An electronic handbook for accessing domain specific generic patterns

C. Rolland, G. Grosz*, S. Nurcan*, W. Yue**, C. Gnaho**
** Centre de Recherche en Informatique*
Université Paris 1 - Panthéon - Sorbonne
90, rue de Tolbiac 75013 Paris, France
Tel : 33 - 1 - 40 77 46 34 Fax : 33 - 1 - 40 77 19 54
E-mail : {rolland, grosz, nurcan, gnaho}@univ-paris1.fr

*** 610054 Computer Science & Engineering College, UESTC, Chengdu, Sichuan, China,*
Currently visiting professor at Université Paris 1

Abstract

This work addresses the issues of defining (a) the structure of a knowledge repository of good business practices for managing change in organisations and (b) a Web based tool for providing a simple means for accessing this knowledge. Following the proposal made within the ESPRIT project ELEKTRA[*], the expression of best business practices is done in terms of generic patterns. The term 'pattern' refers to such knowledge that may be repeatable from one situation to another, and shareable by many different users. The knowledge encapsulated in patterns is expressed in terms of the concepts of the EKD methodology (Enterprise Knowledge Development) such as enterprise goals, enterprise processes etc.. This paper presents the framework for representing the patterns and a set of mechanisms for navigating within the repository of patterns in a WWW environment.

Keywords

Generic Pattern, Reuse Process, Change process, WWW

---

[*] This work is supported by the ESPRIT project ELEKTRA (N° 22927) funded by the EEC in the context of the Framework 4 Programme

1 INTRODUCTION

The concept of pattern has been widely used in the software development community during the last 20 years in particular by those practising Object-Oriented (OO) approaches. It has been more recently introduced in the Information System Method Engineering community (ISME). All these efforts aim to exploit knowledge about *best practice* in some domain. Therefore, this concept is usually used in the context of reuse.

In the OO community, the focus has been put on patterns useful during the design of software and independent of any particular domain (Coad, 1992), (Beck, 1997), (Buschmann, *et al* 1996), (Coplien, Schmidt, 1995), (Gamma *et al*, 1994), (Vlissides *et al*, 1996), (Hay, 1996), (Fowler, 1997). In the ISME community, generic patterns are used for the sake of methodology oriented process modelling. They aim at proposing a means for constructing situation specific methods out of a set of fragments. These fragments are called method chunks (Rolland, Plihon, 1996), (Rolland, Prakash, 1996) or method fragments (Harmsen *et al*, 1994). Such patterns encapsulate knowledge about processes that can be reused and apply in different settings and allows to guide method engineers in the construction of methods.

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 "*to discover generalised patterns of change management for reusing them in similar settings in other electric supply companies*".

Different studies have shown that it seems possible that generic knowledge can be exhibited for the modelling of change in the Electricity Supply Industry (ESI) sector, e.g. (Yajima,1997). We propose to express this knowledge in generic terms and to model it with generic patterns. We consider two types of *generic patterns*:

- generic patterns dedicated to the modelling of the ESI sector what will be called thereafter the *product patterns*, and
- generic patterns tailored to change management in the ESI sector what will be called thereafter the *change process patterns*.

In ELEKTRA, we propose to describe the knowledge encapsulated in patterns in terms of the EKD (Enterprise Knowledge Development) methodology concepts (i.e. enterprise goals, enterprise processes etc.) (Loucopoulos *et al*, 1997). EKD is an approach that provides a systematic and controlled way of analysing, understanding, developing and documenting an enterprise and its components by using Enterprise Modelling. Its purpose is to provide a clear and unambiguous picture of (a) how the enterprise functions currently, (b) what are the requirements for change and the reasons for change (c) the alternatives that could be envisaged in order to meet these requirements and (d) the criteria and arguments for evaluating these alternatives (Bubenko et al. 1997).

The ELEKTRA project addresses the two requirements of (i) assisting in the development of EKD specifications (Rolland *et al*, 1997a), (Rolland *et al*, 1998a) and (ii) assisting in the management of change with the ESI sector (Rolland et al,

1997b). Accordingly, there are three aspects of work within ELEKTRA that need to be addressed:

- The definition of a framework for representing knowledge about the ESI sector, i.e. the *pattern template*, and the *structure of the pattern repository*.
- The definition of a tool for using this knowledge. In order to make the repository of knowledge accessible by any company of the ESI sector that is involved in change management, the tool is defined as a Web navigator over the repository.
- The definition of the knowledge itself, which is used in describing some aspects of the ESI sector.

This paper deals with the two first aspects and proposes a framework for representing knowledge about the ESI sector and a tool for using such a knowledge. In the context of this project, the patterns will be specific to the ESI sector and will not be applicable to any other sector. The framework however, for organising these patterns, is independent of any application and could potentially be used in other domains. The rationale is that all concepts defined and used within this framework are based on the EKD method which is totally domain independent. Furthermore, an important aspect of the ELEKTRA project concerns the dissemination of best business practices within the ESI sector companies. To this end, we encapsulate the knowledge describing the best business practices within a single repository and propose a simple tool that allows to access this knowledge over the Web.

The paper is organised as follows. Section 2 discusses the concept of a pattern, pattern template and pattern repository. The three next sections, respectively sections 3, 4 and 5, are dedicated to the presentation of the ELEKTRA pattern framework in which the ESI specific patterns will be defined. With this purpose, section 3 first sets the structure of a pattern. Then, section 4 defines a typology introducing the concepts of products and change process patterns within the framework. Because there is a need to combine different patterns in different alternative ways to adopt different solutions for a given problem, section 5 presents the structure of such a pattern repository. Section 6 describes the process of using generic patterns and presents the pattern repository navigator called the 'electronic handbook' through a scenario of use.

2 AN OVERVIEW OF PATTERNS

2.1 The notion of pattern

Much of the contemporary work on patterns has been inspired by the work of C. Alexander on the use of patterns within the domain of architecture (Alexander, 1979). In (Alexander *et al*, 1977) a pattern is described as "*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". Here, the emphasis is put on the fact that a pattern describes a recurrent problem and it is defined with its associate core solution.

In the context of object-oriented development, Gamma (Gamma *et al*, 1994) defines a design pattern as "*a description of communicating objects and classes that are customised to solve a general problem in a particular context*". In the same context, (Coad, 1992) proposes that "*an object-oriented pattern is an abstraction of a doublet, triplet or other small grouping of classes that is likely to be helpful again and again on object-oriented development*". From this definition, what should be kept in mind is that a pattern is an '*abstraction*' that can be '*helpful on object-oriented development*'. The term abstraction implies that a pattern is described differently than the classes it has been built from. As a matter of fact, a pattern is *generic* in the sense that it captures the commonalties between the classes at a generic level.

