SUPPORTING HUMAN ACTIVITIES The Meta-Level Issue

Grégory Bourguin, Xavier Lepallec

SUPPORTING HUMAN ACTIVITIES
The Meta-Level Issue

Grégory Bourguin, Xavier Le Pallec
Laboratoire TRIGONE, Université des Sciences et Technologies de Lille 59655 Villeneuve d'Ascq, France.
Email: gregory.bourguin@univ-lille1.fr , xavier.le-pallec@univ-lille1.fr

Key words: CSCW, tailorability, interoperability, meta-level, framework, reflection, Activity Theory.

Abstract: Because we have been involved for many years in both the Computer Supported Cooperative Work (CSCW) and the Computer Supported Cooperative Learning (CSCL) research domains, we take particular interests in the results coming both from the human and the computer sciences. Thanks to this cross-disciplinary culture, we have understood that computer systems aim at supporting human activities and that these activities need systems better supporting their emergence. In other words, the systems we traditionally design lack in supporting the inevitable users emerging needs. This paper presents our new approach founded on the human science framework called the Activity Theory and some advanced software design techniques. It shows the results and promises we have found in intensively using the meta-level of the systems we design, thus better taking into account of the expansiveness property of the human activities we want to support.

1. INTRODUCTION

For the last five years, we have been involved in the particular Computer Supported Cooperative Work (CSCW) research domain. CSCW aims at offering computer systems supporting cooperative activities, usually distributed in time or space. Unfortunately, this research area is still trying to understand why the CSCW (or groupware) systems are generally not accepted by their potential users. Studies show that these systems do not seem to fit the users needs. Facing this problem, we have tried to understand its roots by analysing what is the human activity, which concepts we can use for designing our systems, and which mechanisms have to be supported.

For this purpose, we have been studying the Activity Theory (AT) and we have learned that our systems should support well-known users activities as well as the users emerging needs. Unfortunately, the design of a system supporting unpredictable needs is not a simple problem. We strongly believe that a response can be found considering the meta-level of the computer systems we design. We have applied and concretised these ideas in the realisation of the DARE system.

This paper presents why and how we have introduced a meta-level architecture for CSCW inside the DARE environment. The first part summarizes our understanding of the AT and briefly shows the importance of supporting the users emerging needs. The second part presents two different issues we have identified for supporting these needs. The third part describes the DARE system, particularly underlining how we use meta-modelling as a promise for better supporting human activities.

2. THE ACTIVITY THEORY

Activity Theory (AT) is a strong contribution of human science that has a wide audience in the fields of Human Computer Interactions (HCI) and CSCW, due to contributions from (Engeström 1997), (Kuutti 1991) or (Bodker 1991).

From our point of view, one of the main results from AT is the identification of the expansiveness property of human activity. Human activity is expansive in the sense that it transforms its own context while its execution. Particularly, if a computer tool does not fit the users needs, they will try to understand its foundations and adapt it to their needs. If they are not able to do this, the tool will certainly be abandoned, may be for another one. However, studies (Suchman 1987) have shown that the activity is influenced by its context. Then, activity is always influenced by a context it continually transforms in a reflective way. This explains why it seems impossible to exactly

predetermine the users needs towards a computer system because these needs are emerging during the realisation of the activity where the system is used.

We see that expansiveness is closely linked to the reflective property of human activity. Any activity involves a meta-activity. The activity level corresponds to the realisation of a task. The task describes elements creating the context of the activity. The meta-activity level is a reflection about and a transformation of the activity's context, i.e. transformation of the task and its elements. A computer tool helps but also influences the users trying to reach the task's object. If there is a breakdown in the activity (inability to reach the object), the user goes to the meta-activity level, for example questioning the computer tool foundations for understanding what matters. Once the problem has been identified, they try to create a solution by adapting the computer tool, if possible…

Usually, computer systems are designed to support the activity level. We believe that a 'better' computer tool has to support adaptations during its use too, thus supporting the important meta-activity level.

