A Framework for Generic Patterns Dedicated to the Management of Change in the Electricity Supply Industry

C. Rolland*, P. Loucopoulos**, G. Grosz*, S. Nurcan*

* C.R.I., UFR27, University Paris1 Sorbonne
90, rue de Tolbiac, 75013 PARIS France
{ rolland, grosz, nurcan}@univ-paris1.fr

** Department of Computation, UMIST
P.O. Box 88, Manchester M601QD, U.K.
pl@sna.co.umist.ac.uk

Abstract

This work describes a framework that is being used to fulfil one of the main objectives of the ESPRIT project ELEKTRA[1] : the discovery of the generalised patterns of change management for re-using them in similar settings in other electric supply companies. The term 'pattern' refers to such knowledge that may be repeatable from one situation to another, and shareable by many different users. This paper defines a framework for representing and using knowledge about the Electricity Supply Industry (ESI) sector. In the context of the ELEKTRA project, the patterns will be specific to the ESI sector and will not be applicable to any other sector. However, the framework proposed in this paper, for organising these patterns, is independent of any application and could potentially be used in many other domains.

1. Introduction

Recent years have witnessed an increasing interest in the use of patterns within the software development community and in particular by those advocating and practising object-oriented approaches and re-use. This recent interest in patterns has its origins in [5] and has subsequently permeated into software programming [3], [4], software and system design [8], [12], [21], data modelling [15] and more recently into systems analysis [10] What these efforts have in common is in their attempt to exploit knowledge about *best practice* in some domain. Best practice knowledge is constructed in 'patterns' that are subsequently used as the starting point in the programming, design or analysis endeavours.

The work on generic patterns presented in this paper addresses specifically one of the main objectives of the ESPRIT project ELEKTRA which can be stated as "*the discovery of generalised patterns of change management for re-using them in similar settings in other electric supply companies*". The successful completion of this objective will provide benefits beyond the confines of the two end-user applications that are currently being considered within the ELEKTRA project (one for 'distribution restructuring' and the other for 'human resource management' cases).

This paper presents a framework for the definition of the Electricity Supply Industry (ESI) generic models within the ELEKTRA project. Section 2 discusses the concept of a pattern in terms of its definition and its general use. Section 3 presents the ELEKTRA pattern framework in which the ESI specific patterns will be defined. Section 4 describes the process of using generic patterns. Finally, section 5 concludes with some observations about current work and future directions.

2. An overview of patterns

2.1. The notion of pattern

There are many, more or less similar, definitions of the term 'pattern' such as "… *a group of collaborating objects*" [12], "*... an idea that has been useful in one practical context and will probably be useful in others*", [10] "… *design patterns capture the static and dynamic structures of solutions that occur repeatedly when producing applications in a particular context*" [8].

Much of the contemporary work on patterns has been inspired by the work of Alexander on the use of patterns within the domain of architecture [1]. [2] presents the arguments for the discovery of patterns and their use for achieving quality of designs. Alexander defines a pattern as describing "*a problem which occurs over an over again in our environment and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing the same twice*".

[1] This work is partially supported by the ESPRIT project ELEKTRA (N°22927) funded the by the CEC in the context of the Framework IV programme.

In terms of enterprise development and change management, Coplien argues that "... *patterns should help us not only to understand existing organisations but also to helps us build new ones*" [7] , [9]. A further characteristic of the use of patterns is in their *generative* nature. That is, a good set of patterns should help to indirectly generate the right processes for understanding and developing organisations. According to the majority of proponents of the patterns movement, a pattern should be a self contained logical system which is capable of stating (a) that a given problem exists within a stated range of contexts and (b) that in the given context, a given solution solves the given problem. Different proposals are found in the literature for the description of the desirable properties for a pattern [2], [6], [7], [9], [20]. A synthesis of those properties may be the following:

♦ A pattern should be made explicit and precise so that it can be used time and time again. A pattern is explicit and precise if:

- It defines the *problem* (e.g. 'we want to deregulate a monopoly ESI company') together with the *forces* that influence the problem (e.g. 'customer needs are similar', 'pension rights for employees must be maintained' etc.);
- It defines a concrete *solution* (e.g. 'how buying and selling electricity is done');
- It defines its *context* (e.g. 'the pattern makes sense in a situation that involves the transition from a monopoly to a CBA model'). A context refers to a recurring set of situations in which the pattern applies.

