A Contextual Approch For The Requirements Engineering Process

Colette Rolland

Université de Paris 1 Panthéon Sorbonne
UFR06, 17 Rue de la Sorbonne
75231 Paris Cedex 05
rolland@masi.ibp.fr

Abstract

Requirements specification methods and techniques have hitherto mainly been concerned with promoting various representation formalisms for formally describing information systems. However the emphasis on system modelling is shifting to process modelling. Our concern is the modelling of the Requirements Engineering (RE) process. The particular RE process modelling approach presented in the paper emphasises the notion of RE decision within the context in which it is taken. The context of decision is defined as a means to proceed locally as well as globally in the RE process. The paper concentrates on the presentation of the contextual approach and its illustration through examples extracted from the case studies developed in the Esprit project F3[*] where this approach is defined and experienced in user companies projects.

1. Introduction

It is nowadays commonly agreed that the early stage of the software development life cycle called Requirements Engineering, plays an important role in the delivery of an accurate system with the required level of quality. Furthermore requirements engineering is perceived as an area of growing importance (see for example the new series of conferences on Requirements Engineering, ICRE 92 and 93). The term requirements engineering has been introduced by E.Dubois and J. Hagelstein [1] to refer to this part of the software life cycle which consists of investigating the needs and requirements of the users community and abstracting from that the formal specification of the system to be built.

It is noticeable that most of the research efforts have concentrated on the product of this phase i.e the formal specification. This has caused the development of a number of modelling techniques and especially conceptual modelling techniques [2], [3] by which the system can be modelled in high level conceptual terms.

In contrast very little attention has been paid to the RE process i.e the route to reach the conceptual specification. In [4] the RE process is defined in a three dimensional space constructed over the representation, specification and agreement dimensions. The RE process is stated as getting from the initial input to the desired output and traced as an arbitrary curve within the cube spanned by the three dimensions. The initial input is characterized as opaque and personal views of the system represented using informal languages, whereas the desired output is a complete specification of the system, formally described, on which agreement has been gained. Despite the undeniable progress in recent years to provide answers to some of the issues araising along the three dimensions (and particularly those along the representation dimension) the most unanswered challenging issue in RE seems to be "how to proceed from informal, fuzzy, individual statements of requirements to a formal and complete specification that is agreed upon by all stakeholders".

The process modelling approach presented in this paper is an attempt to address this question.

Both, recent in-depth studies of RE practices in industry [5] and research works (see for example the special issue of the IEEE Trans. on Soft. Eng. on "Knowledge Representation and Reasoning in Software Engineering" and proceedings of the International Conferences on the Software Process) converge to the central idea that process modelling is as important as system modelling is. For lack of process modelling, understanding what the RE process is, what happens during it, when it happens, on what it happens, by whom it is performed is very poor. The process semantics are not well captured, with the required level of granularity in existing process models.

Consequently the way of working of methodologies is badly defined and the derived CASE tools are efficient in recording, retrieving and manipulating system specifications but are almost unable to actually support the creative activities of constructing and transforming the system specifications.

What actually happens when developing an IS is not recorded. Tracing in a stuctured manner the history of an IS development, keeping track of each transformation of the product, of what occured and when, is almost not possible without an adequate process model. Moreover capturing process knowledge is important to reason about the consequences of changing conditions and requirements in systems analysis, to reuse RE decisions in subsequent projects and improve the way of working of the requirements engineering team.

The need for a new modelling approach for requirements engineering motivates the process stream in the ESPRIT project called F3 (From Fuzzy to Formal). F3 is an ESPRIT III project aiming at defining a methodology that integrates into a coherent whole a selection of adequate techniques, all aiming at improving the route from informal, fuzzy and individual statements to complete specification of the IS which is formally expressed and agreed upon by all the stakeholders.

The particular RE process approach chosen in F3 assumes that the *basic building block of the RE process* can be modelled as a *4 - tuple <situation, decision, argument, action>*. It associates the situation the

[*] This work has been partially supported by the ESPRIT F3 (From Fuzzy to Formal) project n° 6612.

requirements engineer has to deal with to one of the decisions he can take to solve the local problem, the argument that motivates the decision and the actions to be performed in order to apply the decision. This view emphasizes the dimension of decisions and introduces as a main concept of process modelling the notion of context. A context is a couple <situation, decision> which amplifies the decision semantics by refining the "when" part of the decision.