There are three key ideas concerning patterns which are embodied in these definitions. First is the fact that a pattern relates a *problem* to a *solution* to this problem. The second point draws our attention to the fact that what is important in pattern identification is not so much to recognise the commonalties among candidate elements but to identify the discriminant criteria. What is the key point in pattern identification is to find what is unique in a problem and makes it different from other similar problems. Finally, the genericity may be at different levels:

- A pattern can be used for different applications in a specific domain repeatable in different organisations/settings (application independent but domain specific). Example: generic patterns for ESI sector.
- A pattern can be used in different domains (domain independent pattern). Example: generic patterns for goal refinement.

2.2. Pattern template

For the pattern to be generic and reusable, the pattern template should be able to state (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. The pattern 'template' is a predetermined structure according to which the generic knowledge has to be represented. Therefore, a set of desirable properties for a pattern must include the *problem* for which the pattern propose a reusable knowledge, the *solution* describing this knowledge and the *context* of use. A context refers to a recurring set of situations in which the pattern applies.

In the literature we found different proposals for the description of those desirable properties. For instance, (Coad *et al*, 1996) defines the structure of a pattern as including information describing the pattern, examples of use along with information describing the relationships that the pattern has with other patterns. In this case, the abstraction takes the form of an OMT diagram describing interacting classes. New classes (or more precisely structures of class) will be generated out of this diagram. According to (Alexander *et al*, 1977), the structure of a pattern contains the following properties:

• *Name*, it should be short and as descriptive as possible
• *Examples*, one or several diagrams/drawings illustrating the use of the pattern,
• *Context*, it focuses on the situation where the pattern is applicable,
• *Problem*, it describes the major forces/benefits of the pattern and its applicability constraints,
• *Solution*, it details the way to solve the problem, it is composed of static relationships as well as dynamic ones describing how to construct an artefact according to the pattern.

This is very similar to the pattern structure proposed by Coad. The additions concern guidelines for using the pattern plus some information on the context of use of the pattern. The structure proposed in (Gamma *et al*, 1994) is the most complete proposed in the literature. The context as well as the relationships the pattern has with other patterns are emphasised, different means are used to describe the pattern and its possible use. The template also includes the explicit motivation for using the pattern (trade-offs and benefits).

2.3 Pattern repository

As specified in section 2.2, a pattern is expressed to capture both the problem and the solution as well as the rationale for the applicability of the solution. The solution described in the pattern relates to a *single* problem. Nevertheless, when we are faced with complex situations, as for example, when a complex solution may not be described in a single pattern or a single solution may be too specific and not reusable, then the overall problem and its complex solution should be factored out into a number of problems and their respective solutions.

Within ELEKTRA, the definition of the ESI generic patterns will constitute, in their totality, a pattern repository that will be designed to be shared by all stakeholders involved in change management projects within this sector.

In order to support an easy navigation in the pattern repository for retrieving the appropriate patterns with regard to the problem at hand, we need some indexing mechanisms (Frakes, Pole, 1994). After a careful study of the state of the art synthesising the different possible indexing methods for pattern descriptions (Rolland *et al*, 1998b), we have chosen a faceted solution that will be developed in section 3.2. The three following sections are dedicated to the presentation of the ELEKTRA pattern framework.

3 THE PATTERN TEMPLATE

This section sets the structure of a pattern, namely the *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; in our framework it is an EKD model fragment. The latter aims to describe the context in which the body of the pattern can be reused. Therefore, a pattern is completely defined with its descriptor and its body (see figure 1).

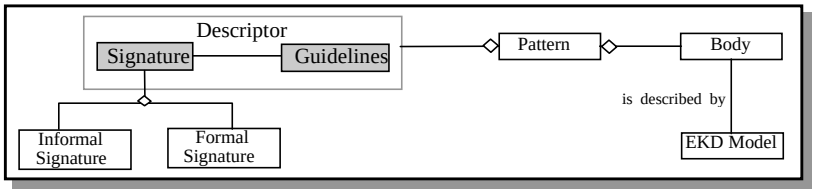

Figure 1 The pattern template

3.1 The body

One of the main objectives of patterns is their sharing between different people and/or situation. The effectiveness of the sharing depends on the language adopted for describing the reusable knowledge which is embedded in the pattern. The use of *natural language* (Coplien, Schmidt, 1995) has the advantage of ease of transferability but falls short on formality. Lack of formality makes the use of patterns problematic and the development of appropriate tools difficult. The use of *conceptual modelling languages* such as UML (Fowler, 1997) overcomes these shortcomings.

We propose a refinement of the conceptual modelling approach by using the EKD concepts to describe the body of a pattern (Loucopoulos *et al*, 1997). In ELEKTRA, the generic patterns are grounded on the ontology of the ESI sector and to this end the generic patterns used within ELEKTRA may not be applicable to other domains. However, the structuring framework for the generic patterns is domain independent, the only dependency being in its use of EKD descriptions (i.e. enterprise goals, enterprise processes, etc.).

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 such as : an identification number, a date of issue, a due date, the amount due, etc.. With reference to the EKD methodology, these properties can be described at a generic level in a generic object class. Following the same line, a state transitions graph described at a generic level is associated to the definition of the ‘bill’ object class.

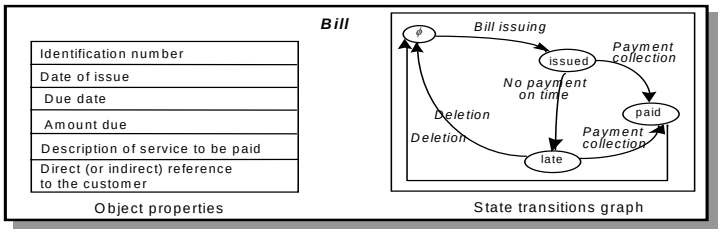

Figure 2 A pattern having an EKD object sub-model fragment as body

3.2 The descriptor

As it will be emphasised in sections 5 and 6, 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 3). 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.. It has a formal part and an informal part as shown in figure 3. 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*. In fact, a pattern should describe not only the 'how', i.e. a solution on how to solve a problem, but also on the 'why', i.e. the reasoning behind the solution-problem pair.