♦ A pattern should be *visualisable* and should be *identifiable*. "Visualisation" may take the form of 'statements in natural language', 'drawings' conceptual models' and so on.

2.2. Pattern language

A *pattern* is an entity that is derived after empirical observation that a certain solution applies well to a recurring problem. A pattern relates to a *single* problem. It is expressed, using some form, to capture both the problem and the solution as well as the rationale for the applicability of the solution. However, a pattern is not an *isolated* entity. When a complex solution may not be describable in a single pattern or a single solution may be too specific and not shareable then the overall problem and its complex solution should be factored out into a number of problems and their respective solutions. The set of patterns, thus constructed, makes up a *pattern language*. Different patterns from a pattern language may be combined in different alternative ways to adopt different solution paths to different facets of the overall problem. Therefore, isolated patterns make sense only in small trivial problems. For problems beyond this trivial level we need to look at the relationships between the patterns in order to provide a more complete solution. Whilst a pattern provides the specifics applicable to a particular context, a pattern language represents a macro view of all the patterns applicable in a particular domain and application.

2.3. Panel of possibilities for pattern description

In order to support an easy navigation in a pattern language, with the aim of retrieving the appropriate patterns with regard to the problem at hand, we need some indexing mechanisms. According to [11], indexing methods can be broken into two main categories: controlled vocabulary and uncontrolled vocabulary. The former places limits on the terms than can be used to describe a classified object and/or on the syntax to be used to combine those terms.

The methods that have been used for nearly all fielded reuse library system are: enumerated, faceted and uncontrolled (free text) keyword. In *enumerated* classification a subject area is broken into mutually exclusive, usually hierarchical classes. The fact that they are so highly structured makes enumerated classification easy to understand and use. The hierarchy provides a natural searching method of navigating in the classification tree. The disadvantage of enumerated classification is that it requires that the indexing domain be completely analysed and broken into exclusive hierarchical categories. The classification scheme is difficult to change as the domain evolves. In *faceted* classification, a subject area is analysed into basic terms that are organised as facets. The development of facets is usually done by identifying vocabulary in a domain and then grouping like terms together into facets. A faceted classification scheme gives freedom to create complex relationships by combining facets and terms. It is much easier to modify than a hierarchical scheme because one facet can be changed without affecting others in the classification scheme.

Expressing the context of use of a pattern in a descriptive manner -even structured- does not allow an easy automatic querying and browsing of the repository of patterns. After a careful study of the state of the art synthesising the different possible indexing methods for pattern descriptions, we have chosen a faceted solution that will be developed in section 3.1.2.

3. The ELEKTRA pattern framework

3.1. Pattern template

Clearly, there is a need to make a difference between the *body* of a pattern and its *description*. The former is a model that is effectively reused whereas the latter aims to describe the context in which the body of the pattern can be reused. In ELEKTRA, we describe the knowledge encapsulated in patterns in terms of the EKD (Enterprise Knowledge Development methodology) concepts (i.e. enterprise goals, enterprise processes, etc.) [17].

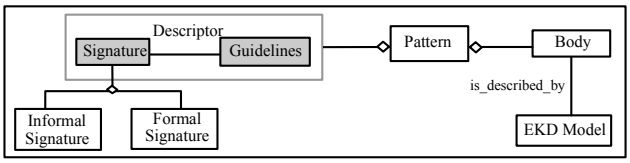

Figure 1 : The pattern template

3.1.1. The Body of a pattern. One of the main objectives of patterns is their transfer and communication between different projects or development situations, from the general case to the individual occurrence, their sharing between different people. An issue of concern therefore, is how to describe patterns so that this sharing can be effective. This question has been answered in terms of two possible alternatives: using *natural language* [9] or using *conceptual modelling* [10]. The former has the advantage of ease of transferability but falls short on formality. Lack of formality makes the use of patterns problematic and hampers the development of appropriate tools. The use of conceptual modelling languages overcomes these shortcomings. To date, work on patterns using conceptual modelling as the pattern language, has made use of existing, primarily object-oriented languages.