3. TWO DIFFERENT ISSUES

Our main idea inspired by the work of (Kuutti 1991) is to allow the users to co-construct or co-evolve their working environment. We want to propose a system supporting its own redefinition. The CSCW domain is a particularly interesting research area for this purpose because the (re)definition of a system is actually a cooperative activity. Thus, following the above approach, we want to create a CSCW system supporting any cooperative activity, including its own redefinition activity. The question is "how to design and implement such a system?". We have identified two different issues to do so.

3.1 Tailorability

The first answer mainly addresses the tailorability research domain. As it as been defined by Morch, "End-user tailoring is the term […] to describe the activity of adapting generic computer applications to local work practices and user needs" (Morch 1997). However, the creation of a tailorable system is not a simple problem. As underlined by Morch "The price of tailoring flexibility is paid at the expense of having to master an increased amount of computational complexity " (Morch 1997, p 76]). One can notice that end-users are generally not computer scientists. We then believe that an equilibrium has to exist between the users motivation for realising their task, and the effort to be furnished for understanding and adapting the system. Our work is to identify the meanings facilitating the existence of this equilibrium. We need to create a system with understandable foundations from the users point of view. The components framework approach is an interesting technique to do so (e.g. Hummes 1999). However, we believe that this technique alone is not sufficient to reach our goals in supporting the end-users emerging needs.

3.2 Interoperability

As we just mentioned, tailorability is generally achieved following a component approach. However, sometimes users may need to evolve their working environment by integrating a new system in their working context. Today for example, many researchers in the CSCW research domain argue that groupware systems like shared workspaces should better care about the regulation of the users activities supported by the system. At the opposite, the workflow research domain tries to find a way to be less prescriptive and somewhere a little bit more cooperative, thus better supporting adaptations in case of breakdown during the workflow execution. If a user needs a groupware system articulated with a workflow one, he should be able to integrate the two systems in its working environment. This cannot be realised following a component approach because none of the two systems will integrate the other. They have to interoperate for creating a global environment fulfilling the users needs. Thus supporting such evolutions of the working environment involves considering the systems interoperability research domain.

4. DARE

Following the above approach, we have designed a groupware system supporting these human activity properties. The system we have built is called DARE that signifies "Distributed Activities in a Reflective Environment". Its goal is to propose a global platform, creating some distributed user environments facilitating and creating a context for the use of many different computer tools. Then DARE aims at offering specific contexts, designed for particular distributed activities, and used by particular communities of users. A particular context is called an *activity-support*. Trying to support the

expansiveness property of human activity, DARE allows its own users to co-construct and evolve their activity-supports during their execution.

4.1 A Tailorable System

4.1.1 The Framework/OI Approach

One of our main goals is to allow users to adapt themselves their computer system to their emerging needs, and thus addresses the tailorability research domain (cf. 3.1). We have already mentioned that many researchers have recently proposed different tailorable systems founded on a components framework approach. A framework allows reusing a full software architecture dedicated to a generic task, thus facilitating further development or adaptations by domain specialists (Pree 1997).

The definition of a framework contains a generic model that can be specialised for particular purposes. In DARE, our goal is to support any distributed cooperative activity. We then have created the DARE generic model of an activity-support that specifies what is an activity-support in a generic way. This model is represented in Figure 1.

An activity-support contains a set of subjects that are users involved in a corresponding activity. Each subject plays a particular role and uses particular tools including shared tools.

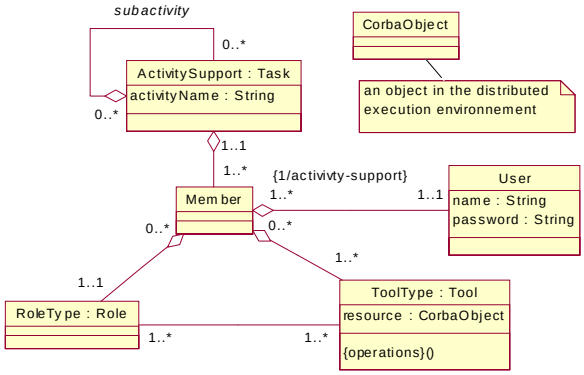

Figure 1. DARE generic model of an activity-support (UML notation).

