal complexity with environmental pressures on both the short- and long-term horizons. The other advantage of the proposed recursive structure is that naturally lends decentralized decision making [Canavesio et al, 2007].

This management fractal unit is made up of a project-manager and a managed object (Fig.1). Each project-fractal unit is an autonomic unit which the project manager agent implements the monitor-analyze-plan-execute (MAPE) loop [Kephart et al, 2003]. Thus, the project manager is able to: i) sense the present situation of the fractal unit, ii) monitor the managed object and its external environment in order to construct and execute plans based on an analysis of available information iii) learn to act based on experience and evaluative feedback of actions taken, and, iv) interact with other project managers. The project-based fractal model separates the management of goals from the management of the resources needed to obtain such goals but in both cases the fractal management unit is conceived as a project. Thus, the managed object may be either an ends or a means in the fractal company system. Furthermore, learning allows the accumulation of knowledge based on the actions executed by other project managers.

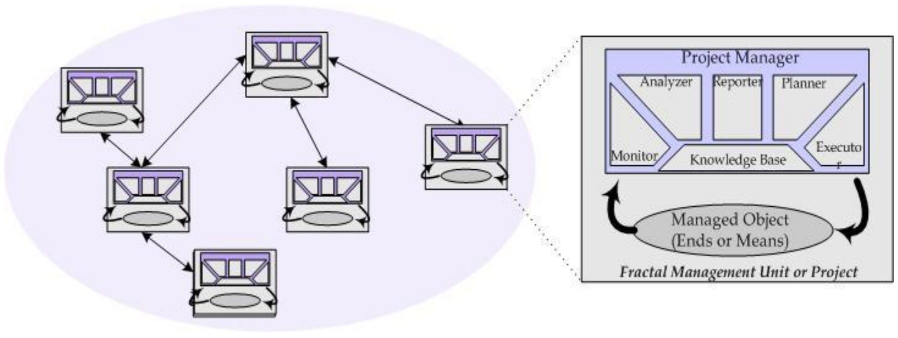


Figure 1. Internal Structure of a project-fractal management unit

In each project, depending on the type of managed object (ends or means), its project manager plays the ends manager role or the means manager role, respectively. Project managers interact and communicate through client-server relationships established among them. In each relationship, the ends-manager is the consumer or client for a given resource and the means-manager is the supplier or server of the concerned resource. These relationships are established by free negotiation among selfish actors in the fractal company that are interested in contributing to achieve business goals and deliverables.

3. Formal Enterprise Modeling

Fox and Gruninger (1998) define an enterprise model as an abstraction that identifies and represents the basic elements that describe an enterprise (structure, process, information flow, resources, goals and relationships). The main roles of an enterprise model are: i) to achieve model-driven enterprise analysis, design, and operation; ii) to improve the efficiency and the effectiveness of the company; iii) to allow exploring alternative models in the design of enterprises spanning the organization structure and behavior, iv) to document the operational and predefined processes of the company. An integrated enterprise model is a representation that describes the main structures and relationships, information flows, roles, goals, resources, behavior of actors and constraints within an enterprise network as a whole [Canavesio et al, 2007]. This representation gives the members of each enterprise the ability to make decisions on how to design the various business processes [Hoverstadt, 2009]. More specifically, [Koubaraskis et al, 2002; Bork, 2014] define a formal enterprise model as an abstraction that describes and represents rigorously and accurately the main elements that describe an enterprise. Thus, the formalization can capture both structural and dynamic aspects of an integrated enterprise model and add more advantages such as: stating the requirements for a detailed design of integrated information and management system for distributed decision-making and actions coordination. Also, a formal model allows the execution of simulations to describe and analyze enterprise network dynamics and constraints. Moreover, model formalization helps defining the semantics of actor interactions in a precise way and without ambiguities as a mechanism. Finally, formal model makes room for representing incomplete knowledge and uncertainty about actor policies in an unambiguous way. Thus, formal integrated enterprise model can be validated and checked for rigor and robustness [Chapurlat et al, 2006; Chapurlat et al, 2008;Jonker et al, 2007].

The formalization of an integrated enterprise model incorporates axioms and reasoning rules allowing modeler to obtain a deductive enterprise model able to answer commonsense queries [Fox et al, 1998; Mueller, 2006]. Commonsense queries require that the information systems be able to deduce answers to questions that one would normally assume can be answered if one has a commonsense understanding of an integrated enterprise, such as: What is the actor X doing? Who is doing the activity Y? When, Where, How and Why is the activity Y done? Thus, a formal enterprise model makes possible to test how complete and correct is the design of the fractal company information systems as far as its capacity to reply to queries associated with the fractal company management allowing tracing of project life cycle, manager decisions, evaluative feedback of actions taken, etc. For all that, it is necessary to translate the formal and deductive enterprise model into an executable enterprise model where all actor decision-making policies and interactions are implemented using some logical programming language. In order to formalize the enterprise model of a project-based fractal enterprise the situation calculus language will be used. For related work in this regard, see [Koubaraskis et al 2002; Jonker et al, 2007].