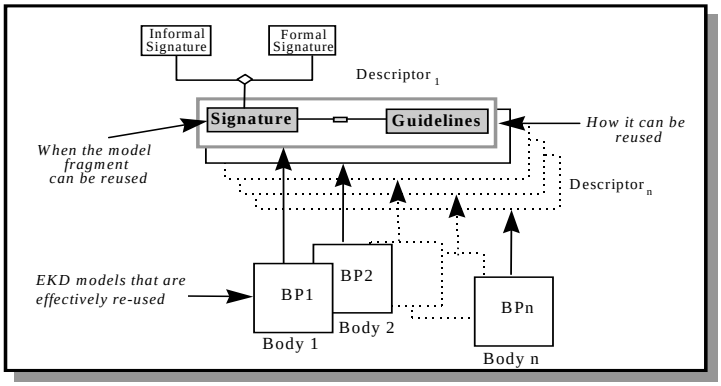

Figure 3 Pattern bodies and descriptors

The formal signature

For describing the *formal signature*, we have chosen to combine both a *faceted approach* with a *contextual approach* (Grosz *et al*.,1997). Accordingly, the formal signature has a *situation* part and a *usage intention* part (see figure 4).

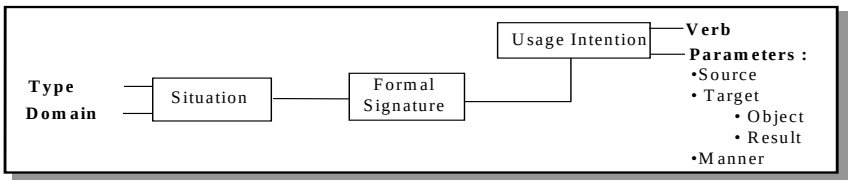

Figure 4 The formal signature description

- The *situation* precisely describes the applicability conditions which must hold for reusing the generic pattern. It comprises two facets :
  - The *type*: the type of the pattern (`Actor/Role, Role/Activity, Object, Rule, Goal, Change process patterns`) (see section 4).

- The *domain*: it describes 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*. Parameters are 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. In the first example below, the target 'electricity consumption' refers to the value of what has been consumed and does not exist prior to the measurement, it is the result of the measurement. In the second example, the 'current situation' is the object of the associated change process whereas the 'CBS structure' is the result of the intention.

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

      Change $_{\text{verb}}$ *(current situation)* $_{\text{object}}$ *(to CBS structure)* $_{\text{result}}$

    - The *source* identifies the origin of what is affected by the usage intention. The source 'from meter reading', in the first example, identifies the origin of the electricity consumption measurement. Similarly, the 'monopoly structure' in the second example, refers to the existing structure model which will be directly affected by the change process.

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

      Change$_{\text{verb}}$ (*current situation*)$_{\text{object}}$*(from monopoly structure)*$_{\text{source}}$
      (*to CBS structure*) $_{\text{result}}$

    - The *manner* parameter is used to express in which way the usage intention is achieved. In the first example, the manner refers to the ways electricity can be measured, this, in fact, could be done in various ways, either automatically or by asking meter readers to collect the appropriate information. Concerning the second example, the intention is to change the structure model from monopoly to CBS in a way which allows to avoid lay-off.

      Measure$_{\text{verb}}$ (*electricity consumption*)$_{\text{result}}$
      (*from meter reading*)$_{\text{source}}$ (*by using meter readers*)$_{\text{manner}}$

      Change $_{\text{verb}}$ (*current situation*)$_{\text{object}}$ (*from monopoly structure*)$_{\text{source}}$
      (*to CBS structure*)$_{\text{result}}$ (*by keeping staff*)$_{\text{manner}}$

Examples of complete formal signatures are given below (figure 5).

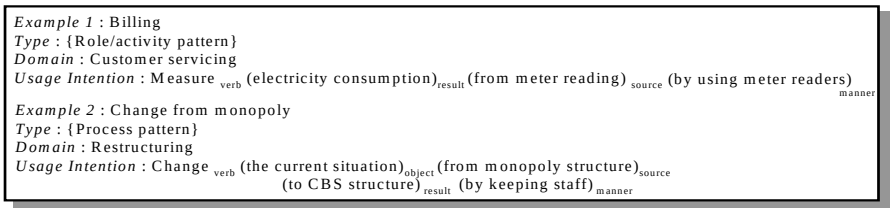

Figure 5 Examples of formal signatures

According to the structure of the formal signature defined above, a thesaurus of all possible values for the introduced facets will be created.

The informal signature

The informal signature of a pattern is composed of mandatory components and optional components. In fact, a pattern should be easily shareable and to this end it should not be cluttered with unnecessary information. This gives rise to mandatory and optional features. 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. The optional components are: the resulting context after using the pattern, possible examples and the rationale used. Figure 6 presents the informal signature related to the pattern about the measurement of electricity consumption.

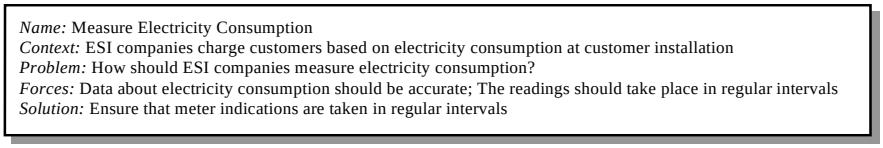

Figure 6 An example of an informal signature

3.3 A complete example of pattern

Figure 7 shows an example of a pattern defined with respect to the pattern template described previously. 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. Note that this pattern is not put forward as the complete solution to this problem. Its purpose at this stage is to be illustrative of the concept of pattern. The pattern contains all the mandatory components specified in the pattern descriptor. The body of the pattern describes an actor-role model. This model is designed in such a way that, according to our belief, it is applicable to all ESI organisations dealing with the problem of measuring electricity consumption.