The 4-tuple concept views the process in the small; it allows to model the process at a microscopic level. In addition, the approach proposes the notion of *context-dependency* as a means to model the *global RE process*. We use a spiral allegory to represent the process "in the large". One turn of the spiral is concerned with only one context. The progression along the spiral, from turn *a* to turn *b* results of the emergence of a new context to work on in *b* as a consequence of the decision making process performed in *a.*

The concept of context is central to our approach : it allows both to proceed locally (performing the decision related to the situation arised in the context) and globally (through out the dependency between contexts).

The paper is organised as follows. Our process modelling approach is introduced as part of the state of art in section 2. In section 3 the basic building for the RE process, namely the 4-tuple <situation, decision, argument, action> is presented. The global view of the RE process is sketched and discussed through examples in section 4. Our conclusions are drawn in section 5.

2. Process modelling : State of Art

The term process is heavily used in different contexts but has most often, in SWE (Software Engineering) and ISE (Information System Engineering) the following meaning "a related set of activities conducted to the specific purpose of the product definition" [6]. The rest of this paper will lay on the definitions proposed by Feiler and Humphrey in their attempt to define a core of concepts about the software process [7].

A *process* is defined as "a set of partially ordered steps intended to reach a goal". A *process step* is "an atomic action of a process that has no externally visible sub-structure. The process step is the basic process abstraction".

The term *process model* will be used to refer to "an astract representation of a process architecture (a conceptual framework for consistently incorporating, relating, and tayloring process elements into enactable processes), design (an embodiment of a process architecture) or definition (an implementation of a process design in the form of a partially ordered set of process steps that is enactable)".

The prevalence of this process view in most of development approaches has resulted in a number of process models which can be according to [8], referred to as, "Activity-oriented models".

Activity-oriented models [9], [10], [11] come from an analogy with problem solving, i.e. finding and executing a plan of actions leading to the solution. They are sequential in nature and provide a frame for manual management of projects developed in a linear fashion. Such a linear view of the design process is inadequate for methodologies which support backtracking, reuse of previous designs and have to support parallel engineering activities.

Dowson introduces two other classes of process-models, namely the "Product-oriented models" and the "Decision-oriented models".

The product-oriented process models [12], [6] represent the development process through the evolution of the product. They promote a view of development processes which is still centred around the notion of development activity but emphasises the linkage between the development activities and their output, the product.

A more recent class of process models follows a *decision-oriented paradigm*. The successive transformations of the product are looked upon as consequences of decisions. The process model of the DAIDA project [13]; [14] and of [15] fall into this category. Such models are semantically more powerful than the previous ones because they explain not only how the process proceeds but also why transformations happen. They provide a more complete knowledge about the process which is particularly useful for reuse of developed products, or of design decisions, and for backtracking purposes when a modification in requirements has to be considered either during the design process itself or during the maintenance of the information system.

To summarize we can say that a *step* (in the sense of Feiler & Humphrey) corresponds to an *activity* in "activity-based models", to a *product transformation* in "product-based models" whereas it is viewed as a *decision* which implies actions in "decision-based models".

We propose a contextual approach which consists in viewing a *process step* as a *context handling activity* and in modelling the *process* as a *sequence of dependent context handling activities.*

Indeed considering the highly non-deterministic nature of the Requirements Engineering process we believe that only the last of these three approaches appears to be partially appropriate. It is probably difficult to impossible to write down a realistic state-transition diagram that adequately describes what has to happen or actually happens in requirements engineering. But

relying on the pure artifact history is also insufficient. Even the decision-based approaches offer only limited hints about when and how to decide on what.
Analysts react *contextually* according to the domain knowledge they acquire and by analogy with previous situations they have been involved in. In order to take into account the very nature of the RE we have chosen to emphazise the contextual aspect of decisions. Our process modelling approach strongly couples the context of a decision to the decision itself, otherwise we loose information about the decision. This approach aims at capturing not only activities performed during the RE process but also why these activities are performed (the decisions) and when (the decision contexts).
The central aspect of our process way of modelling is that it makes the notion of *situation* (in which to decide) explicit and relates it to the broader question of context handling. Therefore a step in the RE process aims at processing a context : it is a context-handling activity. This is detailled in the next section.

3. Process modelling : the basic building block

This section aims at presenting the basic building block of our process modeling approach, i.e the one which corresponds to a process step. As introduced before we make te assumption that one step in the RE process consists of handling a context, i.e to take the appropriate decision in the right situation. Therefore the basic building block is centered around the notion of context and is composed of a set of interrelated concepts which contribute to the context definition. As illustrated in figure 1 it takes the form of a 4-tuple <situation, decision, argument, action>[16], [17], [18]:
The four key concepts are the ones of situation, decision, action and argument :
- *situation* to explain the decision context,
- *decision* to guide the RE process,
- *action* to perform the product transformations,
- *argument* to support decision-making.