3.1. Situation Calculus

The situation calculus [Reiter, 2003] is a logical language for representing the dynamics of a "world" of objects. Situation calculus perceives the world as being comprised of a sequence of situations, each of which is a snapshot of the state of the world. Situations are generated from previous situations by actions performed by actors (Fig. 2). These actions may be ordinary or knowledge-producing actions [Reiter, 2003; Scherl et al, 2003]. Ordinary actions cause changes on an actor environment, for example, *StartProjectPlanning(p), EndExecutionTask(t,pl,p), AllocateResource(r,t,pl,p)*, whereas knowledge-producing actions affect the actor knowledge state by performing actions of sensing or reading over any object in the environment. The effect of knowledge-producing actions typically satisfies the prerequisite of a later action. For instance, the $sense_{achievedgoal}(pr)$ action causes that an actor knows that his/her project goal has been achieved, and then he/she performs an action signaling concerned managers the completion of that project.

A possible world history is a sequence of actions represented by a first-order term called situation, and each action is denoted by a function symbol with an arbitrary number of arguments, for example, *CreateInstanceProject(pr,g)* could stand for the action of creating project *pr* to achieve goal *g*. The constant *S0* is used to denote the initial situation, the situation in which actions have not occurred yet (Fig. 2(a)). The term *Do(a,s)* denotes the successor situation to s that results from the execution of the action a in situation s. For instance, in the situation *S0* (See Fig. 2(a)), the ACC209 project receives approval to send to the market the new drug. Thus, its manager decides to close it and thus a new situation is created and depicted by Fig. 2(b). Formally, the new situation S1 is denoted by the term *Do(FinishProject (ACC209),S0)*.

In the situation calculus, the world dynamics is represented by relations and functions whose values may differ from one situation to another and they are called relational and functional fluents, respectively. They are denoted by predicates or functional symbols having a situation term as their last argument. For instance, *ProjectOnSchedule(ACC209,S0)* is a relational fluent that returns the true value in *S0* and expresses that in S0 the ACC209 project is in progress and on schedule (Fig. 2(a)). *ProjectProgress(ACC209,S0)* is a functional fluent that will return the value 98 in the situation S0. It expresses that in S0 the ACC209 project has progressed 98%.

An action is specified by conditions stating when it can be performed. The precondition axiom form is $Poss(a(x),s) \Leftrightarrow \pi a(x,s)$ where $\pi a(x,s)$ is a formula specifying the preconditions for action a(x). For example, it is possible to finish a project p in the situation s if and only if the project p has been executed as it was planned and the project goal g has been achieved in s. Formally, the precondition axiom for the action *FinishProject(p)* might be:

$$Poss(FinishProject(p),s) \Leftrightarrow ProjectOnSchedule(p,s) \wedge AchievedGoal(g,p,s)$$

Then, it is necessary to specify what has changed in the world after performing some action with effect axioms. But these axioms are not sufficient to reason about changes in the modeled world. It is usually necessary to add axioms that specify when fluents remain unchanged by actions. These axioms are called frame axioms. The frame problem arises because the number of these frame axioms is very large, in general, of the order of 2 times # actions times #fluents. The solution of the frame problem was presented by [Reiter, 2001] who defined a successor state axiom for each fluent, relational or functional. Thus, successor axioms specify how actions change the value of a fluent. For example, in a situation denoted by *Do(a,s)*, the relational fluent *ProjectOnSchedule(p,do(a,s))* will be true if and only if either an action that start project p is performed or project p is in progress and an action that finishes or interrupts project p does not occur in s. Formally, the successor axiom for this fluent is:

$$ProjectOnSchedule(p,do(a,s)) \Leftrightarrow a = StartProject(p) \vee ProjectOnSchedule(p,s) \wedge (a \neq FinishProject(p) \wedge a \neq StartNoFeasibleProject\ (p))$$

(a) (b)