Thus, adopting a framework helps in creating high abstraction levels for system adaptations. However, this approach is not sufficient for allowing end-users to adapt themselves their computer system during its use. From the end-user viewpoint, the application built over a framework still remains a black box. We have to open the black box, thus allowing end-users to access to the framework level of their application. An answer can be found in the open implementation approach.

The *Open Implementation* (OI) approach described by (Kiczales 1996) discusses the limits of the *black-box abstraction* broadly used in software engineering domain. Kiczales argues that the black box has to be opened to allow users to understand the implementation strategy that relies under the system and/or to allow them to chose the strategy that better fits their needs. For us, the black box is an activity-support and the strategy is the underlying task. OI is characterised by considering well-defined and separated base level and meta-level. This is achieved by providing a base interface and a meta-interface on the black box. The base interface specifies what can be done with the black box in the normal use. The meta-interface specifies what can be done for understanding and/or modifying the implementation of the black box. Our idea is to adapt OI by merging it to the framework approach described before.

Our application is built over a framework defining a generic model dedicated to the application domain. The base interface allows the users to work with the application without knowing or taking care about the underlying framework. However, an associated meta-interface allows accessing to the meta-application level where the users can discover, understand and transform the application structure by rearranging, adding or removing components inside the framework. Moreover, as any component can itself be built following the OI approach, it can propose a base interface allowing its normal use, and a meta-interface allowing its extension.

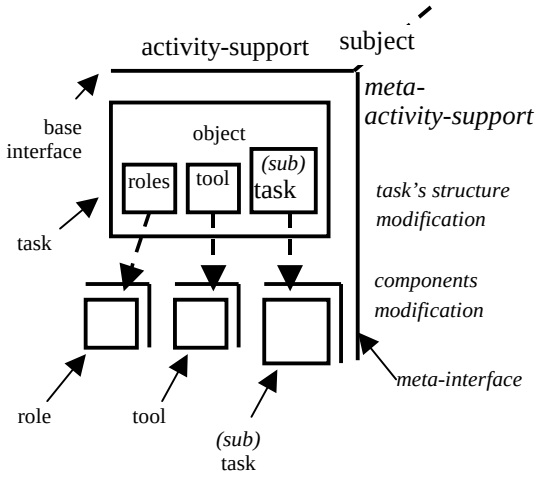

Figure 2. DARE : a tailorable groupware.

We have applied this technique to DARE. Figure 2 shows that a subject uses a base interface to interact in the distributed environment corresponding to an activity-support. The subject uses some tools, plays a role and may have to

perform subtasks. These components appear in the activity-support because the corresponding task specifies their use. At the activity-support level, the subject only uses these components and does not really care about their definition. The meta-activity level is reached thanks to a meta-interface corresponding to the meta-activity-support in which the subjects can (re)define their activity-support.

An existing technique for OI is *computational reflection* (Maes 1987) that helps in providing system interfaces for examination and modification, in other words, providing a meta-interface. Maes defines a reflective system as a "causally-connected meta-system that has as object system itself". In DARE, the task contains a representation of the activity-support, and the meta-activity support allows accessing to the task. Then, we have defined a causal relationship between any activity-support and its task. A modification of a task involves direct repercussion on its instances, i.e. its corresponding activity-supports. We have used the *Meta Object Protocol* (Kiczales 1991) technique to implement this.

Meta Object Protocol (MOP) defines a meta-level description of the system allowing examination and manipulation of the system during its own execution while maintaining a causal connection between them. MOP implements the OI using object-oriented programming techniques. In languages implementing MOP like Smalltalk or Java, the meta-level corresponds to the classes, and the execution to their instances. In MOP, any class is itself considered as an instance of a meta-class. As meta-classes are themselves classes, everything is an object offering methods that can be invoked at the run time. It is then possible to "browse" or transform a class from its instances, thus changing the behaviour of the system from the system's execution. So, thanks to computational reflection and the MOP approach, it is possible to implement DARE as a reflective groupware allowing its own users to co-construct their working environment during their use.

