Manufacturing Reporting Knowledge Representation: A Case Study at STMicroelectronics

Manel Brichni, Sophie Dupuy-Chessa, Lilia Gzara, Corinne Jeannet

Manufacturing *Reporting* knowledge representation: A case study at *STMicroelectronics*

Manel Brichni[1,2,3], Sophie Dupuy-Chessa[2], Lilia Gzara[1], and Corinne Jeannet[3]

[1] Univ. Grenoble Alpes, G-SCOP, F-38000 Grenoble, France
CNRS, G-SCOP, F-38000 Grenoble, France
`manel.brichni,lilia.gzara@g-scop.grenoble-inp.fr`
[2] Laboratoire d'Informatique de Grenoble `sophie.dupuy@imag.fr`
[3] STMicroelectronics `corinne.jeannet@st.com`

Abstract. This paper addresses the problem of manufacturing *Reporting* knowledge representation in manufacturing companies. An approach to characterize these knowledge is proposed. The solution is applied to the *Reporting* process at *STMicroelectronics* for capitalizing knowledge in the *Wiki* of the company and responding to users' needs. In such an approach, the user participates throughout the knowledge representation definition process, even in choosing knowledge characteristics to represent it. For that aim, three dimensions are taken into consideration: the know *What*, the know *Why* and the know *How*.

1 Introduction

The rapid growth of companies and their business needs, the departure of employees, the complexity of new technologies and the rapid proliferation of information, are reasons why companies seek to capitalize their expert knowledge. In the *Reporting* team at *STMicroelectronics*, the number of created reports is highly growing, while knowledge about their creation is lost. Consequently, this requires to capture and to capitalize knowledge about their creation in order to help not only users to understand the purpose of the report but also engineers to analyze the way it was created, in less time. Among other solutions, *STMicroelectronics* has opted for the use of a *Wiki* to capitalize its expert knowledge, called *Stiki*. In a previous publication [4], an evaluation of *Stiki* use for knowledge sharing was performed. One of the evaluation findings concerns the importance of pages structuring in *Wiki*. This helps readers searching for knowledge, but also contributors capitalizing their work, in an appropriate and effective way. However, this depends on the way knowledge is represented. Our aim, therefore, is to propose a way to represent knowledge while including its different aspects. In this paper, the definition of knowledge characteristics is proposed to integrate those required by users in the capitalization tool and to promote their reuse and exploitation. The proposed solution is applied to the *Reporting* process at *STMicroelectronics* that encounters capitalization issues.

Before presenting the proposed solution for knowledge characteristics representation (section 3), we present, firstly, related work and background (section 2). We end this paper with a conclusion and some perspectives (section 4).

2 Background and related work

This section describes related work in knowledge representation and motivates the need for a new solution. The capitalization process involves knowledge capture, representation, storage, sharing and reuse [1,12,15]. Many approaches have discussed the capitalization issue [9, 13]. For example, the CommonKads approach is based on the construction of a collection of models, where, each one captures specific aspects of knowledge [1, 15]. Even tough our goal is not to develop expert systems, we can retain from it the construction of a collection of models in order to capture different aspects of knowledge (The UML models for example). The AKM (Active Knowledge Modeling) [8] is an other approach supporting the knowledge capitalization. Being able to support collaborative work and participative learning in managing knowledge will decide of its quality. That is why, in addition to using different and complementary models to represent knowledge, our approach should support the user participation through the whole process. Among steps described in the knowledge capitalization process, in this paper, we focus on the representation problem [2,6], as discussed in the following.

Knowledge presents three different and complementary characteristics: *What*, *Why* and *How*. Each one describes knowledge in a different level and should be modeled in an appropriate way. First, the *What* corresponds to the conceptual level of knowledge. It represents manipulated objects and their relationships. Second, the *Why* is described in the behavioral level of knowledge which involves an understanding of principles behind processes. Finally, the *How* addresses the level of the knowledge configuration and integration in systems as well as *How* it could be used by different types of users. We note that these characteristics are generalized so that they could include other ones (when, who, etc.). We note that these three different characteristics are related through sharing common concepts, mainly manipulated *Reporting* objects.