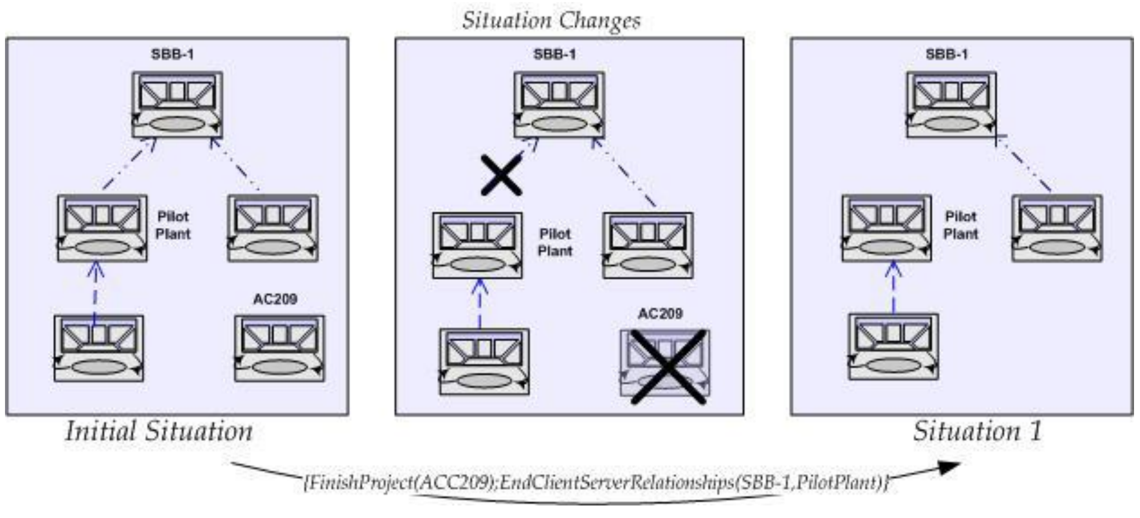


Figure 2. Sequence of situations

3.2. Formal modeling of a project-based fractal company

In a fractal company model, actor actions correspond to some project manager playing its role (e.g., to create a project instance, to start/end project planning). Then, an action sequence involving both ordinary and knowledge-producing actions that were executed by different project managers affect the dynamics of the project-based fractal company. Thus, the dynamics of situations is represented by changes in the state of managed objects (projects and resources) and client-server relationships over time [Canavesio et al, 2007]. With this formal representation, a project manager may know the state of a managed object at any time and at different levels of details. In order to answer different types of queries, an ordered sequence of actions is applied from a given initial situation to the current situation using precondition axioms and successor state axioms defined in the formal model of the fractal company.

Thus, a formal model of the project-based fractal company is a comprehensive set of axioms that describe how situations change as a response to actions taken by project managers. Knowledge base representation and changes is done through four different types of axioms:

- Precondition axioms for each possible ordinary and knowledge-producing action that can be executed by a project manager (Table 1).
- Successor state axioms for each fluent which describe the situation of the fractal company at a given time. These fluents represent object management states, properties, and relationships between actors (Table 2).
- Axioms that describe predicates independent of a situation. For example, Company(C), Resource(R).
- Axioms describing the initial situation of the fractal company.
- Unique names axioms for the primitive actions.
- Foundational axioms of the situation calculus (for details, see [Reiter, 2001]).

Table 1. Examples of precondition of project actions

Action	Preconditions
CreateInstanceProject(pr,g)	$(\exists g).\ Goal(g,s) \wedge \neg CreatedInstanceProject(pr,s) \wedge \neg(\exists pr^*).Project(pr^*,g,s)$
StartPlanningProject (pr)	$CreatedInstanceProject(pr,s) \wedge CreatingPlan(p,pr,s)$
EndPlanningProject(pr)	$PlanningProject(pr,s) \wedge CreatedPlan(p,pr,s)$
StartProjectProgress(pr)	$PlannedProject(pr,s) \wedge ProjectPlanOnSchedule(p,pr,s)$
StartNoFeasibleProject(pr)	$ProjectOnSchedule(pr,s) \wedge NoFeasiblePlan(p,pr,s) \vee [FinishedPlan(p,pr,s) \wedge \neg AchievedGoal(pr,g,s)]$
ResumeProjectProgress (pr)	$NoFeasibleProject(pr,s) \wedge (\exists pl^*)CreatePlan(pl^*,pr,s)$
AbortProject(pr)	$NoFeasibleProject(pr,s) \wedge \neg(\exists pl^*)CreatePlan(pl^*,pr,s) \vee ClosedProject(pr,s)$
FinishProject(pr)	$ProjectOnSchedule(pr,s) \wedge FinishedProjectPlan(pl,pr,s) \wedge AchievedGoal(pr,g,s) \vee AbortedProject(pr,s)$

Table 2. Examples of successor state for relational fluents

Relational Fluents	***Sucesor states for relational fluents***
CreatedInstanceProject(pr, do(a,s))	*a = CreateIntanceProject(pr,g)* $\vee$ *CreatedInstanceProject(pr,s)* $\wedge$ *a* $\neq$ *StartPlanningProject(pr)*
PlanningProject(pr,do(a,s))	*a = StartPlanningProject(pr)* $\vee$ *PlanningProject(pr,s)* $\wedge$ *a* $\neq$ *EndPlanningProject(pr)*
PlannedProject(pr,do(a,s))	*a = EndPlanningProject(pr)* $\vee$ *PlannedProject(pr,s)* $\wedge$ *a* $\neq$ *StartProjectProgress(pr)*
ProjectOnSchedule(pr,do(a,s))	*a = StartProjectProgress(pr)* $\vee$ *ProjectOnSchedule(pr,s)* $\wedge$ *a* $\neq$ *FinishProject(pr)* $\wedge$ *a* $\neq$ *StartNoFeasibleProject(pr)*
NoFeasibleProject(pr,do(a,s))	*a = StartNoFeasibleProject(pr)* $\vee$ *NoFeasibleProject(pr,s)* $\wedge$ *a* $\neq$ *ResumeProjectProgress(pr)* $\wedge$ *a* $\neq$ *AbortProject(pr)*
AbortedProject(pr,do(a,s))	*a = AbortProject(pr)* $\vee$ *AbortedProject(pr,s))* $\wedge$ *a* $\neq$ *FinishProject(pr)*
ClosedProject(pr,do(a,s))	*a = FinishProject(pr)*

This set of axioms that defines the knowledge-base of the project-based fractal company was implemented using ECLiPSe Prolog [Niederlinski, 2013]. Thus, this executable enterprise model allows querying about situation of the fractal company.