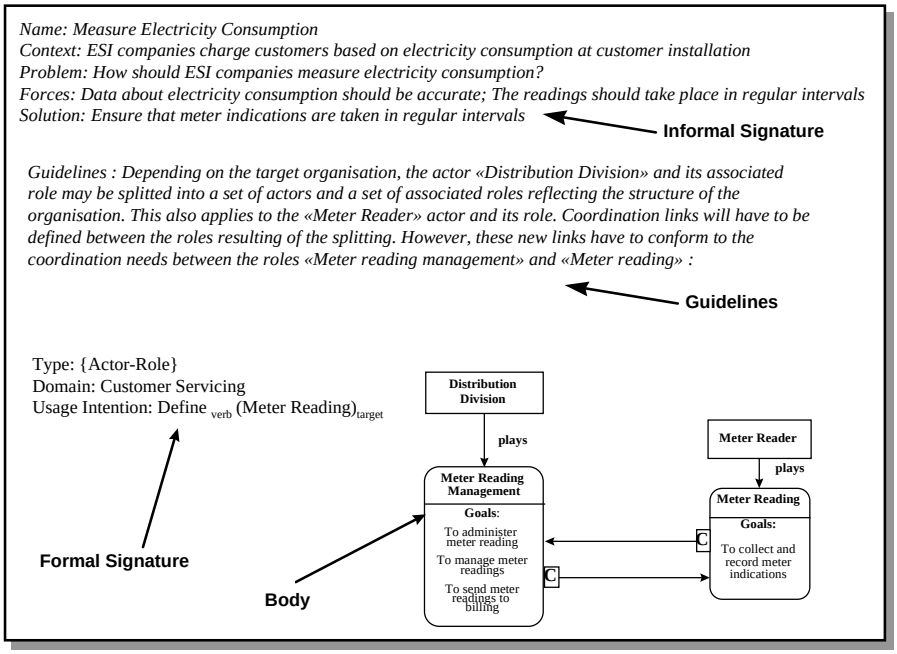

Figure 7 An example of a complete pattern

4 PATTERN TYPOLOGY

4.1 Product patterns

EKD methodology providing the notations which are used in the ELEKTRA project, the simplest position for describing generic patterns consists of representing their bodies as EKD models fragments. On the other hand, since EKD has its own typology of models, one possibility is to extend this typology to the patterns. The EKD sub-models that are greyed in figure 8 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.

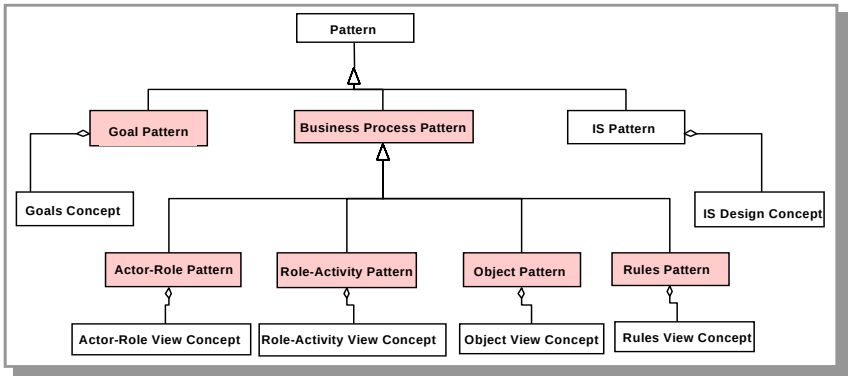

Figure 8 Generic patterns according to EKD models

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*. As depicted in figure 9, both types of patterns are refinements of the 'pattern' concept.

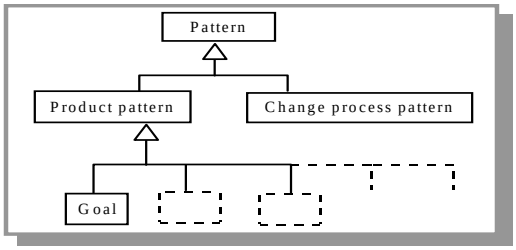

Figure 9 Distinguishing between product and change process patterns

4.2 Change process patterns

Similarly to the product patterns presented in the previous section, 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 10). 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 as EKD models) and where it wants to go (the target situation, e.g. the ISO structural model also represented as EKD models). Note that the remaining of a change process pattern is expressed as a change intention decomposition. The leaves of a goal hierarchy are called operationalisable change intentions. An operationalisable change intention corresponds to an intention that does 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).

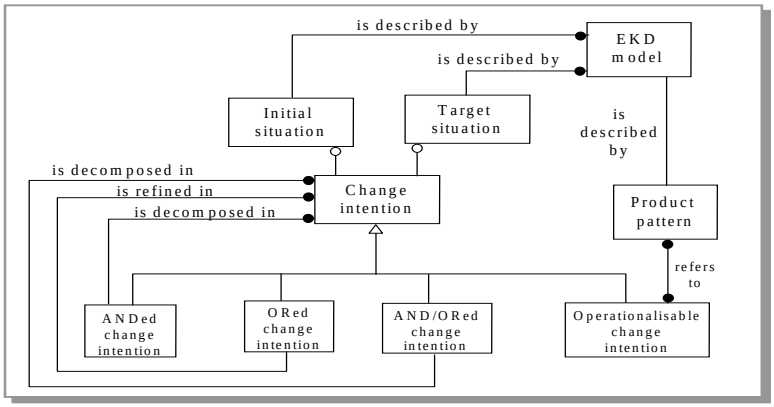

Figure 10 The structure of the body of a change process pattern

4.3 Summary

In summary, we will 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. In the context of the ELEKTRA project, generic patterns are ESI specific, i.e. they encapsulate knowledge about the ESI sector. For instance, we have product patterns related to the different ways the customer's bills can be managed as well as change process patterns for handling change from monopoly to ISO structure. Indeed, application specific EKD models are also instances of the EKD meta-model as depicted in figure 11. These instances can be the expansion or the refinement of the generic patterns as we will develop in section 5.

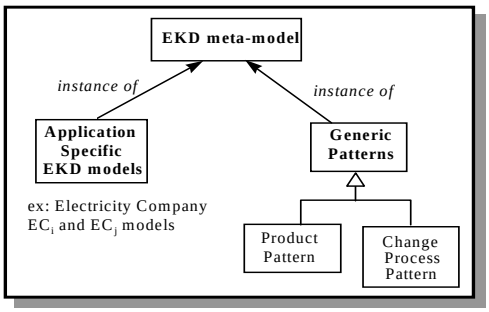

Figure 11 Generic patterns and enterprise specific EKD models

5 PATTERN REPOSITORY

Clearly, composed patterns are meaningful. The first possibility is to define a pattern as an aggregation of other patterns, in other words to compose complex patterns using more elementary patterns. The drawback of this solution is that it hinders a flexible reuse process. A more flexible solution consists of (i) keeping atomic patterns at the knowledge level in the repository, and (ii) expressing their possible composition through an indexing hierarchy (meta-knowledge) level. This is the option that we propose.