Know-*What*: According to the author's thesis, [5] considers that the know-*What* concept treats procedural aspects of knowledge, while [12] describes the know-*What* as a set of concepts, knowledge and experience. In both cases, it concerns describing what knowledge to be considered throughout the execution of a task. In the literature, the *What* model is represented in different ways: Symbolic models [3] or a UML diagram based on classes and associations [14]. In fact, our aim is to describe effectively and simply the manipulated objects and their relationships during the *Reporting* processes at *STMicroelectronics* while involving different types of users (experts or simple users). That is why during our representation process, complex solutions like [3] are not favorite, but we retain the simplicity and efficiency of proposals like [14], while involving users throughout the capitalization process, even in defining knowledge characteristics.

Know- *Why*: Generally, the reasons for design decisions, can change throughout the designing business process and context, which could make them easily lost. Usually, a system is defined in terms of specifications to describe the way it works, but it does not include a description of *Why* it is designed the way it is, which forms the basis for a causal ambiguity. In the literature, the *Why* model is represented in different ways: Expert systems could be a solution, but mainly in problem resolving, which is not our goal, or design rationale notations [5] such as QOC (Questions, Options, Criteria) or DRL (Decision Representation Language) [7]. As our goal is to provide users with a comprehensible way to represent the know *Why*, design rationale could be adapted to our case study.

Know- *How*: it can represent both knowledge spaces (types, accesses, configurations, etc.) and business processes. Generally, it is neglected compared to the know *What* and *Why*. Our solution is based on a user centred approach requiring, therefore, accessible languages and solutions. For this purpose the IRTV solution (Information-Roles-Tasks-Views) [11] deals, among other objectives, with platform configurations, delivery, extension and improvement. Technical consultants build this model to design and implement systems, to define *how* knowledge is stored, roles access control is enforced, tasks should be executed, views are presented in workplaces, etc. In this paper, we will study how this methodology could cover our know *How* representation. As we have seen in the previous section, studies in the literature do not define (ie with the same names) and represent knowledge aspects similarly (know *What*, *Why* and *How*). In the following, we present our proposal for defining knowledge characteristics that we will apply it later to the *Reporting* activity at *STMicroelectronics*.

3 Our proposal for knowledge representation: a case study on manufacturing *Reporting* process

Our approach consists in exploring current uses, users' expectations and needs, in co-designing with them a new solution and in validating it by applying it on real problems. Because of the lack of space, we do not detail these steps. To create a report, the *Reporting* engineer at *STMicroelectronics* has to define his/her objectives, to access to data, to select his/her data, to define his/her objects (indicators, dimensions, etc.), to create the report, share and capitalize it and to exploit it. Throughout the process, many objects interact via their relationships which are of different natures. The main ones are the "Indicators" or "Dimensions", where, a *dimension* is an analysis parameter that carries the analysis in a query and an *indicator* provides numerical information used to quantify a dimension object. Generally, it is the calculation result on data from databases, for example, the evolution of the turnover of different product lines within an organization. In this case, "the turnover" represents the indicator that is calculated according to the dimensions "years" and "product lines". In the following, we describe each characteristic representation (*What*, *Why* and *How*).
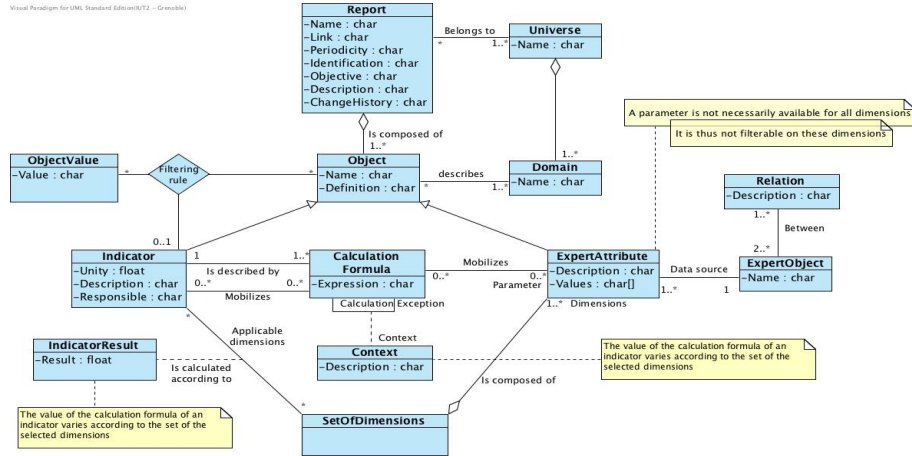
**Fig. 1.** Know *What* representation through *Reporting* process

The know *What* representation By describing the know *What* related to the *Reporting* activity, we aim at representing manipulated objects through the process and their relationships. Such a model will not only facilitate the comprehension of objects interaction but also will guide the way knowledge should be simply and effectively capitalized and shared. In the business intelligence domain, the relationship between *Indicators* and *Dimensions* is represented with a cube, where edges of the cube are made of dimensions and the content of the cube cell corresponds to the value of the indicator according to the combination of the selected dimensions. [10] demonstrated how the cube presentation can be modeled using UML to be more easily perceived by designers and programmers. In our work, we are interested in the meta level of the cube, i.e, not the cube but its concepts. We propose in the following our representation of the know *What* with a UML model applied to the *Reporting* process. Since in the *Reporting* process, engineers and users manipulate expert objects of different nature, we choose therefore to represent our know *What* by a class diagram (figure 1).

As depicted in figure 1, a **report** belongs to a *Universe* and one or more *Domains* in order to classify them according to their content description. It is composed of **objects** that can be **indicators** measuring a production activity (for example, the number of products successfully achieved) or an **expert attribute** representing an **expert object**. For example, a report describing the number of achieved products per technology, is composed of an **indicator** measuring the **"number of products"**, as well as **the expert attribute "technology group"** representing **the expert object "technology"**. In this case, the *"technology group"* is the dimension representing analysis parameter that carries the analysis of the report. In the *Business Intelligence* domain, depending on the objective of the report, the engineer has the possibility to filter objects (data, indicators, dimensions) in order to restrict results and target specific knowledge

(about a specific type of products, for example). That what makes the Business Intelligence interpreting voluminous data friendly, properly and effectively.

One particularity of the reports at *STMicroelectronics*, is that an **indicator** could be calculated by several *Calculation Formulas* according to its *Context* of use. For example, in the Crolles300 manufacture, an indicator could be differently calculated from Crolles200 manufacture, while having the same name and the same objective of use. This could be due to the difference between data used in the manufacturing process between the two sites. Thus, the location could be a *Context* that differentiates the way an indicator is calculated. Besides, an *Indicator Result* varies according to the selected set of dimensions. For example, the number of finished products per technology is not the same as the number of finished products per month. The UML model could present only the way objects are designed together, but not *Why* are they designed in that way? This will be the goal of the *Why* characteristic.

The know *Why* representation Our aim is to represent the behavioral aspect of knowledge in order to effectively capitalize explanations about decisions that were made while performing tasks. Therefore, keeping track of the know-*Why* will provide help to users and designers to resolve problems and to explore more design options. Relying on knowledge described in the conceptual level through objects interaction in the *Reporting* process, we particularly consider, *Why* creating a report? *Why* is an indicator differently created? *Why* choosing a formula instead of another to calculate an indicator? *Why* an indicator could not be calculated or associated to a specific dimension? etc. As in our work, we aim at studying the various directions explored during the *Reporting* process, identified alternatives or why certain options have been made (for the calculation of indicators for example). Therefore, we decided to study the QOC model, present our vision of a possible solution, as well as discuss how we could represent it to deal with the know *What*, previously described. It is composed of *questions* about certain design *options* which represents alternative solutions and *criteria* to explicitly justify the options' selection. The model presented in figure 2 is based on the QOC model (Questions-Options-Criteria). Based on the know *What* model described above (figure 1), the principles behind the use of manipulated objects through the *Reporting* process are detailed.