Like an illustrative example, Fig. 3 shows Prolog clauses that translate precondition and successor state axioms of the formal enterprise model.

The usefulness of a formal enterprise model is determined by the abstraction level and type of commonsense queries that are able to answer. Thus, a formal enterprise model will be complete if it is possible to respond a set of competency questions [Fox et al, 1998]

```
%Successor State Axioms

createdInstanceProject(Pr, Do(A,S)):- A = createIntanceProject(Pr,G),
      createdInstanceProject(Pr,S), not A = startPlanningProject(Pr).

planningProject(Pr,Do(A,S)):-A = startPlanningProject(Pr),
      planningProject(Pr,S), not A = endPlanningProject(Pr).

plannedProject(Pr,Do(A,S)):- A = endPlanningProject(Pr),
      plannedProject(Pr,s), not A = startProjectProgress(Pr).

projectOnSchedule(Pr,Do(A,S)):- A = StartProjectProgress(Pr),
      projectOnSchedule(Pr,S), not A = finishProject(Pr), not A =
      startNoFeasibleProject(Pr).

noFeasibleProject(Pr,Do(A,S)):-  A = StartNoFeasibleProject(Pr);
      noFeasibleProject(Pr,S) ∧ not A = resumeProjectProgress(Pr), not A =
      abortProject(Pr).

abortedProject(Pr,Do(A,S)):- A = abortProject(Pr); abortedProject(Pr,S)),
      not A = finishProject(Pr).

closedProject(Pr,Do(A,S)):- A = finishProject(Pr).

%Precondition axioms

poss(createdInstanceProject(Pr,G),S) :- goal(G,S), not
      createdInstanceProject(Pr,S),not project(Pr*,G,S).

poss(startPlanningProject (Pr),S) :- createdInstanceProject(Pr,S),
      creatingPlan(Pl,Pr,S).
```

```
poss(endPlanningProject(Pr),S) :- planningProject(Pr,S),
      CreatedPlan(p,pr,s).

poss(startProjectProgress(Pr),S) :- plannedProject(Pr,S),
      projectPlanOnSchedule(Pl,Pr,S).

poss(startNoFeasibleProject(Pr),S) :-  projectOnSchedule(Pr,S),
      noFeasiblePlan(Pl,Pr,S);finishedPlan(Pl,Pr,S), not
      achievedGoal(Pr,G,S).

poss(resumeProjectProgress (Pr),S) :-
      noFeasibleProject(Pr,S),createPlan(Pl*,Pr,S).

poss(abortProject(Pr),S) :- noFeasibleProject(Pr,S), not
      createPlan(Pl*,Pr,S); closedProject(Pr,S).

poss(finishProject(Pr),S) :- projectOnSchedule(Pr,S),
      finishedProjectPlan(Pl,Pr,S), achievedGoal(Pr,G,S);
      abortedProject(Pr,S).

%Initial situation

closedProject(AC2990,s0). projectOnSchedule(sbb-1,s0).
      taskOnSchedule(t4,pl1,sbb-1,s0). taskProgress(t11,pl3,sbb-1,80,s0).
      taskResourceConsumption(t11,pl3,sbb-1,98,s0).
```

Figure 3. . An extract of the knowledge base implemented using ECLiPSe Prolog

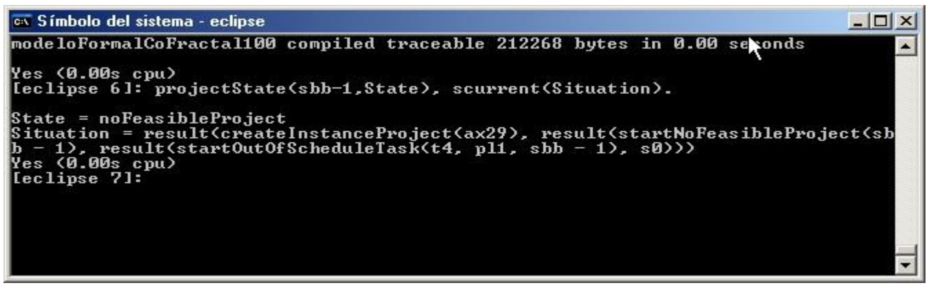
from the knowledge represented in it. Accordingly, the competency questions define the scope and objectives of the enterprise model while allowing its validation and correctness. In order to assess the fractal company model utility, the executable enterprise model must be instantiated for a particular enterprise network, in this case a fractal company dedicated to the discovery and development of new drugs.