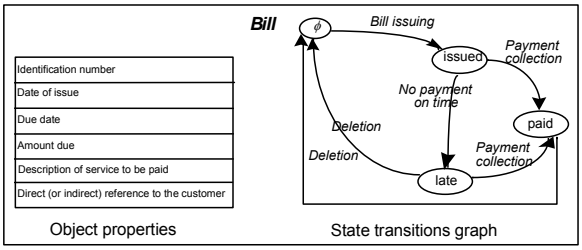

Figure 2 : An example of pattern having an EKD object sub-model fragment as body

In ELEKTRA, we propose a refinement of the conceptual modelling approach. The refinement is based in the qualification of details in the pattern in terms of the EKD concepts. Therefore, the body of a pattern is described in terms of an EKD model as shown in figure 1. Figure 2 shows the body of a pattern describing some generic properties and associated state transitions graphs of the objects of the class « Bill ». All bills across ESI companies have a set of common properties and state transitions. For a specific enterprise, these generic properties and state transitions will be adapted to the specifics of the business behaviour of the enterprise.

3.1.2. Descriptor. As stated in section 2, descriptors play a key role in the reuse process and for this reason it is very important to define them as accurately as possible. Our proposal is to have a pattern descriptor defined as an aggregation of a *signature* and *guidelines* (figure 1). The former describes in which situation it is relevant to reuse the body of the pattern whereas the latter are recommendations on the way the body of a pattern can be reused. A *signature* aims at describing the characteristics of a pattern, where it can be used, why, etc. A signature has a formal part and an informal part. As we will develop in section 5, *formal signatures* are used in the reuse process in order to retrieve patterns which are appropriate for a given situation having a given usage intention in mind. There are however, some additional requirements for describing patterns that are encapsulated in the *informal signature*.

For describing the *formal signature*, we have chosen to combine both a *faceted approach* with a *contextual approach* [14]. Accordingly, the formal signature has a *situation* part and a *usage intention* part (see figure 3).

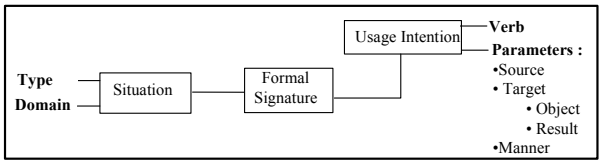

Figure 3 : The formal signature description

♦ The *situation* precisely describes the applicability conditions which must hold for re-using the generic pattern. It comprises two facets :

- The *type* of the pattern (Actor/Role, Role/Activity, Object, Rule, Goal, Change process patterns)(section 3.2).
- The *domain* describing the activity domain for which the pattern is applicable to (Customer servicing, restructuring, etc.).

♦ The *usage intention* expresses the goal to be achieved by the use of the generic pattern. It has two facets:

- A *verb* (invoice, change, ...), and
- A *set of parameters* described as a set of sub-facets: source, target and manner. Each parameter plays a different role with respect to the verb, some of them having sub-types (e.g. target has two sub types : object and result).

– The *target* designates what is affected by the usage intention. We distinguish two types of targets: objects and results. As opposed to *objects*, the *results* are affected by the usage intention. They do not exist prior to the usage intention.

$$\text{Measure}_{\text{verb}}\ (\textit{electricity consumption})_{\text{result}}$$

– The *source* identifies the origin of what is affected by the usage intention.

$$\text{Measure}_{\text{verb}}\ (\textit{electricity consumption})_{\text{result}}\ (\textit{from meter reading})_{\text{source}}$$

– The *manner* parameter is used to express in which way the usage intention is achieved.

$$\text{Measure}_{\text{verb}}\ (\textit{electricity consumption})_{\text{result}}\ (\textit{from meter reading})_{\text{source}}\ (\textit{automatically})_{\text{manner}}$$

The *informal signature* of a pattern is composed of mandatory components and optional components. The mandatory components are: the name of the pattern, the context of the pattern, the problems that it is trying to

solve, the constraints (forces) characterising the problems and qualified by the context (i.e. giving priorities, etc.), and the solution that solves the problem.