A *situation* is most often a part of the product under development it makes sense to take a decision on. It represents the context of a decision. It can be a statement about the application domain or a requirement. It can also refer to part of existing products resulting from previous developments or a piece of generic domain knowledge. Accordingly, in figure 1, the situation *is built on* either the *product under development* or *existing products*. A complete product model is presented in [19].
A *decision* reflects a choice that a requirements engineer makes at a given point in time of the RE process. A decision can be looked upon as the intention of product transformations. It explicitly represents the resolution of an issue.
An *action* performs a transformation on the product. It is a materialisation of the decision. Performing it, changes the product and may generate new situations which, in turn, are subjects of new decisions.

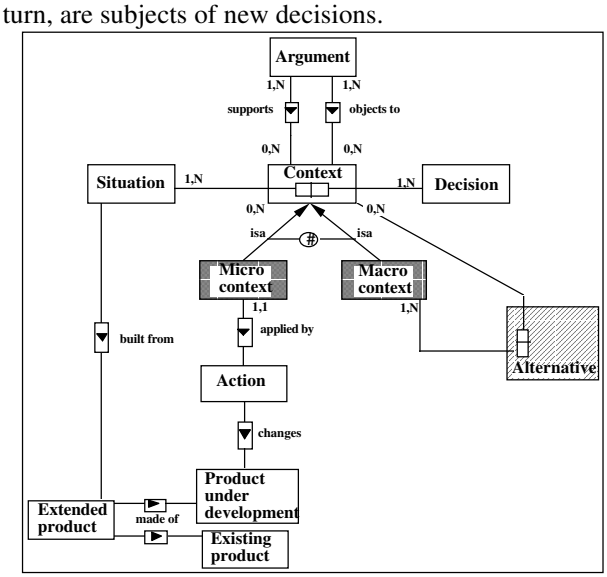


Figure 1 : Simplified overview of the process model

Arguments are statements which support or object to decisions within a given context.
These four concepts are exemplified in Figure 2a where we consider a step in the development of an E/R schema of a library system.

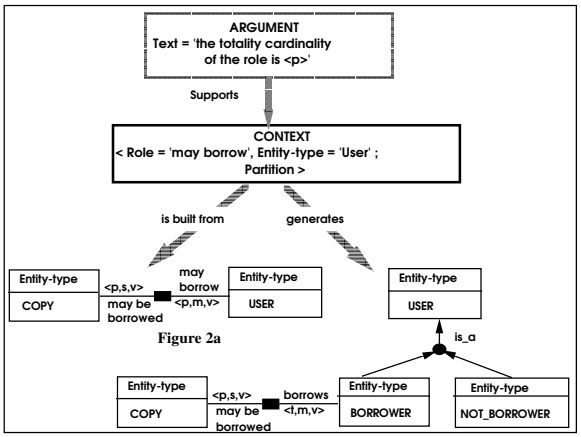


Figure 2a : A step in the development of an E/R schema

The context subject to this step is composed of the role <may-borrow> and the decision 'partition'. This decision is supported by arguments based on the cardinality <p> (for partial, meaning that a user may not be related to a copy of a book) associated to the role <may-borrow>. Indeed if borrowers are not always associated to copies of books it is worth to classify them into borrowers and not borrowers. The related action changes the product in

the way indicated in Figure 2a and raises a new situation, namely the role < borrows> which suggests the 'historise' decision. Thus, the new context is <[borrows]; historise>(Figure 2b) in which the decision to historise the loans of a given copy is supported by the argument based on the <v> cardinality (v for variable, meaning that copies are not always linked to the same subscriber). Applying the corresponding action transforms the product as shown in Figure 2b.