The executable model of the fractal company allows answering to factual queries, such as "who is the SBB-1 project manager in the initial situation?" or "Is the ETC3-126 project in progress in the initial situation?" whose answers are obtained from the very information represented explicitly in the model itself, but also can infer answers for queries referred to the situation dynamics of the fractal company. The following paragraphs present some examples of commonsense queries about a project-based fractal company situation.

Query example 1: A project manager may need to know its project state in the current situation. This would be expressed as

ProjectState(SBB-1,State,result(createInstanceProject(ax29), result(startNoFeasibleProject(SBB-1,result(startOutOfScheduleTask(t4,pl1,sbb-1),s0)))).

The result is shown by Figure 4(a).

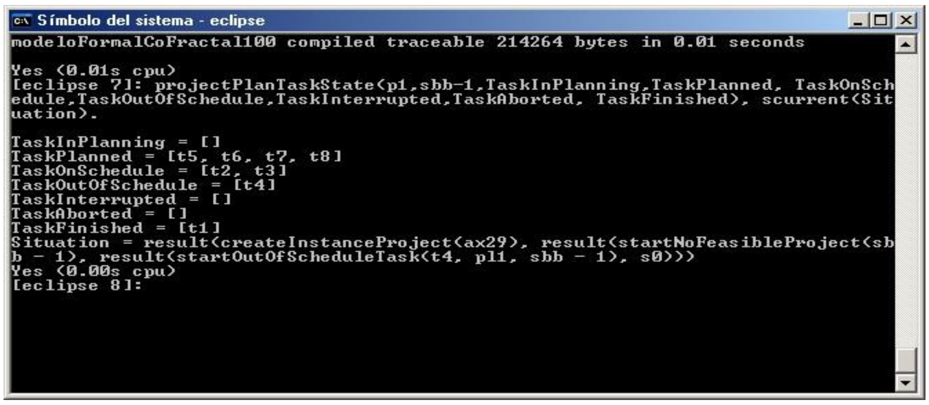


```
modeloFormalCoFractal100 compiled traceable 212268 bytes in 0.00 seconds

Yes (0.00s cpu)
[eclipse 6]: projectState(sbb-1,State), scurrent(Situation).

State = noFeasibleProject
Situation = result(createInstanceProject(ax29), result(startNoFeasibleProject(sb
b - 1), result(startOutOfScheduleTask(t4, pl1, sbb - 1), s0)))
Yes (0.00s cpu)
[eclipse 7]:
```

(a)



```
modeloFormalCoFractal100 compiled traceable 214264 bytes in 0.01 seconds

Yes (0.01s cpu)
[eclipse 7]: projectPlanTaskState(p1,sbb-1,TaskInPlanning,TaskPlanned, TaskOnSch
edule,TaskOutOfSchedule,TaskInterrupted,TaskAborted, TaskFinished), scurrent(Sit
uation).

TaskInPlanning = []
TaskPlanned = [t5, t6, t7, t8]
TaskOnSchedule = [t2, t3]
TaskOutOfSchedule = [t4]
TaskInterrupted = []
TaskAborted = []
TaskFinished = [t1]
Situation = result(createInstanceProject(ax29), result(startNoFeasibleProject(sb
b - 1), result(startOutOfScheduleTask(t4, pl1, sbb - 1), s0)))
Yes (0.00s cpu)
[eclipse 8]:
```

(b)

Figure 4. The knowledge base shows (a) the SBB-1 project state and (b) the task state of the SBB-1 project

In relation to the previous answers from the knowledge base, the project manager could require to know what is the cause making its project non feasible. This is expressed as

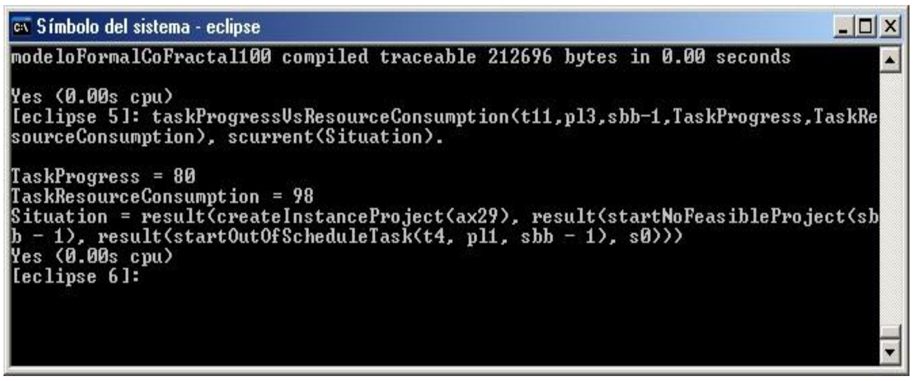
ProjectPlanTaskState(SBB-1,pl1,result(createInstanceProject(AX29),
result(startNoFeasibleProject(SBB-1,result(startOutOfScheduleTask(t4,pl1,SBB-1),s0)))).