5.1 Relationships among patterns and their descriptors

Besides, as the descriptor is merely intentional (i.e. goal oriented), 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). The two first are traditionally used in goal modelling; they are part of the EKD types of goal relationships. The AND/OR relationship is required in order to make possible the expression of a multiple choice between several options.

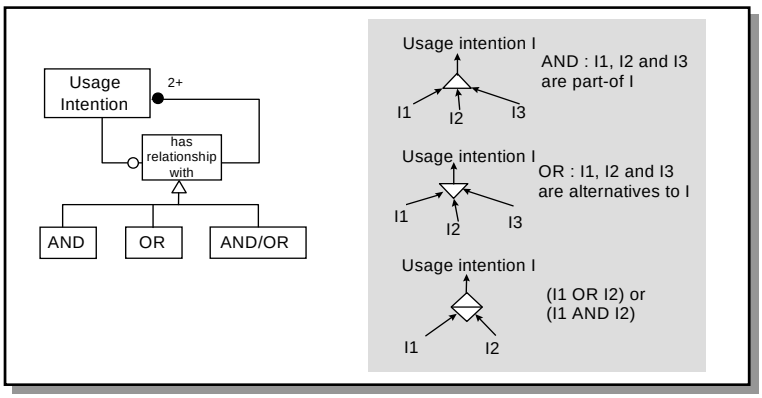

Figure 12 Relationships among usage intentions

This way of expressing the possible combinations of patterns leads to a hierarchical representation of the domain knowledge which is intentional as it is expressed with domain goals. 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 § 3.2) that are meaningful and familiar to them. Figure 12 shows the relationships among usage intentions.

5.2 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 patterns, 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 descriptor of the pattern is an expression of the problem whereas the body is the reusable solution provided to this problem.

The indexing hierarchy part describes this knowledge in terms of the ESI domain goals (usage intentions) that can be fulfilled by reusing the patterns. Indeed, the hierarchical organisation of the usage intentions of descriptors helps structuring the problem in intentional terms that are (or 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.

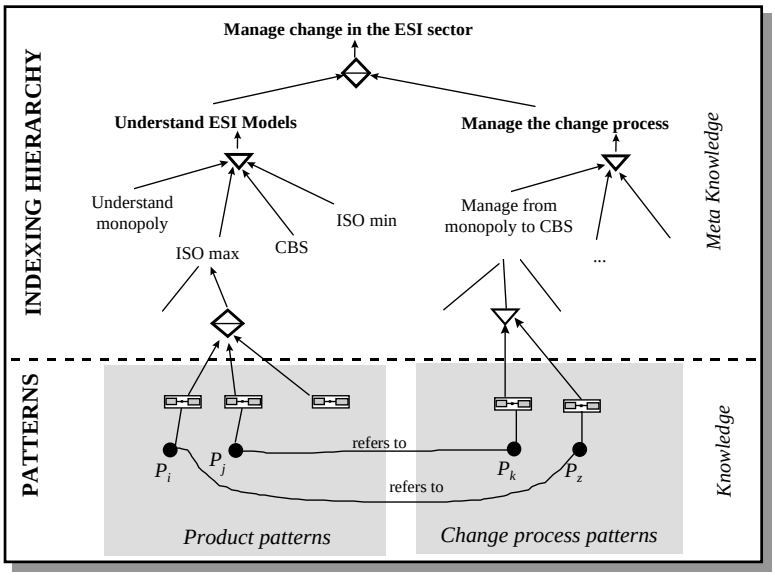

Figure 13 The pattern repository structure

In figure 13, the top level usage intentions figures out what is the overall objective of the ELEKTRA project, 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 10).

6 AN ELECTRONIC HANDBOOK SUPPORTING THE REUSE PROCESS

In this section, we first describe the general process of reuse of the generic patterns based on the framework presented in the previous sections that can be summarised as a process of retrieval, selection and customisation of generic patterns. Then, we detail the architecture of the ‘electronic handbook’, it is a Web based tool that is dedicated to the two first steps of the reuse process, namely retrieval and selection of generic patterns. The third step is planned to be handle by the ‘electronic handbook’. We plan to extend the tool architecture with editing facilities allowing to directly manipulate EKD models and to customise patterns.

6.1 The process of reusing patterns

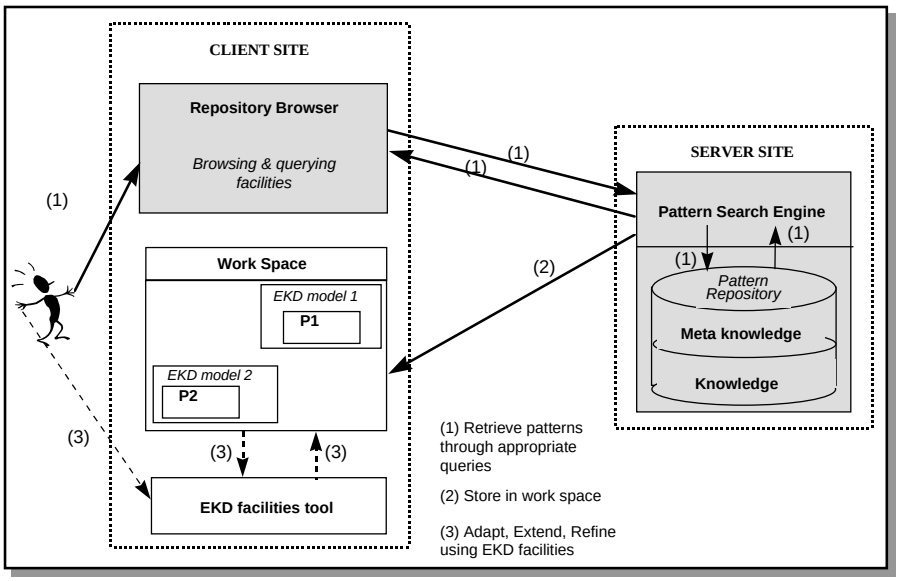

Figure 14 The illustration of the reuse process

Let us now describe the process of reusing a generic pattern. As illustrated in figure 14, it is a three steps process which consists of :

1. the retrieval and selection of the generic patterns from the repository 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 descriptor and use the thesaurus of the terms corresponding to the values of the descriptor 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.

During this step, the domain expert is facing the list of patterns that match his/her requirements. He/she has to select the appropriate ones. This selection process requires to carefully browse the informal signatures of the retrieved patterns in order to understand the differences between them and to determine the one that conforms his/her needs. This step ends up with the selection of one (or several) relevant pattern(s).