3.1.3. A complete example of pattern. Figure 4 shows an example of a generic pattern defined with respect to the pattern template described in figure 2. This pattern is independent of any particular application. It is domain dependent and it is applicable only in the ESI sector. This example deals with the particular problem of measuring the consumption of electricity by customers. This pattern is not put forward as the complete solution to this problem. Its purpose is to be illustrative of the concept of generic pattern. The pattern contains all the mandatory components specified in the pattern descriptor. The body of the pattern describes an actor-role model. It reflects the actors, their roles and dependencies regarding the management of electricity consumption through meter readings.

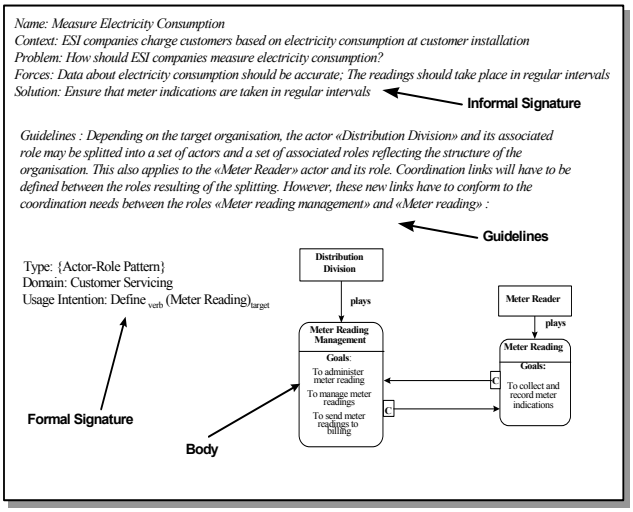

Figure 4 : An example of a complete generic pattern

3.2. Pattern Typology

In the proposed framework, we consider two types of *generic patterns*: (i) generic patterns dedicated to the modelling of the distribution and the human resource management in the ESI sector (called thereafter the *generic product patterns*), and, (ii) generic patterns tailored to change management of the distribution and the human resource management (called thereafter the *generic change process patterns*).

This approach addresses the twin requirements of (i) assisting in the development of EKD specifications and (ii) assisting in the management of change with the ESI sector. In the following, product patterns and change process patterns are detailed in turn.

3.2.1. Product patterns. The body of a pattern is described by the means of an EKD model fragment. The sub-models that are greyed in figure 5 will be used for ESI patterns. Therefore, the body of a pattern can be either a goal pattern represented with the goal model concepts or a business process pattern represented with the business process model concepts. In this framework, we do not tackle with IS patterns. A business process pattern is specialised in turn into actor/role pattern, role/activity pattern, object pattern and rule pattern, according to the corresponding sub-models of the EKD.

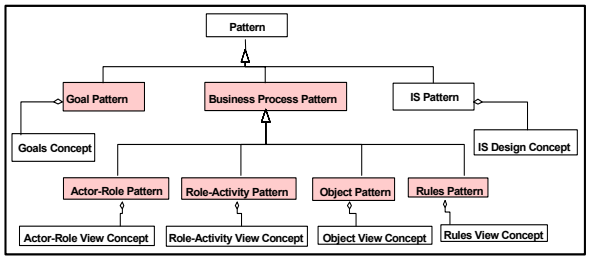

Figure 5 : Generic patterns according to EKD models

All these patterns describe ESI structural models and are referred to as *product patterns*. However, EKD addresses both the description of the business processes (and their associated goals) and the description of the change process itself. The former leads to product patterns whereas the latter introduces the needs for another type of pattern called, *change process patterns*.