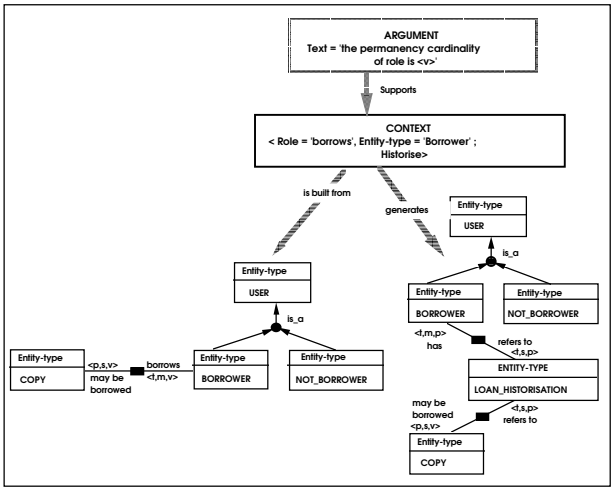
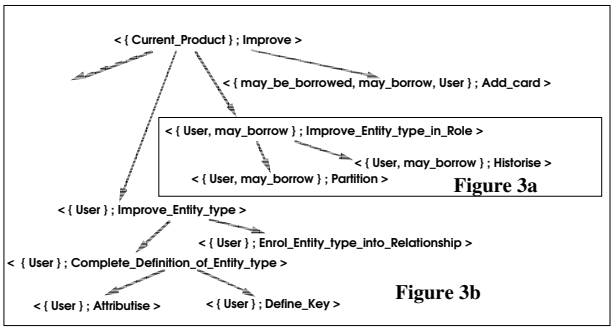


Figure 2b : Another step in the development of an E/R schema

Furthermore, the process model is intended to handle different levels of granularity of contexts, i.e. granularity of situations, as well as of decisions. Indeed for intance, requirements engineers are not always able to think in terms of executable decisions such as the former 'partition' or 'historise' decisions. They might sometimes express intentions such as 'refine a goal' or 'complete the schema description' which require further refinement before being implented through actions changing the product under development. This leads to the specialisation of contexts into *macro* and *micro* ones.

The former corresponds to a situation which requires to explore alternatives in decision making for its treatment. The refinement of a macro context is achieved by considering various alternatives all of which are relevant for the resolution of the decision of this context. This is modelled in terms of the decomposition of a macro context into other contexts and represented in Figure 1 by the loop upon context through macro context and alternative. As an illustration Figure3a shows the decomposition of the macro-context <[user, may-borrow]; Improve_Entity_type_in_Role > into the two contexts previously introduced. The two decisions 'partition' and 'historise' are two alternative ways to 'Improve_Entity_type_in_Role'

Decomposition can be performed iteratively. For instance the <[user, may-borrow]; Improve_Entity_type_in_Role> context can be itself regarded as one alternative of a more macroscopic context <[Current_Product]; Improve> as showned in figure3b. This introduces the notion of a *process chunk* as being a tree of alternatives which expresses the hierachical decomposition of an initial macro-context. The chunk is obtained recursively by connecting sub-trees, each of them corresponding to the decomposition of an alternative. The leaves of the chunk correspond to micro-contexts directly implementable through actions affecting the RE product. Thus the chunk encapsulates the hierarchical decomposition of an initial context allowing the refinement of a strategy till the discovery of the tactics through which it can be implemented.



On the other hand, when the context cannot be associated to multiple ways of solving the issue it raises then it is referred to as a *micro context*. Such a context implements one decision through execution of actions which transform the product under development. This raises new situations which are themselves subjects to decisions. This is shown in Figure 1 by the loop upon context through micro context, action, and situation. The two contexts considered in figure 2 are micro-contexts There are two major differences between the macro-context and the micro-context : the first one lies in the absence of any alternatives in the latter and the second one is that a macro-context has no direct consequence on the product under development.

A micro-context is only concerned with the manner in which the decision shall be implemented. If the consequence of the decision stated in the micro-context consists of a unique action, it is referred to as an *atomic context*. An *atomic context* triggers only one action In this case, the decision is directly implemented. Even though a commitment to implementing the decision has been made, it is possible that the implementation is complex requiring transformations of several elements of the product under development. In this case the context is said a *compound context.* A compound context is itself a pair <situation, decision> that has to be

further decomposed into more atomic contexts. This decomposition proceeds till atomic contexts are reached, which, by definition, are immediatly implementable. This is modelled through the decomposition of contexts ("composed of" relationship in figure1). For instance, the decomposition of the compound context<[user, borrows]; Partition > is as follows (figure 4) :
The micro-context <[user,borrows]; Partition> is further decomposed into seven atomic contexts including each, a sub-micro situation of the initial one and the corresponding micro-decision. Each micro decision is implemented through the execution of one action which creates or removes elements of the product under development. For instance the first atomic-type context is composed of the situation <borrower> and the micro-decision 'Borrower Creation' which results in the creation of the sub-type 'Borrower' of the Entity-type 'User'. The fifth atomic context associates to the two sub-typing links introduced to the Entity-type 'User', the decision of defining a partition constraint which is immediatly implemented by the correspondig action.

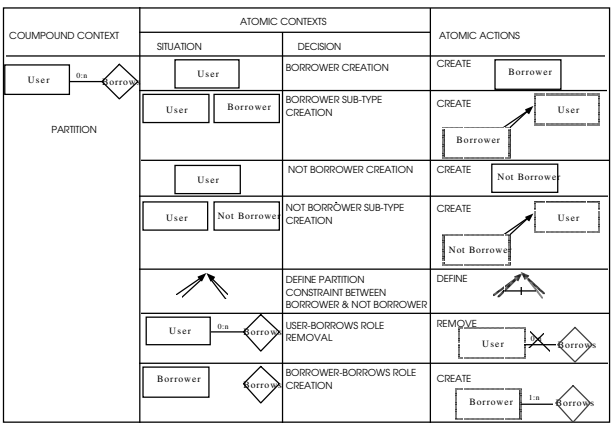


Figure 4: Example of a compound context

In this model the granularity in decision making relates to consequences of decisions. Some decisions have consequences that are only product transformations whereas others imply the taking of decisions as consequences. The former are referred to as *micro decisions* and the latter as *macro decisions* ; A macro decision can be qualified as a strategic one whereas the micro decision implements a tactic.
Consequently, the requirement engineering process is decision oriented and recognises two kinds of decisions, the strategic and the tactical. The former needs to be refined into tactical decisions through a certain number of steps of decision making. A tactical decision can be immediately implemented through a set of actions.
Wathever the nature of the decision, micro or macro, the associated context constitutes the basic building block of our RE process approach and its processing corresponds to a process step; we shall consider now the sequencing of steps in order to tackle the process "in the large".

4. Process modelling : the global view

4.1 The spiral view

We look to the RE process has being an incremental process allowing to build the RE product progressively through the adequate ordering of context handling steps. The global view of the RE process is depicted as a spiral (Figure 5). Each turn in the spiral corresponds to one problem solving activity resulting in some changes in the RE product The progression from one turn to another i.e the angular movement in the spiral, represents the incremental production of the RE product. The radial movement, i.e the unfolding, represents the extend of completness achieved by the Re product.
Both, the activity corresponding to one turn and the progression from one turn to another, are context driven. For example, as showned in Figure 5, turn1 (considering the Air Traffic Control (ATC) case study submitted to the F3 consortium by one of its industrial partner) is concerned with the decision 'Improve understanding' to be applied on the situation ' Traffic handling activity' which is a component of the entreprise model built as a part of the F3 product. As we will see later on, performing the decision will result in the creation of a new component 'goal safety' in the entreprise model and the emergence of a new context aiming at refining this goal that will be tackled in turn2.

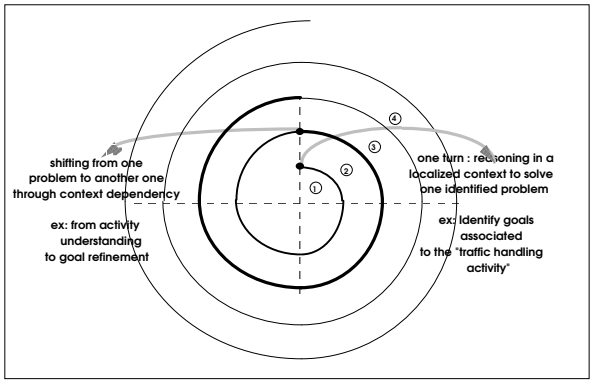


Figure 5 : A spiral view of the R.E. process

4.2 Structuration of a spiral turn

The input of each turn in the spiral is a context The activity performed in the turn aims at refining the decision associated to the input context till it can be implemented through atomic actions of product transformation.
In other words the turn is dealing with one process chunk. If the initial context is a macro-context the chunk

will provide the potential hierarchical decomposition of the input macro-decision. Therefore the process performed in one turn will consist of selecting the appropriate branch in the tree i.e choosing at each level of the tree, the adequate alternative till a micro-context is reached. Then the micro-context will be executed. If it is an atomic one this will result in only one action; if the micro- context is a compound one it will imply the execution of a sequence of atomic actions. The execution of actions lead in product transformations and raise new situations subjects to decisions.We propose that the spiral turn includes the selection of the next context to be worked on. This results in the structuration of the spiral turn into *four quadrants* as showned in Figure 6. Each quadrant corresponds to a phase in the process.