Fig. 4 (b) shows the answer to this query. As it is highlighted in this figure, task 4 "to produce batches of compound SBB-1 to carry out the clinical trials" is behind the original schedule which makes the project non feasible. This abnormal situation has the following explanation. The clinical trials of SBB-1 are being very successful. As a result, the demand for more batches of the SBB-1 compound used in clinical trials is increasing, but pilot plant production cannot respond fast enough and delays arise in ramp-up production.

Query example 2: A project manager may need to know, for a given task, which are the task progress and the percentage of resource consumptions regarding planned usage in the current situation. This would be expressed as:

TaskProgressVsResourceConsumption(t11,pl3,sbb-1,%Progress,%Resource,
result(createInstanceProject(AX29),result(startNoFeasibleProject(SBB-1,
result(startOutOfScheduleTask(t4,pl1,SBB-1),s0)))).

The result of this query is shown by Fig. 5.



```
modeloFormalCoFractal100 compiled traceable 212696 bytes in 0.00 seconds

Yes (0.00s cpu)
[eclipse 5]: taskProgressVsResourceConsumption(t11,pl3,sbb-1,TaskProgress,TaskRe
sourceConsumption), scurrent(Situation).

TaskProgress = 80
TaskResourceConsumption = 98
Situation = result(createInstanceProject(ax29), result(startNoFeasibleProject(sb
b - 1), result(startOutOfScheduleTask(t4, pl1, sbb - 1), s0)))
Yes (0.00s cpu)
[eclipse 6]:
```

Figure 5. The knowledge base shows the task progress and the resource consumption percentages in a given situation

Query example 3: A resource manager needs to know for a given resource which resource commitments are not going to be fulfilled during a time period. This may be due to a fault causing a resource is out of service due to an unplanned maintenance service in the current situation. This query is expressed as follow:

AffectedCommitments(resourceTX,AffectedProject,ProjectManager,From,Since,AffectationLevel,result(createInstanceProject(AX29),result(startNoFeasibleProject(SBB-1, result(startOutOfScheduleTask(t4,pl1,SBB-1),s0)))).

The answer from the knowledge base is that there exist two affected commitments (tasks) by the current resource situation.

4. The ProFCo Architecture

The ProFCo (Project-Based Fractal Company) architecture is proposed for designing information systems to implement the project-based fractal company model. This architecture considers all requirements defined by the integrated enterprise model in Section 2. ProFCo is made up of a server module (ProFCo Server) and several client modules (ProFCo Clients) which communicate over the internet (Fig.6).

The ProFCo Server is a public module and responsible for:

- Registering integrated companies of the fractal company and the actors who will play the ends or means roles.
- Administering user names, password and restrictions of user access.
- Registering a file of client-server relationships established between project managers.
- Publishing information on project manager performance.
- Storing project data views that are acceded via Internet only by those users who have the corresponding permissions (father project, client project, member of the project). Thus, these users will be able to observe the progress of a given project, allocations of tasks and resources, consumptions of resources, costs, also to analyze the risks, to report progresses and abnormal situations.

- Storing in the catalog of the resources of the fractal company and to display complete information about them, with the aim of making agile the search and the contacts for the provision of them.
- An e-mail server that allows of the negotiation and communication among project managers and also it allows sending of messages about allocation of tasks, progresses, warnings, reminders.

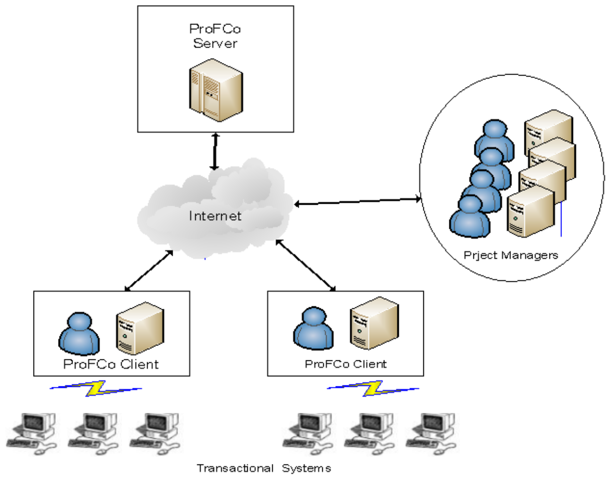


Figure 6. The ProFCo Architecture

The ProFCo Client is a private module for each one of the project manager. This module allows project manager

- to create, publish and manage his/her projects,
- to communicate with other members of the company fractal,
- to browse the resource catalog in the fractal company and thus to negotiate the provision of them.

In both server and client modules the decisions and actions carried out by project managers are registered in order to know performance of the project managers and to control that enterprise model constraints are not violated. For example, a project never could be published if it does not have a created plan; this is a plan with its defined attributes and restrictions, and the necessary designation of the people in charge for each task and resources have already assigned.