3.2.2. Change process patterns. Similarly to the product patterns, the description of change processes might be done at a generic level in terms of change process patterns. Our proposal is to use an *extended goal model* as a means to represent the body of generic change process fragments. Following the semantics of an EKD goal graph, a change process pattern is represented as a goal hierarchy using the AND/OR connectors (see figure 6). In order not to confuse between company business goals and the company change goals, we decide to rename for the latter the concept of « goal » as « change intention ».

The proposed extension consists of expressing the top level change intention in its context as a triplet <initial situation, change intention, target situation>. The initial situation refers to the current state of the organisation whereas, the target situation refers to the target state. The two situations are described by EKD models. For instance, if an electricity company wants to change its structural model from monopoly to ISO (the change intention), it must be specified where the company is now (the initial situation, e.g. a monopoly situation represented in EKD models) and where it wants to go (the target situation, e.g. the ISO structural model also represented as EKD models). The leaves of a goal hierarchy are called *operationalisable change intentions* which correspond to intentions that do not require any further decomposition, it means that its realisation can be expressed in terms of one or several product patterns (i.e. a set of EKD models).

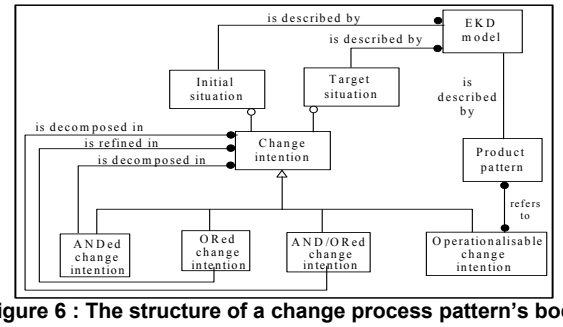

Figure 6 : The structure of a change process pattern's body

In summary, we use the EKD meta-model as a language for describing both the bodies of ESI specific generic patterns (being it product or change process pattern) and application specific EKD models. Application specific EKD models are also instances of the EKD meta-model as shown in figure 10. These instances can be the expansion or the refinement of the generic patterns as we will develop in section 5.

3.3. Pattern repository

3.3.1. Relationships among patterns and their descriptors. Since a pattern descriptor is intentional the proposal is to express the relationships using intention connectors. We identify three types of connectors, AND, OR (exclusive OR) and AND/OR (inclusive OR). Our belief is that this representation eases the retrieval of relevant patterns for a given situation. Indeed, the users of the repository can understand its contents by browsing through a hierarchy of goals (usage intentions of patterns, see § 3.1.2) that are meaningful and familiar to them. Figure 7 shows the relationships among formal signatures.

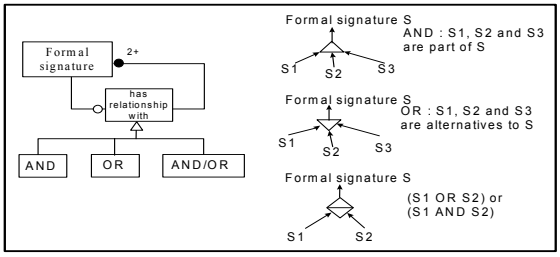

Figure 7 : Relationships among formal signatures

3.3.2. Repository organisation. Based on the introduction of the hierarchy of usage intentions made above, we can now introduce the structure of the repository of the ESI tool set, following our proposal. First, the repository is composed of two parts :

- the *patterns part* (defined at the knowledge level), and
- the *indexing hierarchy part* (defined at the meta-knowledge level).

The patterns part represents the ESI knowledge for a single problem. The indexing hierarchy part describes this knowledge in terms of the ESI domain goals that can be fulfilled by reusing the patterns. The hierarchical organisation of the pattern formal signatures helps structuring the problem in intentional terms that should be easily understood by the domain experts. It supports a top down approach for retrieving the appropriate patterns for a given situation in a given setting.