The *first quadrant* aims at selecting the adequate strategy for solving the problem raised by the input context of the spiral turn. This will be achieved by choosing among the alternatives proposed by the chunk associated to the input context.

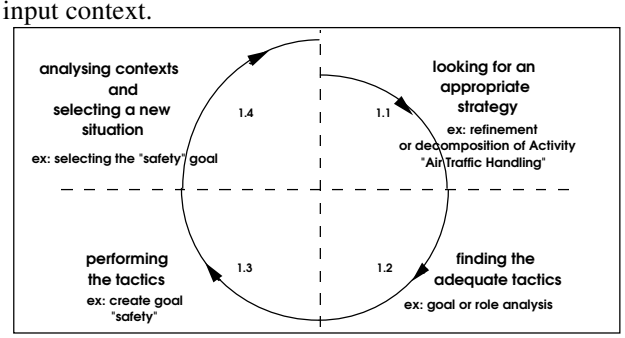


Figure 6 : Decomposition of a spiral turn

As an illustration Figure7a shows the decomposition of the context <[Air Traffic Handling]; Improve_Activity > into a number of macro-contexts among which are <[Air Traffic Handling]; Decompose_Activity> and <[Air Traffic Handling]; Refine_Activity>. This means that there are at least, two ways for improving our understanding of the Air Traffic Activity : one is to decompose the activity into sub-activities and the other one is to refine it through goal analysis. Let assume that the requirements engineer chooses the second of these alternatives.

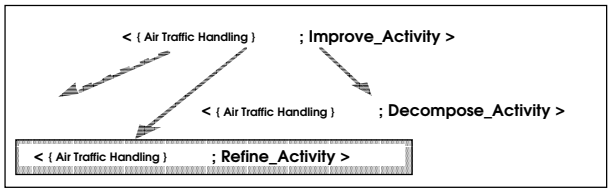


Figure 7a : Illustration of phase 1

In *quadrant 2* the macro-context resulting of phase1 will be decomposed further on in order to reach a micro-context. This will consists of selecting a branch in the hierachical decomposition of the contexts encapsulated in the chunk associated to the spiral turn. This is illustrated with the ATC case study in Figure7b.

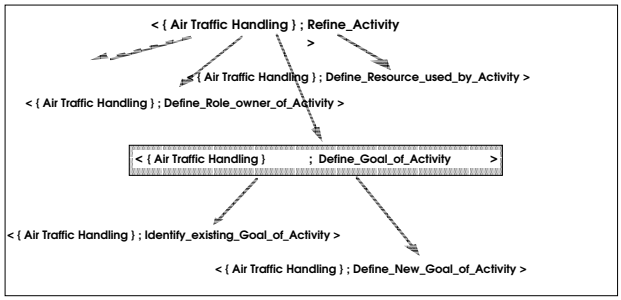


Figure 7b : Illustration of phase 2

There are a number of possible alternatives to handle the 'Refine_Activity' of the context <[Air Traffic Handling]; Refine_Activity>; Choosing the <[Air Traffic Handling]; Define_Goal_of_Activity> requires further refinment. Among the two suggested decompositions of this context we choose the <[Air Traffic Handling]; Define_New_Goal_of_Activity> which is a micro-contex that will be implemented in phase3.

Phase3 is tactical. It is concerned with the implementation of the decision of the micro-context resulting of the decomposition process performed in phase2. This is exemplified for the ATC case study in Figure 7c. The 'Define_New_Goal_of_Activity' decision requires four actions to be performed : the creation of the new component in the entreprise model, the goal ' Guarantee of a high degree of safety' and its relationships with the other components of the model.

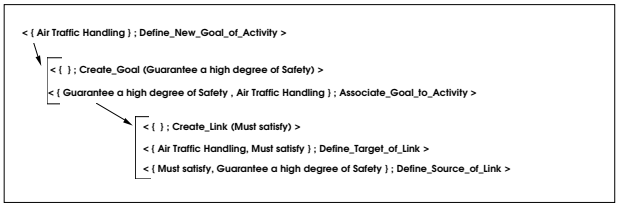


Figure 7c : Illustration of phase 3

Phase 4 has the objective to select the new context of working. Performing the actions resulting of phase3 raises new contexts among which one may be selected. In the example in hand the creation of the component 'Guarantee of a high degree of safety' generates a new situation to work on, namely the context <[Guarantee of high degree of safety]; Refine Goal> that will be the subject of next spiral turn.