A prototype of the ProFCo architecture was implemented using the Microsoft Office Enterprise Project Management (EPM) SolutionTM1 and Visual NetTM. The project management tool EPM is a relatively low-cost, easy-to-use, flexible, and powerful enough for taking advantage of the Web and Intranets. This software tool allows an individual enterprise to optimize its resources, prioritize its works, align its project portfolio with overall business objectives and enables effective collaboration. Visual NetTM was used to create an interface between project actors and project-relevant

[1] MS Office Project Professional 2007, MS Office Project Server 2007, MS Project Web Access, MS Office Outlook 2007, SQL Server and Visual Net are trademarks of Microsoft Corporation.

knowledge and information bases, and thus, tailor EPM SolutionTM to meet specific project-based fractal company constraints and requirements. Visual NetTM was chosen to avoid difficulties of compatibility between applications. In the following paragraphs, the EPM SolutionTM architecture and the ProFCo architecture are presented and a brief description of each architectural component follows.

The EPM SolutionTM architecture is deployed across three tiers: a client tier, an application tier, and a database tier (Micro, 2013). Figure 6 shows the architecture of prototype ProFCo where components of the EPM Solution architecture (those whose names are not underlined) are included. The client tier includes Project Professional™ 2007, Project Web Access™ 2007 and Outlook™ 2007. Project Professional™ 2007 is a desktop application that is designed to enable project managers to create, publish, and manage projects. In addition to scheduling and tracking tools, it provides project managers with enterprise resources and portfolio management capabilities. Microsoft Project Professional™ can publish information to Microsoft Project Server™ and update information from Microsoft Project Server™ into project plans. Project Web Access™ is a web-based client that is designed for users who are not project managers, such as team members. Project Web Access™ provides access to timesheets, project views, status reports, document libraries, and risks. Project Web Access™ uses Internet Explorer™ to access project and resource information on Project Server™, view updates and analyze information about projects and resources. With Outlook™ users can receive e-mail reminder notifications for tasks that they are assigned in projects that are stored in the Project Server™ database.

The Application tier includes the central component of an EPM Solution: Project Server™ 2007. It is a web-based server application that integrates with several client applications. It provides both workgroup and enterprise project management features to client applications. When a project manager publishes a project, it is available to other project managers and project stakeholders.

In an EPM Solution™, the Data tier is based on the SQL Server™. This tier manages and stores project-related data consisting of several sets of database tables: The Project DB is a set of tables with project data that project manager access by using Project Professional™, and the Project View is a set of tables that represent a rationalized view of each project data that is contained in the project database and they are accessed by means of Project Web Access™.

Also, the prototype of the ProFCo architecture adds new components in each tier in order to tailor the commercial solution to meet all the project-based fractal company requirements. To achieve them were necessary to create a new resource perspective that allows separating resources management and allocation functions from resources utilization functions and establishing client-server relationships through negotiation among project managers (Contract module - Client tier - and Negotiation and Contract module – Application tier-). In addition, project managers have a browser that allows them to find information about the fractal company resources and to open the negotiation for the resource allocation with their respective managers (Resource Catalog module – Application tier-). To this aim, the Resource DB and Client-Server DB databases were included in the Data tier.

Also, there exist a web-based application that allows enterprises to register actors as potential project or resource managers, to access, analyze and update information about actors, roles, and their performance (Project-Manager Registry module – Client tire). Furthermore, this application is responsible for authenticating usernames, passwords and access permissions of such actors, being allowed them to carry out queries and operations only on those projects for which they are qualified (User Administration and Control module – Application tier -). In order to do this, the Actor DB database that stores all information about actors and roles are included in the Data tier.