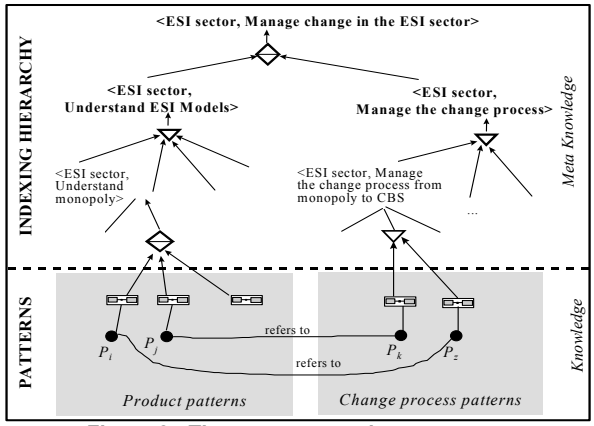

Figure 8 : The pattern repository structure

In figure 8, the top level usage intentions figures out what is the overall objective of the ELEKTRA project (see figure 8), namely to '*Manage change in the ESI sector*' and tells us that there are two problems in managing the change which are supported by pattern based solutions in the repository: '*Understand ESI models*' and '*Manage the change process*'. While browsing through the indexing hierarchy associated to the former usage intention, the possible paths lead to a set of product patterns, whereas while browsing through the hierarchy associated to the latter, the possible paths lead to change process patterns. The leaves of these change process patterns (expressed as change intentions graphs) make explicit references to product patterns (as shown in figure 6).

4. The reuse process

As illustrated in figure 9, the re-use process is a three steps process which consists of : 1) the retrieval and selection of the generic patterns from the set of generic patterns, 2) the storage of the selected generic patterns in the work space and 3) the customisation/expansion/ refinement of the generic pattern.

Step 1 shall be performed by the domain expert, through browsing facilities or queries. Queries are useful when the domain expert knows what he/she is looking for. Queries are based on the structure of the formal signature and use the thesaurus of the terms corresponding to the values of the facets. The browsing facilities are adapted to get a full picture of the repository contents and to select the appropriated patterns using a top-down approach. This step starts with the description, using the structure of the formal signature, of the properties of the generic pattern the domain expert is looking for. This description may not be complete with

regard to the structure of the formal signature. The domain expert is then facing the list of generic patterns that match his/her requirements and has has to select the appropriate ones. This selection process requires to browse the informal signatures of the retrieved generic patterns in order to determine the one that conforms his/her needs.

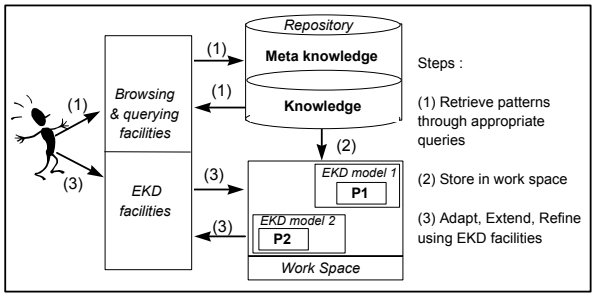

Figure 9 : The illustration of the reuse process

In step 2, the selected patterns are stored in a work space in order to be adapted to the application in hand in the following step. The generic elements of the selected patterns are tailored to the enterprise vocabulary. In step 3 the selected patterns are progressively adapted to the specific requirements of the case under study, and extended into EKD models corresponding to the desired solution. All these transformations and extensions shall be supported by EKD editors and tools. Figure 10 introduces the different levels of abstraction in pattern re-use.

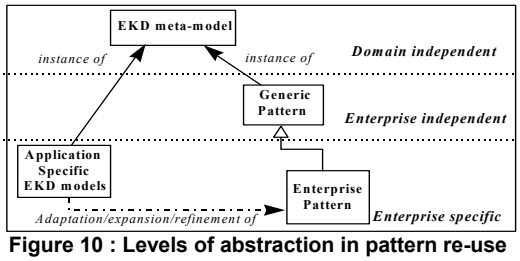

Figure 10 : Levels of abstraction in pattern re-use

5. Conclusions

A critical factor in being able to share best business practice (including best practice for system development) is the appropriate re-use of existing 'chunks' of best practice. Patterns provide the mechanism for achieving this.