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. It shall be noticed that the selected patterns are not instances of the patterns but called *implemented-specialisations* of them. 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. This step starts with the recognition, in the current EKD specification under development (if any), of the parts of the selected pattern which are already described. This leads to define a set of correspondence links between the stored pattern and the current EKD specification. Based on this, the domain expert considers all the elements of the selected pattern which do not have a correspondence link and adds them, and customises them if necessary, in the current EKD specification. It may also be necessary to expand the selected pattern with additional information specific to the company. This can be triggered by a careful study of the guidelines defined in the signature of the pattern.

As shown by the grey elements of figure 14, the ‘electronic handbook’ that is described in the next section handle the two first steps of this process and proposes browsing and querying facilities tailored to the pattern repository.

6.2 The architecture of the ‘electronic handbook’

The electronic handbook is a Web based tool that supports a top-down approach for retrieving the appropriate patterns for a given situation in a given setting. It is composed of two inter-related modules: the *Repository Browser* which runs at the client site and *Pattern Search Engine* which runs at the server site (figure 14).

The *Repository Browser* is the interface provided to the domain expert, it consists in a set of HTML forms which allows him/her to interact with the electronic handbook and to access the pattern repository through either Internet or locally.

The *Patterns Search Engine* is composed of a Web server, a set of CGI applications, and the pattern repository. The Web server is responsible for all exchanges between the client(s) and CGI applications that run on the server. It receives request from client(s) in the form of descriptors elements and sends the resulting patterns that are selected from the pattern repository.

A more detailed presentation of the electronic handbook can be found in (Grosz *et al*, 1998).

6.3 Using the ‘electronic handbook’

We shall now detail the different windows that the domain expert goes through while using the ‘electronic handbook’ For the sake of readability, the presentation follows an imaginary scenario of use.

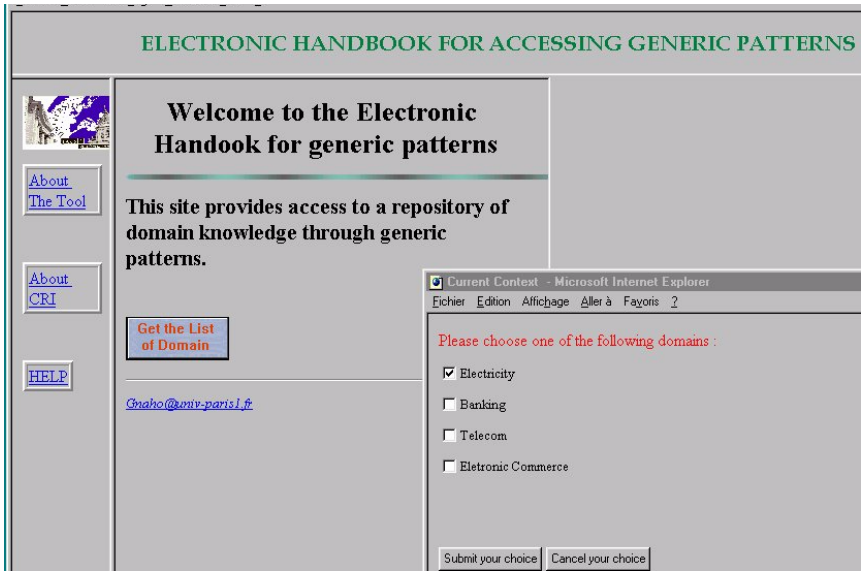

Figure 15 The home page of the ‘electronic handbook’

The background window of figure 15 presents the home page of the 'electronic handbook'. Beside some general information, the main suggestion is to get the list of domain for which generic patterns have been defined within the repository.

After the domain expert had clicked on the button, a query is sent to the Patterns Search Engine to retrieve the list of domains for which patterns have been defined. The result is sent to the Repository Browser which displays the foreground window of figure 15. The domain expert has now to select the one he/she has interest in.

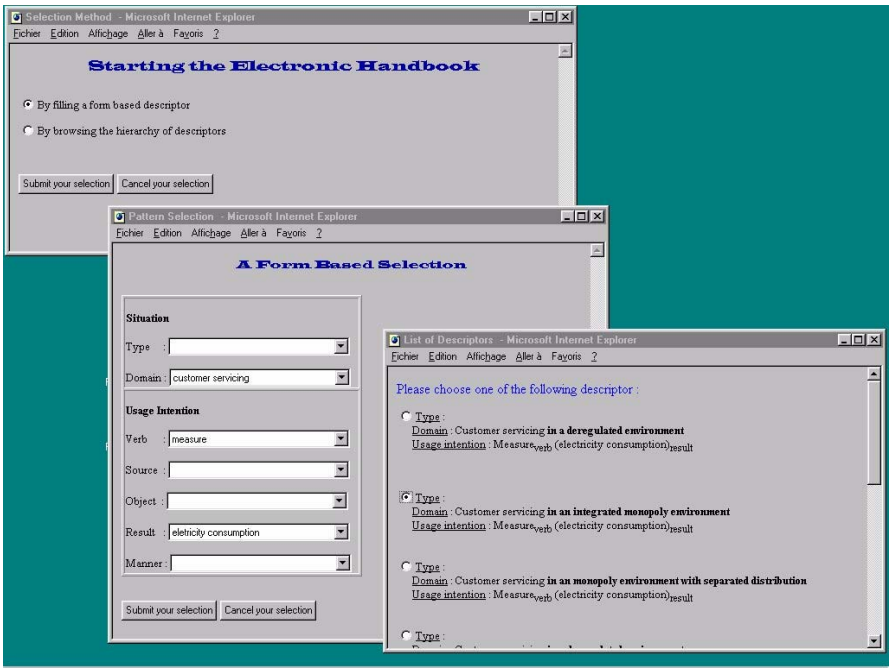

Figure 16 Submitted query and associated matching descriptors

Then, as shown in the background window of figure 16, two options for querying or browsing the pattern repository are proposed. Assume the domain expert selects the query facility, he/she is now asked to fill a form based on the descriptor structure. For each field, a list of possible values (those that are in the thesaurus) is accessible. Selecting an item from this list guarantees the conformity of the values towards those stored in the repository. Indeed, this description may not be complete with regard to the structure of the formal signature. The domain expert fills the properties he/she knows the value of. For instance, he/she may fill the property 'domain' = '*customer servicing*' and describe partially its usage intention, '$Measure_{verb}$ (*electricity consumption*)$_{result}$.