4.3 Progression in the spiral

This example shows that the progression along the spiral, from one turn to another one is based on *context-*

dependency. Figure 8 extends the example to illustrate how the process can proceed from turn to turn through context dependency.

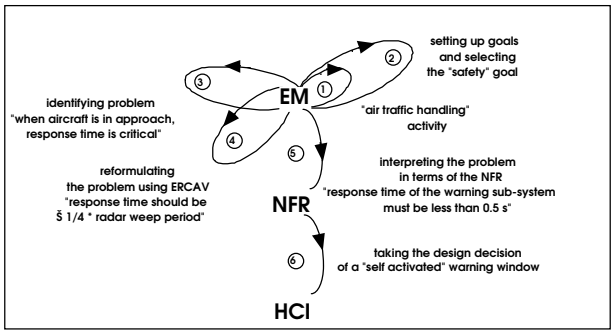


Figure 8 : Example of context dependency

Turn1 starts (as described in the previous section) with the problem of a better understanding of the activity-type component called 'Air Traffic Handling'.

Turn2 handles the context raised by turn1, namely the new goal-type component in the entreprise model : 'Guarantee of a high degree of safety'. Understanding the goal leads to the identification of a new problem-type component in the entreprise model called 'When aircraft is in approach response time is critical'. This will be performed in turn3.

The F3 set of tools includes an expert system aiming at elicitating the user requirements through cooperative work. In turn4 the so-called ERCAV tool will be used to reformulating the problem stated in turn3 into 'Response time should be less than 1/4 radar weep period'.

Thus, the output of turn4 is a new component in the entreprise model which suggests the decision to interpret the statement as a non-functional requirement for the system to be built. The result of this interpretation is the following non-functional requirement 'Response time should be less than 0.5 second' to be added as a new component of the non-functional model of the F3 product. This occurs in turn5.

Combining the decision to interpret the non-functional requirement in HCI terms to the situation consisting of the non-functional requirement itself defines the new context to be handled in turn6. The output of this turn is the design decision affecting the Human Communication interface : 'The window will be a self-activated warning window'.

Context-dependency is the key element of our global process approach. In order to guide the requirements engineer in his progression along the spiral we propose to predefine dependencies. Therefore, within a turn of the spiral the requirements engineer will be supported by the associated *chunk*; from one turn to another he will be supported by a *chunk cluster* grouping and ordering chunks which are dependent one from the other. This leads to the notion of a RE process guided through the use of a *chunk library*.

Criteria for clustering are of two kinds : *intention* (or goal) and *mode* (or reasoning drive).

We believe it is relevant to group chunks that together allow the application of the same *intention*. In the above example the intention was to interpret the knowledge acquired about the domain in terms of information system functional or non functional specification. Let's us consider, as another example, an object oriented method providing concepts such as 'class', 'event' and 'actor' and the intention to complete the schema. This intention is applicable through the specification, to all classes, all events and actors, i.e through the application of the three chunks, 'complete class', 'complete event' and 'complete actor'. These three chunks are thus gathered into a 'complete schema' cluster'.

Clustering will be dependent of the method in use. However one can think of a set or preferably a hierarchy, of generic intentions that could serve as entry point to define the adequate hierarchy of goals dedicated to a given method. Knowledge Representation in AI proposes such a list of so-called intentions expressed by verbs : elicitate, interpret, demonstrate, convince, justify, explain and decide.

The second criterion we propose for gathering chunks is the notion of *mode* or reasoning drive. We identified two pairs of modes : *depth-first/width-first* and *top-dodwn/bottom-up*. Keeping the example of an object oriented method, the way-of-working may either prescribe that all classes must be specified before the specification of the events and actors can be undertaken, or it may prescribe that classes, events and actors can be undertaken in parallel. In other words the process can proceed either in a depth-first manner or in a with-first one.

Similarly, the way-of working may either prescribe that classes must be specified by first creating them before defining their attributes and methods or it may prescribe that attributes must be defined first and thereafter grouped to create classes. Here, classes can be specified either in a top-down manner or in a bottom-up one.

We model the influence of modes in the process by characterising clusters not only by their intention but also by the particular combination of modes they promote. A combination of modes is a pair (m1, m2) where m1 is either depth-first or width-first and m2 is either top-down or bottom-up. In the ATC example the intention was, in short, to 'interpret' and the combination of modes was (depth-first , top-down). For a given intention, applicable by a same set of gathered chunks, there may be several clusters for several combinations of modes.