This paper has attempted to provide a framework for defining generic patterns for the ESI sector. These patterns will have to be built by empirical observation and tested on a range of examples within the adopted project case studies. Prior to discovering patterns however in these case studies, there is a need to establish the framework for maintaining and using the knowledge pertinent to the patterns. This paper is an attempt to define such a framework. In this paper we advocate a formal approach to defining the knowledge to be re-used, in terms of (a) both product and process dimensions, thus fully addressing the issue of 'change management' and (b) the indexing of these patterns in such a way so as to maximise their potential for re-use in situations that demand an informed approach to change.

References

[1] Alexander C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. and Angel, S. *A Pattern Language*, Oxford University Press, New York, 1977.

[2] Alexander, C. *The Timeless Way of Building*, Oxford University Press, New York, 1979.

[3] Beck, K. *Smalltalk Best Practice Patterns*. Volume 1: Coding, Prentice Hall, Englewood Cliffs, NJ, 1997.

[4] Buschmann, F., Meunier, R., Rohnert, H., Sommerland, P. and Stal, M. *Pattern-Oriented Software Architecture - A System of Patterns*, John Wiley, 1996.

[5] Coad, P. *Object-Oriented Patterns*, Communications of the ACM, Vol. 35, No. 9, 1992, pp. 152-159.

[6] Coad, P. et al. *Object Models - Strategies Patterns and Applications* , Yourdon Press Computing Series, 1996.

[7] Coplien, J. *A Development Process Generative Pattern Language*, AT&T Bell Laboratories, http://www.bell-labs.com/people/cope/Patterns/Process/index.html, 1995.

[8] Coplien, J.O. and Schmidt, D.O. (ed.) *Pattern Languages of Program Design*, Addison-Wesley, Reading, MA, 1995.

[9] Coplien, J.O. *A generative Development - Process Pattern Language*, in 'Pattern Languages of Program Design', J. O. Coplien and D. O. Schmidt (ed.), Addison-Wesley, 1995.

[10] Fowler, M. *Analysis Patterns: Reusable Object Models*, Addison-Wesley, 1997.

[11] Frakes, W.B. and Pole, T.P. *An empirical study of representation methods for reusable software components*, IEEE Trans.s on Software Engineering, 20(8), August 1994.

[12] Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.

[13] Grosz, G. and Rolland, C. (1997) *Using the EKD Approach: Modelling the ESI Generic Patterns*, University Paris 1, (ELEKTRA Project), 20 November 1997, 1997.

[14] Grosz, G. , Rolland, C., Schwer, S., Souveyet, S., Plihon, V., Si-said, S., Ben Achour, C., Gnaho, C. *Modelling and Engineering the Requirements Engineering Process : An Overview of the NATURE Approach***,** Requirements Engineering Journal, (2), pp 115-131, 1997.

[15] Hay, D. *Data Model Patterns: Conventions of Thought*, Dorset House, New York, 1996.

[16] Loucopoulos, P. *The generic pattern model*, UMIST, (ELEKTRA Project), 31 December 1997.

[17] Loucopoulos, P., Kavakli, V., Prekas, N., Rolland, C., Grosz, G. and Nurcan, S. *Using the EKD Approach: The Modelling Component*, UMIST, (ELEKTRA project), March 1997.

[18] Rolland, C., Grosz, G., Nurcan, S. *Generic Patterns for the ESI Sector*, University Paris 1, (ELEKTRA Project), Jan. 1998.

[19] Rolland, C., Grosz, G., Nurcan, S., Yue, W., Gnaho , C., *An electronic handbook for accessing domain specific generic patterns.* IFIP WG 8.1 Working Conference : Information Systems in the WWW environment, 15-17 july 1998, Beijing, Chine.

[20] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Loresen, W. *Object-oriented modelling and design,* Prentice Hall Englewood Cliffs, NJ, 1991.

[21] Vlissides, J.M., Coplien, J.O. and Kerth, N.L. (ed.) *Pattern Languages of Program Design 2*, Addison-Wesley, 1996.