This example (figure 2) treats the *indicator* object and the principles behind its use. We will study in the future if this model could be applied or adapted to treat other objects of the *What* model. The **indicator WIP** (Work In Progress), which is an object of the know *What* model, calculates the number of lots being processed in the clean room at *STMicroelectronics*. It could be calculated by more than one **formula** according to its **context** of use. Therefore, one of the most important questions asked when selecting an indicator is " *Why* is the **WIP** indicator calculated in that way?". Such an indicator is one of the most used for creating reports. It is important to understand its behavior. In fact, the first part of the QOC model used in figure 2, treats the *Question* part that corresponds to the Indicator **WIP**. *Options* presents its second part. In our
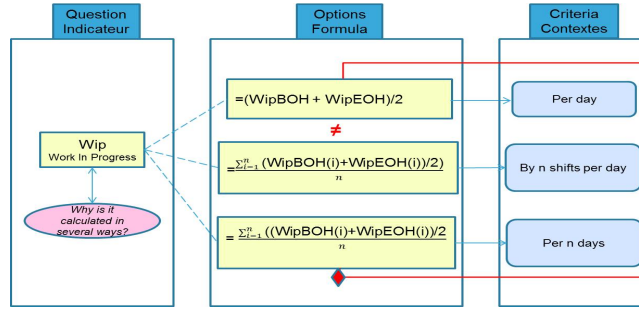
Fig. 2. Know *Why* representation with the QOC model

context of use, options provide different formulas calculating the same indicator. For example the indicator **WIP** could be calculated by either **Formula 1, 2** or **3**. For each one, *Criteria* explain its selection. In our case, contexts of selection of formulas correspond to the *Criteria* in the QOC model. For example, if the user is in the **Context 3** when he/she wants to calculate the **WIP**, then he will select the **Formulas 3** to calculate it. The particularity of this example is that logically, the obtained result of an indicator should be the same whatever the formulas, when considering the same conditions and goal. For example, in both calculation formulas 1 and 2 of the *WIP*, the objective is to calculate the number of achieved lots at the end of the day. However, the results of both formulas are different. Besides, while studying some existing indicators, we realized that the existence of different calculation formulas is due to the selection of the indicators' names. This is the case of **WIP** indicator where even having the same name, the formulas 1 and 2 are actually used to calculate different types of indicator. It could generate a misunderstanding problems between users. To this end, in the future, we will study how the QOC Model could be adapted to deal with such a problem. Using the QOC model facilitates the formulas selection and understanding, depending on their context of use, but understanding *How* each formula was created and integrated in the production system, is considered as a technical knowledge. Such knowledge is not necessarily capitalized and risks to be lost. This will be the goal of the *How* model.

The know *How* representation As we have discussed above, the know *How* treats the knowledge integration and configuration in systems. Capitalizing such knowledge in industrial contexts is a crucial step since the technology evolves and important configuration process could be easily lost. Keeping track of technical details facilitates engineers' work and saves time. The IRTV (Information-Roles-Tasks-Views) methodology deals with platform configuration, delivery, extension and improvement [11]. In figure 3, the IRTV solution is applied to describe the know *How* related to the use of a report (the WIP Status Report) that actually represents an expert object of the know *What* model, as follows:
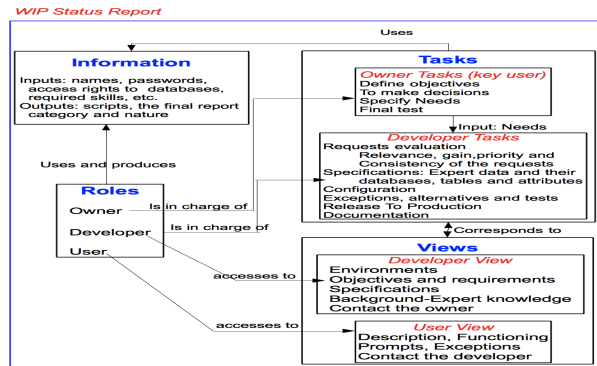
Fig. 3. *How* modeling using the IRTV methodology (the WIP Status Report)

Information: considers knowledge required for the users to create, to manage or to use a report and knowledge produced by them (inputs and outputs). This is not what we described in the *What* model, but, technical knowledge required by an IT member. For example, to create a report, inputs are passwords, access rights, etc. Outputs are scripts or the final report category and nature.

Roles: represents different users who are confronted with the use of a report (Owner, developer and simple user).