After the form has been submitted (click on the 'submit your choice' button), the Pattern Search Engine searches within the repository for the descriptors that match the properties that were given. The search through the indexing hierarchy is

performed in a top-down manner. In order to minimise the number of retrieved descriptors for the sake of guiding the domain expert, the search engine only retrieves the most general descriptors that match the query. The foreground window of figure 16 displays the resulting list of descriptors. All elements that complete the initial elements of the query are displayed in bold.

The domain expert is asked to select one of these retrieved descriptors. In our case, assume he/she selects the descriptor having $Measure_{verb}$ $(electricity\ consumption)_{result}$ as usage intention and '*customer servicing in an integrated monopoly environment'* as domain.

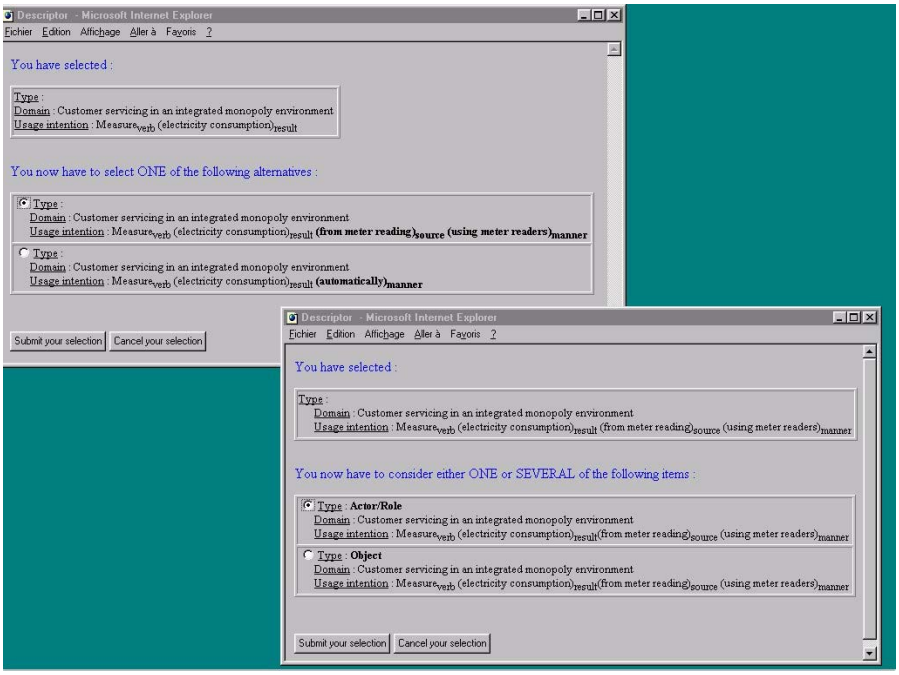

Figure 17 Descending the hierarchy of descriptors

Based on the previous selection, the Pattern Search Engine continues to descend the indexing hierarchy to retrieve more specialised descriptors that refine the selected one as shown in the background window of figure 17. In our example, two are matching : $Measure_{verb}$ $(electricity\ consumption)_{result}$ $(from\ meter\ reading)_{source}$ $(using\ meter\ readers)_{manner}$ and $Measure_{verb}$ $(electricity\ consumption)_{result}$ $(automatically)_{manner}$. Because these two are alternative, the domain expert is asked to select one of them. Note that if the specialised elements would have been connected through the AND or AND/OR connectors, the domain expert could have selected several ones. Assume, he/she selects the first one. The next steps consist in descending the hierarchy until product or change process patterns are

found. Let us just remind that patterns are associated to the leaves of the indexing hierarchy. In our example, the next step, shown in the foreground window of figure 17, leads the Pattern Search Engine to reach leaves of the hierarchy.

Assume the domain expert selects the first item of the list, the associated product pattern is then displayed as shown in figure 18 with all its components (i.e. formal signature, informal signature, guidelines and body). Note that this pattern is the one presented in figure 7 of section 3.3. The domain expert has the ability to display the example(s) associated to this pattern in a different window. If this pattern satisfies his/her expectations, he/she can download the body of the pattern into his current work space. At this point, he/she can choose to stop the retrieval because he/she found what he/she was looking for. Otherwise, he/she has the possibility of continuing the retrieval guided by the Pattern Search Engine thanks to the 'what to do next' button. Clicking on this button will lead the tool to propose the pending options that were listed for this query. In our example, this would first lead the tool to suggest the study of the product pattern of the type 'object' (the second item in the foreground window of figure 17).

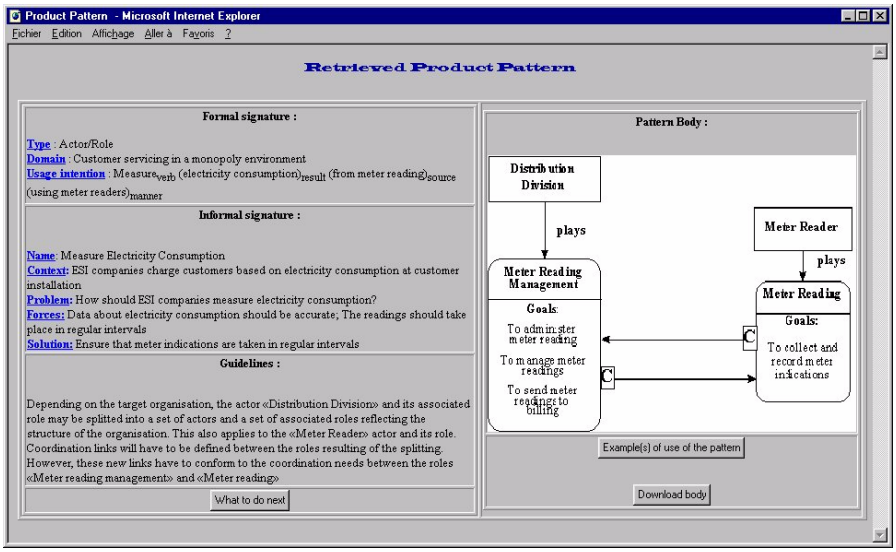

Figure 18 The retrieved product pattern

7 CONCLUSIONS

A critical factor in being able to share best business practice (including best practice for system development) is the appropriate reuse of existing 'chunks' of best practice. Patterns provide the mechanism for achieving this. The generic patterns will have to be built by empirical observation and tested on a range of examples within 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 has attempted to provide first a framework for defining generic patterns for the Electricity Supply Industry (ESI) sector and second the associated electronic handbook.

The framework sets the structure of a pattern template and the structure of the pattern repository. A pattern template comprises two main parts: a descriptor that characterises the problem to be solved and a body that provides a solution to this problem. We distinguish between product and change process patterns. The former are dedicated to the modelling of the ESI sector whereas the latter are tailored to change management in the ESI sector. The repository is composed of two parts : the pattern part and the indexing hierarchy that aims at facilitating the reuse of patterns.

We have developed a mock-up that describes how to navigate through the repository over the Web. More precisely, the tool will offer browsing and querying facilities for the retrieval of the appropriate patterns for a given situation in a given setting using a top-down approach. A first prototype, comprising the browser and the search engine is now under development.

Both the patterns framework and the tool are independent of any application and could potentially be used in other domains. However, the use of the proposed approach on large scale applications may require adaptations in both the repository structure and the underlying method. Furthermore, business problems are sometimes wicked and ill structured. Consequently, the design of generic patterns tailored to such problems is perceived as a difficult task that will be described in some future research papers.

8 REFERENCES

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

Alexander, C. (1979) The Timeless Way of Building, Oxford University Press, NY.

Beck, K. (1997) Smalltalk Best Practice Patterns. Volume 1: Coding, Prentice Hall, Englewood Cliffs, NJ.

Bubenko, J., Strina, J. (1997) EKD User Guide, research report, ELEKTRA project.

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

Coad, P. (1992) Object-Oriented Patterns, in *Communications of the ACM*, Vol. 35, No. 9, 152-159.

Coad, P. et al. (1996) Object Models - Strategies Patterns and Applications, Yourdon Press Computing Series.

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

Fowler, M. (1997) Analysis Patterns: Reusable Object Models, Addison-Wesley.

Frakes, W.B. and Pole, T.P. (1994) An empirical study of representation methods for reusable software components, in *IEEE Transactions on Software Engineering*, Vol. 20, N° 8.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1994*)* Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, MA.

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

Grosz, G., Gnaho, C., Barrios, J. and Yue, W. (1998) A Web based tool for helping information systems engineers in method use, research report, ELEKTRA project, 1998..

Harmsen, F. et al (1994) Situational method engineering for informational system project approaches, *in Method and Associated Tools for the Information Systems Life Cycle*, Verrijn-Stuart and Olle (eds.), North Holland, 169-194, Maastrischt, The Netherlands.

Hay, D. (1996) Data Model Patterns: Conventions of Thought, Dorset House, NY.

Yajima, M.(1997) Deregulatory reforms of the Electricity Supply Industry, Quorum Books.

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

Rolland, C., Prakash, N.A. (1996) Proposal For Context-Specific Method Engineering », in *IFIP WG8.1 on Method Engineering*, Atlanta, August 1996.

Rolland C., Plihon, V. (1996) Using Generic Method Chunks to Generate Process Models Fragments, in *the 2nd International Conference on Requirements Engineering (ICRE'96)*, Colorado-Spring, USA, April 1996.

Rolland, C., Nurcan, S. and Grosz, G. (1997a) Guiding the participative design process, in *Association for Information Systems, Americas Conference on Information Systems*, Indianapolis, Indiana, 15-17 Aug. 1997, USA, 922-924.

Rolland, C., Nurcan, S. and Grosz, G. (1997b) A way of working for change processes, in *International research Symposium: Effective Organisations*, September 4-5, 1997, Dorset, UK.

Rolland, C., Nurcan, S. and Grosz, G. (1998a) A unified framework for modelling co-operative design processes and co-operative business processes, in *the 31st Annual Hawaii International Conference on System Sciences*, Big Island, Hawaii, USA, 6-9 January 1998.

Rolland, C., Grosz, G., Nurcan, S. (1998b) Generic Patterns for the ESI Sector, Intermediary Research Report, (ELEKTRA Project), January 1998.

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

9 BIOGRAPHY

Colette Rolland received her PdD in Mathematics in 1972 from the University of Nancy, France. She is currently a full Professor at the University of Paris 1 Sorbonne. She leads the research team of the 'Centre de Recherche en Informatique' working on Information Systems Development. Hers current research concentrates on requirements engineering, design for and by reuse, process modelling and CASE tools. Professor Rolland is currently the chairman of the IFIP WG8.1.

Georges Grosz received a PhD in Computer Science from the University of Paris 6 in 1991. He is currently an associate professor at the 'Centre de Recherche en Informatique' of the University Paris1- Sorbonne. His research interests lies in the area of information system modelling, process, reuse at the conceptual level, method modelling and CASE tools. He is now mainly involved in method

engineering and organisation change process modelling following a meta-approach.

Selmin Nurcan obtained her engineer diploma in Computer Sciences at INSA-Lyon, (National Institute of Applied Sciences) in 1986. She joined then the Information Systems Engineering Research Centre where she received her PhD in 1990. She is currently an associate professor at the 'Centre de Recherche en Informatique' of the University of Paris 1-Sorbonne. Her research activities include process modelling, computer supported co-operative work, workflow, business process reengineering, requirement engineering and information systems modelling.

Yue Wu received the B.S. and M.S. degrees from Harbin Institute of Technology in 1982 and 1984, respectively, both in computer science and Engineering. Since August 1984, he has been on the faculty of the Department of Computer Science and Engineering at University of Electronic Science and Technology of China, Chengdu, P.R. China. From 1991 to 1993, he was awarded the National Research Fellowship and was a visiting scholar at University of Geneva. From 1995 to 1997, he worked as an expert of 863 National High Technology CIMS in Networking DataBase Group. He is currently a Vice-Dean of the Computer Science and Engineering College at University of Electronic Science and Technology of China and a visiting associate professor at University of Paris 1, France. He has co-authored several journal papers and conference publications. His current areas of research interests include distributed object computing, Java and mobile agent technologies.

Christophe Gnaho graduated in Computer Science and Telecommunication Network Science, in 1992, from "Ecole supérieure d'Ingénieurs en Informatique et Génie des Télécommunications" , France, and received a DEA (Diplôme d'Etudes Approfondies) in electronic, from Université d' Orsay (Paris IX), in 1993. He is currently a PhD student at CRI/Université Paris 1. His research topics focus on defining formalisms and tools for guiding information system development and information system methods modelling activities.