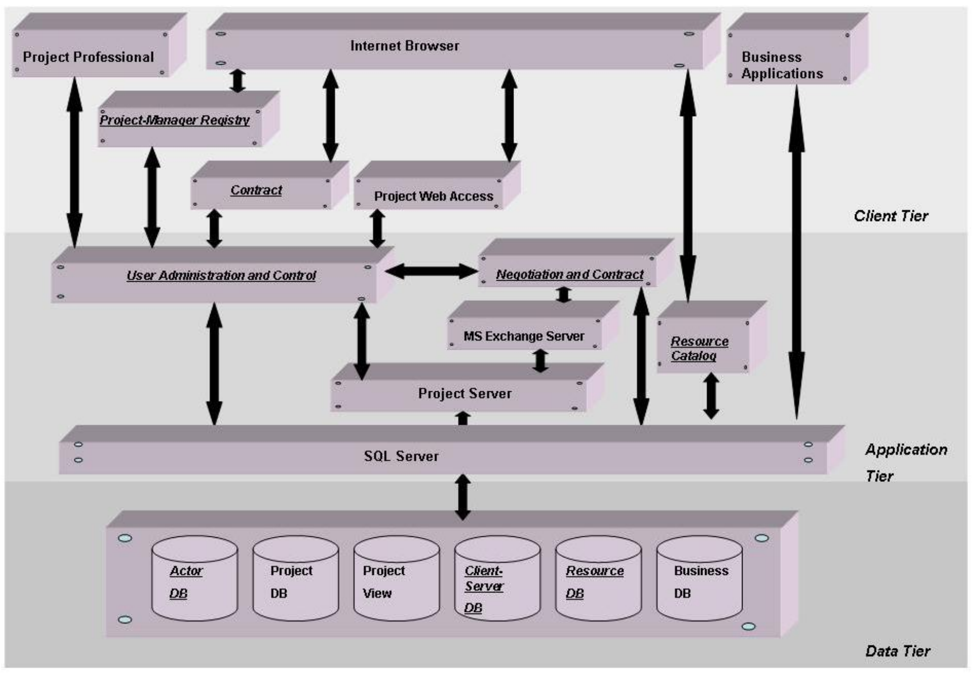


Figure 6 The prototype of the ProFCo Architecture

5. Conclusions

A formal fractal company for inter-firm networking has been proposed. Each project is an independently acting self-similar unit within the network. The key to the project-based fractal company is establishing client-server relationships between project managers that allow a more effective device for collaborating in a competitive environment. Formal enterprise modeling is based on the formalism of situation calculus. This formalism allows representing the multiple/relationships among projects, resources, and actors at different levels of abstraction over time. The set of axioms representing the SMEs networking dynamics is implemented in the logical programming language ECLiPSe Prolog. The execution of model simulation allows us to describe and analyze emergent behaviors and constraints as well as elaborated queries to the fractal company knowledge base. A prototype of the project-fractal company information system was developed using the EPM Solution of Microsoft Project 2007. This commercial set of project management tools was being tailored in order to describe the interactions between project managers according to the proposed model for enterprise networking.

References

[1] Canavesio, MM Martinez, EC.(2007) Enterprise modeling of a project-oriented fractal company for SMEs networking. Computers in Industry. Vol 58. Pp 794-813.

[2]Basole, R.C., Rouse,W.B., McGinnis, L.F., Bodner, D.A., Kessler, W.C.(2011) Models of Complex Enterprise Networks. Journal of Enterprise Transformation. 1:3, Pp 208-230.

[3] Warnecke, H.J..(1993) The fractal company: a revolution in corporate culture. Springer-Verlag. Berling.

[4] Ramanathan, J. Fractal architecture for the adaptive complex enterprise. (2005) Communications of the ACM. Vol 48. No 5. Pp 51-57.

[5] Kirikova, M. Towards flexible information architecture for fractal information systems.(2009) I International conference on information, process and knowledge management. IEEE Computer Society.

[6] Bider,I., Perjons, E., Elias, M.. (2012) Untangling the dynamic structure of an enterprise by applying a fractal appoach to business processes. Proceedings of PoEM

[7] Fox, M., Gruninger, M.. (1998) Enterprise modeling. AI Magazine, AAAI Press, Pp.109-121.

[8] Hoverstadt,P. (2009). Fractal organization: creating sustainable organizations with the viable system model. John Wiley & Sons.

[9] Koubarakis, M., and Plexousakis, D.. (2002). A formal framework for business process modeling and design. Information Systems. Vol. 27 No 5, Pp229-319.

[10] Chapurlat,V., Kamsu-Foguem,B., Prunet,F.(2006) A formal verification framework and associated tools for enterprise modeling: Application to UEML. Computer in Industry. No 57.Pp 153-166.

[11] Chapurlat,V., Braesch,C..(2008) Verification, validation, qualification and certification of enterprise models: statements and opportunities. Computers in Industry, No 59, Pp.711-721.

[12] Jonker, C.M., Sharpanskykh,A., Treur,J., Yolum,P,.(2007) A Framework for formal modeling and analysis of organizations. Appl.Intell. No 27. Pp. 49-66..

[13] Mueller,E.T. (2006) Commonsense reasoning. Morgan Kaufmann Publishers.

[14] Reiter, R.. (2001) Knowledge in action. Logical foundations for specifying and implementing dynamical and implementing. MIT Press. Massachusetts Institute of Technology.

[15] Scherl, R.B., and Levesque, H.J. (2003) Knowledge, action, and the frame problem. Artificial Intelligence. No 44, Pp. 1-39.

[16] Niederlinski, A., (2011) A quick and gentle guide to constraint logic programming via ECLiPSe. Jacek Skalmierski. Computer studio. ISBN 978-83-62652-08-2.

[17] tech.microsoft.com/library. 01/07/2013.

[18] Kephart, J., Chess, D. (2003) The vision of autonomic computing. Computer. Vol 36 No.1. IEEE Computer society. Pp 41-50.

Bork,D., Fill, H. (2014) Formal aspects of enterprise modeling methods: a comparison framework, 47st Hawaii International Conference on System Sciences. Pp 3400-3409.