Tasks: each role is in charge of specific tasks. For example, as depicted in figure 3, a developer is in charge of evaluating requests of users (the owner role in the Figure, for example) in terms of relevance, priority, gain, etc. He/she needs also to detail data technical specifications (databases, tables, attributes and how to extract them). In addition to documenting these technical knowledge, he/she is in charge of specifying exceptions, alternatives and tests. If needed, he/she has to contact the owner since he is in charge of the request.

Views: Each role accesses to a specific view. As depicted in figure 3, a developer performing technical tasks needs to have a technical view which is different from the view needed by a simple user. He/she needs to have a clearer view about the environment he/she should work on (platforms, tools, etc.), as well as requirements and specification definition, etc.

These knowledge need to be capitalized and shared with other users, particularly between the same category, for an eventual reuse in an effective way.

4 Conclusion

Applied to the *Reporting* process at *STMicroelectronics*, three complementary characteristics know *What*, *Why* and *How* were defined and represented. Our short-term goal is, first, to study how the QOC Model can be adapted to deal with the confusion problem in the know *Why* representation, as well as if it could be applied to other objects of the *What* model or if it has to be completed with

other methods. Second, the representation of the know *How* will be improved in order to involve both process modeling and control mechanism. Finally, the relationship between knowledge representation and its sharing and reuse in *Stiki* will be studied. The idea consists in providing, automatically through proposed models, *Stiki* templates for objects, where each one treats all knowledge characteristics, according to the concerned models. In the long term, we aim at studying the semantic aspect of knowledge representation, as well as knowledge reuse and evolution as other stages of our capitalization process, through which, we will discuss how capitalized knowledge could be reused in an effective way by users. Eventually, we will discuss how new users can search for information while learning existing vocabulary. At *STMicroelectronics*, such an approach for knowledge capitalization could be generalized to other contexts.

References

1. Bera, P.: Knowledge requirements analysis: A case study. PhD thesis, The University of British Columbia (2002)
2. Blaise, J.C., Lhoste, P., Ciccotelli, J.: Formalisation of normative knowledge for safe design. Safety Science (2003) 241–261
3. Bombardier, V., Lhoste, P., Mazaud, C.: Expert knowledge modeling and expert knowledge integration for defect identification by fuzzy linguistic rules. traitement du signal **21** (2004)
4. Brichni, M., Dupuy, S., Gzara, L., Mandran, N., Rozier, D.: A wiki for knowledge sharing : A user-centered evaluation approach. Unpublished
5. Frey, A.G.: Quality of Human-Computer Interaction: Self-Explanatory User Interfaces by Model-Driven Engineering. PhD thesis, Laboratoire d'Informatique de Grenoble (2013)
6. Horkoff, J., Borgida, A., Mylopoulos, J., Barone, D., Jiang, L., Yu, E., Amyot, D.: Making data meaningful : The business intelligence model and its formal semantics in description logics. ODBASE (2012) 700–717
7. Lee, J., Lai, K.Y.: What's in design rationale? Human-Computer Interaction (1991) 251–280
8. Lillehagen, F., Krogstie, J.: Active knowledge modeling of enterprises. ACM Computing classification (2008)
9. Malvache, P., Prieur, P.: Mastering corporate experience with the rex method. ISMICK'93 (1993)
10. Maniatis, A.S.: Olap presentation modeling with uml and xml. Proceedings of the 1st Balkan Conference in Informatics (2003)
11. Marthinusen, I.: The active knowledge modeling approach-to configurable product and process design. Master's thesis, Norwegian university of science and technology (2011)
12. Mathieu, S.: Knowledge management appliqué à la rénovation d'une unité de production de produits biologiques. PhD thesis, Faculté de pharmacie de Grenoble (2012)
13. Matta, N., Ermine, J.L., Aubertin, G., yves Trivin, J.: How to capitalize knowledge with the mask method? IJCAI WORKSHOP (2001) 1–13
14. Page, M., Gensel, J., Capponi, C., Bruley, C., Genoud, P., Ziebelin, D.: Representation de connaissances au moyen de classes et d'associations. LMO (2000)
15. Schreiber, G.: Knowledge enineering and management. MIT Press (2000)