5 Conclusion

In this paper, we proposed a modelling approach for the RE process which intensively uses the notion of context to deal with the various levels of granularity of the RE process. The *atom* of process modelling is the 4-tuple <situation, decision, argument, action> which identifies the context of decision as the situation in which this decision occurs, recognizes the need for argumenting the decision, and models the outcome of the decision as actions transforming the product under development. Classification of contexts into micro-contexts concerned with the concrete effects of the decision and macro-contexts introducing alternatives in decision making leads to the process *molecule* , called process *chunk*. The chunk encapsulates the decision tree which progressively allows to refine a macro-context by decomposing it into more microscopic ones till micro-contexts are reached. Finally *Clusters* of chunks gather chunks with the same intention and a certain combination of modes. Clusters encapsulate knowledge for mastering process flows. They are the process *bodies*.

Processing a chunk corresponds to one step in the RE process and is graphically represented in our spiral view of the process as one turn of the spiral. Clusters support the progression from one turn to another one, i.e they help in the control of process flows. This conveys the idea of a RE process being guided by using a library of chunks and clusters to support the requirements engineer both locally and globally. Further work is mainly concerned with the specification and development of a guidance tool based upon the ideas presented in this paper.

6. References

[1] Dubois E.,Hagelstein J, Rifaut A. : "Formal Requirements Engineering with ERAE", Philips Journal Research, Vol 43, No 4, 1989.

[2] Olle T.W., Hagelstein J., MacDonald I., Rolland C., Van Assche F., Verrijn Stuart A. : "Information Systems Methodologies : A Framework for Understanding", Addison Wesley, 1988.

[3] Loucopoulos P., Zicari R.(eds) : "Conceptual Modeling, Database and Case", Wiley, 1992.

[4] Pohl K.; "The three dimensions of requirements engineering", Proc. CAISE'93 Conference, C. Rolland, F. Bodart, B. Pernici (eds), Sringer-Verlag (pub), 1993.

[5] Curtis B., Kasner H., Iscoe N. : "A field study of the software design process for large systems"; Comm. ACM, vol. 31, 1988.

[6] Peugeot C., Franckson M.: "Specification of the Object and Process Modeling Language", ESF Report n° D122-OPML-1.0, 1991.

[7] Feiler P.H, Humphrey W.S; "Software Process Development and Enactment: Concepts and Definitions"; *Proc. 2nd Int. conf. on the Software Process*, Berlin, Germany, 1993.

[8] Dowson M. : "Iteration in the Software Process"; *Proc 9th Int Conf on "Software Engineering*, Monterey, CA,1988.

[9] Royce W.W.: "Managing the Development of Large Software Systems"; Proc. IEEE WESCON 08/1970.

[10] Boehm B.W. : "A Spiral Model of Software Development and Enhancement"; IEEE Computer 21.

[11] Henderson-Sellers B., Edwards J.M.; "The Object-oriented Systems Life-Cycle"; Comm. ACM, Vol. 09, 1990.

[12] Finkelstein A., Kramer J., Goedicke M.: "ViewPoint Oriented Software Development"; *Proc. Conf "Le Génie Logiciel et ses Applications*, Toulouse, p 337-351, 1990.

[13] Jarke M., Mylopoulos J., Schmidt J.W., Vassiliou Y.; “DAIDA - An Environment for Evolving Information Systems”; *ACM Transcactions on Information Systems*, Vol. 10, No. 1, 1992.

[14] Rose T., Jarke M., Gocek M., Maltzahn C., Nissen H. : " A Decision-Based Configuration Process Environment", Software Engineering Journal, 6, 5.

[15] Potts C.: “A Generic Model for Representing Design Methods”; *Proc. 11th International Conference on Software Engineering*, 1989.

16] Grosz G. : " Using Artificial Intelligence Techniques to Formalize the Information System Design Process"; *Proc Int Conf on "Databases and Expert Systems Applications*, 1990.

[17] Rolland C., Cauvet C. : "ALECSI : An Expert System for Requirements Engineering", in "Advanced Information Systems Engineering", R.Andersen, J.A.Bubenko, A.Solvberg (Eds), Springer Verlag, 1991.

[18] Rolland C.: "Modeling the Requirements Engineering Process", *Proc. Fino-Japanese Seminar on "Conceptual Modeling*, 1993.

[19] Schmitt J.R.: "Product Modeling in Requirements Engineering Process Modeling", IFIP TC8 *Int. Conf. on "Information System Development Process*, Prakash N., Rolland C., Pernici B. (eds.), North Holland (pub.), 1993