4.1.2 Meta-modelling

The OI and MOP approach has already been used in CSCW by (Dourish 1998) in Prospero, a toolkit for CSCW designers. The main difference between Prospero and our system is that Prospero addresses computer scientists. In DARE, we want to adapt OI to directly address end-users. For this, there are more things to be done. Particularly, we have to define the meta-interface allowing our users to access to the task and its components definition. In traditional OI approach, the meta-interface uses a programming language paradigm. Unfortunately and using MOP, even if our framework is made accessible to the end-users, the generic model still corresponds to a set of abstract classes that have to be specialised for particular purposes. Such a specialisation is usually performed thanks to an object-oriented language. Class, attribute, method, and inheritance concepts have to be mastered to do so. These concepts are generally not understood by end-users. As we want to support expansiveness and users co-construction of the environment, we have to make our generic model understandable and manageable from the user viewpoint. We have to find a meaning for opening the framework in an understandable way, and to provide a meta-interface founded on concepts that are not object-oriented but domain-oriented. This has been realised thanks to meta-modelling.

A meta-model is the model of a model. Computer scientists are used to work with different meta-models (Frankel 1998). For example, UML is a meta-model that defines entities and relations for describing object models. The role of our meta-model is similar to the UML one. The difference is that the concepts it defines are oriented towards our domain of interest.

The elements constituting a meta-model are called meta-types. For example, the UML meta-types are the Class, the Operation, the Attribute or the Association. Meta-types instances are used to create models. In a reflective system, meta-types are implemented and offer a meta-interface. In MOP, the Class meta-type is implemented as a meta-class.

In DARE, we aim at creating a new meta-model defining concepts close to our domain abstraction level for creating our meta-interface. The DARE meta-model defines the basis of a language for understanding and describing activity-supports. The description of a particular activity-support is a particular activity-support model. We have already shown that an activity-support is an instance of a task. In other words, the model of a particular activity-support is a particular task. Then, the DARE meta-model reifies what is a task, i.e. what are its components and its structure. The specification of a task contains a set of role types, tool types and (sub)tasks. These entities and their relations are described in Figure 3. Thanks to this meta-model, the modification of the elements involved in the activity-supports is not performed in terms of class, method or attribute, but in more domain dependent terms like task, tool, role, action, etc., thus allowing and facilitating the creation and management of some particular activity-supports by their own users. The kernel part of DARE has been realised in Smalltalk that implements the MOP.

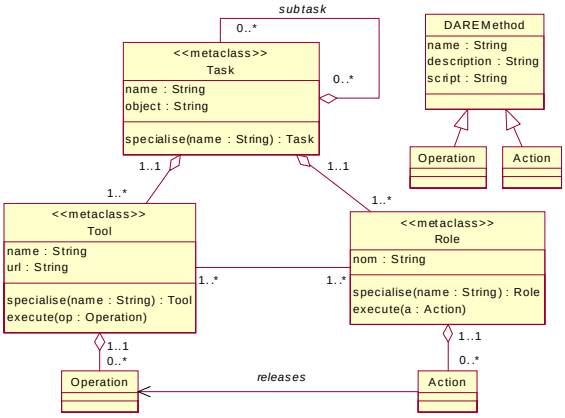

Figure 3. The DARE meta-model (UML notation).

4.2 The Interoperability

4.2.1 Interoperability Context

As argued in part 3.2, making information systems interoperate seems to be necessary for extending the users working environment. The interoperability problem has been studied for more than 20 years. It is a complex problem that has not perfect solution. In fact, various solutions about interoperability exist which focus on one or more aspects of interoperability (autonomy, ease of use…). With regard to these solutions, (Paepcke 1998) counts five approaches : strong standards, family of standards, external mediation, specification-based interaction and mobile functionality.

For our objective, specification-based interaction and mobile functionality solutions are not adapted. Specification-based interaction solutions, like ontology, are powerful, but the description languages used in such solutions, like KIF (Weinstein 1998), are complex. Current solutions of mobile functionality, like Java applets, require a common execution environment. This constraint cannot be respected in context of interoperability between information systems. In addition, such solutions do not really provide a way to create bridges between systems. So we choose external mediation and standards.

External mediation consists in creating translation component between others components. