Guiding the Participative Design Process

Colette Rolland, Selmin Nurcan, Georges Grosz
Université Paris 1 - Panthéon - Sorbonne
Centre de Recherche en Informatique
17, rue de Tolbiac, 75013 Paris FRANCE
email : {rolland, nurcan, grosz}@univ-paris1.fr

I. Introduction

It is traditional to look to any engineering activity from both the product point of view and the process point of view. The product is the desired result, the process is the route followed to reach the result. Methods were classically focused on the product aspect of systems development and have paid less attention to the description of formally defined ways-of-working. Clearly, there is an important demand on methods and tools where *process guidance* is offered to provide advice on which activities are appropriate to which situations and how to perform them [Rolland95], [Rolland96], [Downson94], [Wynekoop93]. We propose a way-of-working which intents to provide such a guidance.

Our approach is composed of three complementary elements :
(1) a set of models used for describing the system to be constructed and the organisation in which it will operate,
(2) a way-of-working supporting the usage of concepts,
(3) a set of tools supporting the way-of-working.

This approach is currently being applied, under the name of EKD (Enterprise Knowledge Development) process, in the context of the ESPRIT project ELEKTRA (This work is partially supported by the ESPRIT project ELEKTRA (N° 22927) founded by the EEC in the context of the Framework 4 programme) [ELEKTRA96] for re-organising electricity companies and designing new solutions.

This paper is dedicated to the presentation of the *way-of-working* along with the tool supporting it. The way-of-working aims at organising, structuring, the design process. It provides advices on what should be considered during this process, why and how it should be analysed following some relevant techniques. It also suggests which problem should be tackle next and provides some argument to help in the making of the most appropriate design decision. Finally it includes means to support participative design processes including brainstorming, exchange and emergence of ideas. Thank to the tool support, some process automation is possible and tracing facilities emphasise the recording of the rationale and argumentation provided through out the all process.

This paper is organised as follows: Paragraphe 2 presents the EKD way-of-working for participative design. Paragraphe 3 focusses on the guidance offered by the EKD process.

2. The EKD way of working for participative design

2.1 The EKD process is guided

We look to any participative design process as a *decision making process* i.e. a non deterministic process. The process is performed by responsible agents having the freedom to decide how to proceed according to their evaluation of the situation they are faced to.

However, the participative design process cannot be an ad-hoc and chaotic process. We look to it as a repeatable process made of steps resulting each of the application of the same *pattern for decision making*. The proposed EKD way-of-working is entirely based on this pattern.

The pattern views a decision as the choice of the *way to proceed* in a given *situation* to achieve an *intention*. In the EKD approach, a decision is contextual i.e. both situation and intention driven. An intention can be fulfilled in different ways depending on the situation being considered.

In order to take this aspect into account, we propose to fully associate the intention and the situation in what we call a *context*. Therefore, if we visualise the pattern (Figure 1) as having an input, a body and an output, the *input* is a *couple < situation, intention>* i.e. a context.

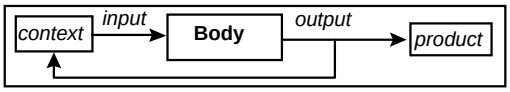

Figure 1 : *the EKD decision making pattern*

Any time an EKD engineer is faced to a given situation to which he/she looks with a certain intention/goal in mind, he/she can apply the decision making pattern. The output of the EKD decision making pattern is either a modification of the product or the generation of new contexts.

Participative design requires a complex process to take place. However, it exists some steps during the performance of this process which are grounded on knowledge [Rolland93]. There is first, *heuristical knowledge* which partly constitutes the know-how of EKD engineers. Secondly, an engineer may try to reuse knowledge independent of any particular domain but *specific to EKD.* Finally, when an engineer has to solve a new design problem, he/she could structure his/her

reasoning by looking for alternative ways to solve the problem or by decomposing the problem into smaller problems. This type of knowledge is fully *generic* and not tailored to EKD.

We identified thus three different types of guiding knowledge : *domain specific knowledge*, *EKD knowledge* and *generic knowledge*. The decision making pattern is taylored to *always provide guidance*.

Therefore, the *body* of the decision making pattern is intended to provide guidance on how to proceed to achieve the intention in the given situation. Our approach provides three types of guidance which can be related to the levels of abstraction introduced in figure 2.

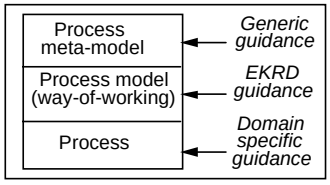

Figure 2 : *Relationship between the different type of guidance and the abstraction levels*

Decision making might require emergence of ideas, exploration of choices, argumentation of various alternatives and perhaps deliberation among the stakeholders involved in the process. The generic decision making pattern takes these aspects into account.

2.2. The EKD process is incremental and dynamic in nature

The suggested way-of-working makes the EKD process *iterative*, each step of the process repeating the EKD decision making pattern. As a consequence, the product which is the target of the process (i.e. the new company organisation of its business processes) is *incrementally constructed*.

In addition, the sequencing of steps is not fixed a priori. Steps are *dynamically* following one the other. The dynamicity is brought by the decision making pattern which allows the EKD engineers to switch from one context to another depending on new happened situations and changes in his/her intentions.

2.3. The EKD process is supported by software tools

The way the EKD environment provides guidance in the performance of the process can be explained using the Dowson's framework [Dowson92] (figure 3).

The framework introduces three interacting domains: *process modelling, process performance and process enactment*. The *process modelling* domain captures all activities performed for modelling software development processes : process model definition, process model specialisation, etc.. The *process enactment* domain encompasses what takes place in a process to support process performance based on the process definitions. The *process performance* domain is defined as the set of activities conducted by human agents and non human agents (e.g. computer). The process enactment domain supports, controls and monitors the activities of the process performance domain. The process performance domain provides feedback information on the current process performance, to enable process adjustment.

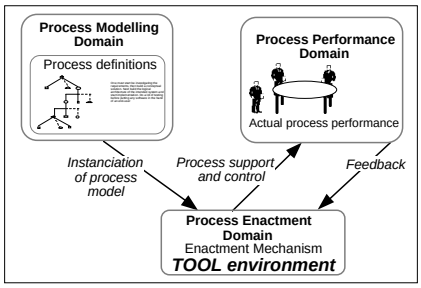

Figure 3 : *The three domains of process performance*

The process model supporting the EKD way-of-working comprises three classes of process model fragments; each of them being adapted to the three types of guidance introduced in paragraph 2.1. We call them *method chunks*, and therefore the participative design environment uses *generic method chunks*, *EKD method chunks* and *domain specific method chunks.* All chunks are stored in the library of the EKD environment and accessible at any point in time of the process performance.

3. Guidance in EKD process

3.1. Generic guidance

The library comprises only one generic guideline that we refer to as the *generic method chunk* or simply *generic chunk.* The chunk is applicable in *situations* where the two other types of guidelines do not hold. The guideline aims to fulfil the goal called "*progress*". It proposes a help strategy for progressing in the EKD process which offers four options: *do*, *plan* and *choose*, each of them corresponding to a given type of context, *executable*, *plan* and *choice context*, respectively.

- The *do* option corresponds to a straight forward resolution strategy. It should be chosen when the engineer knows exactly what needs to be done in order to fulfil the context's intention.
- The *choose* option corresponds to a resolution strategy which requires the exploration of alternative paths. It should be selected when the engineer thinks about different alternative ways for progressing with regard to the input context.
- The *plan* option follows a planning strategy. The engineer has in mind a plan for achieving the context's intention and will progress by building a plan of decisions to be made.

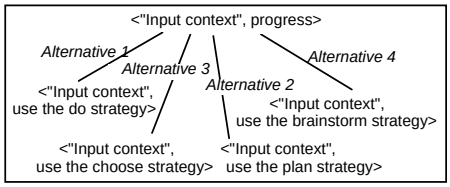

Figure 4 : *The generic method chunk*

Parts of the EKD process are dealing with wickled and ill-defined problems for which even the generic guidance provided by these three options might be found too much inflexible. The discovery of goals is an example. As pinpointed in [Potts89, Dardenne93, Anton96], finding goals is very hard and there is not yet, a way of solving this problem which has proven to be efficient. Organising cooperative work sessions and brainstorming are probably the most adapted approaches to deal with this kind of highly creative activity in order to make ideas emerge.

In order to take into account the cooperative work, we complete the generic chunk by a fourth strategy called *brainstorm* (figure 5). This strategy is supported by the argument "the current situation requires a cooperative brainstorming".

3.2. EKD guidance

EKD guidance is based *on EKD knowledge*. It means knowledge for supporting EKD engineers to specifically undertake the participative design process in an organisation using the EKD models. The EKD knowledge supports for example, the construction of the different models representing the initial enterprise state as well as the future enterprise state of the organisation, the expression of alternative strategies, the evaluation of these strategies, as well as other kinds of activity such as brainstorming, co-operative work, etc..

The EKD knowledge is used within the EKD decision pattern as follows: Assuming that the engineer has chosen the input context, he/she is has to select an EKD method chunk where (1) the situation type matches the input context' situation and (2) the intention of the method chunk matches the input context's intention. This selection is greatly facilitated by the use of a software tool.

We use a *matrix presentation* to overview the collection of chunks included in the EKD knowledge base. The *chunks* are the *matrix elements*. The *columns* of the matrix are *intentions* which arise during the EKD process. The *rows* of the matrix are *techniques* used in the guidelines. The same technique can be used in different ways in different chunks. For example, brainstorming strategy is a technique which might be used for satisfycing the intention of "Detect goal conflict" and for "Solve goal conflict" as well. The SWOT analysis might be used for satisfycing the intention of "Analyse the context of participative design" and for "Argument alternative design models".

By following the heuristical knowledge embedded in the EKD method chunks, EKD engineers are constantly guided. Part of the solution they have to find is provided by the chunk.

3.3. Domain specific guidance

EKD Domain specific guidance is based on *EKD Domain specific knowledge. The Domain specific knowledge* aims at providing guidance to EKD engineers for solving very well focused problems related to a specific domain. It is grounded on experience based knowledge and suggests to reuse and/or to adapt concrete and already tested solutions which have shown their efficiency and feasibility in different organisational settings of the same domain.

The step starts with the retrieval of the Domain specific chunk matching the input context. If there exists such a matching chunk, the EKD engineer can decide to use it. Because domain specific chunks are defined at the instance level, there is no need for context instantiation (as for EKD method chunks). Then the engineer has just to follow the guideline provided.

4. Conclusion

The EKD decision making pattern is a *reasoning mechanism* supporting decision making by providing a *library of guidelines*. In some cases, the pattern offers a domain specific guidance. This happens when the library contains knowledge about the domain of the project which matches the current context of work. The library contains EKD specific guidelines which are tailored to the way the EKD approach suggests to work with the different EKD models. These guidelines are independent of any particular domain but are based on EKD method knowledge. Finally, if none of the two previous types of guidelines matches the current context of work, the generic guideline may operate. Clearly, the more specific the guidance provided is, the more efficient it is. However, the generic guideline, by offering a general frame for decision making, makes the EKD process entirely based on guidance. Currently, we are implementing all these guidelines in an electronic handbook which will eventually be availaible on the World Wide Web.

To sum-up, the EKD process model suggests an incremental production of the design product through a co-operative work. It has two major advantages: it makes change traceable and it helps participants in the participative design process to share awareness by making the product under construction being discussed, visible and explicit.

Bibliography

[Anton96]: Anton, A., "Goal-Based Requirements Analysis", ICRE '96, IEEE, Colorado Springs, Colorado USA, 1996.

[Dardenne93]: Dardenne, A., Lamsweerde, A.v. and Fickas, S., Goal-directed Requirements Acquisition, Science of Computer Programming, Vol. 20, 1993.

[Dowson92]: Dowson, M., "Consistency Maintenance in Process Sensitive Environments", in Proc. of the Process Sensitive SEE Architecture Workshop, Boulder, CO. September 1992.

[Dowson94]: Dowson, M., Fernstrom, C., "Towards requirements for Enactement Mechanisms", Proc. of the 2th European Workshop on Software Process Technology, 1994.

[ELEKTRA96]: "ELectrical Enterprise Knowledge for TRansforming Applications", The ELEKTRA Project Programme, ELEKTRA consortium, 1996.

[Nurcan96]: Nurcan, S., Gnaho, C., Rolland, C., : "Defining Ways-of-Working for Cooperative Work Processes", In procedings of the First International Conference on Practical Aspects of Knowledge Management (PAKM) Workshop on Adaptive Workflow, October 30-31, 1996, Basel, Switzerland.

[Potts89]: Potts, C., "A Generic Model for Representing Design Methods ", Proc. 11th Int. Conf. on "Software Engineering", 1989.

[Rolland93]: Rolland, C., Prakash N., "Reusable Process Chunks", DEXA'93, Int. Conf. on Databases and Expert Systems Applications, Prague, 1993.

[Rolland95]: Rolland, C., Souveyet, C., Moreno, M., : "An Approach for Defining Ways-of-Working", Information Systems Journal, Vol. 20, No 4,1995.

[Rolland96]: Rolland C., "Understanding and Guiding Requirements Engineering Processes", invited talk, IFIP Worl Congress, Camberra, Australie, 1996.

[Wynekoop93]: Wynekoop, J.D., Russo, N.L., "System Development methodologies: unanswered questions and the research-practice gap", Proc. of 14th ICIS (eds. J. I. DeGross, R. P. Bostrom, D. Robey), Orlando, USA, 1993.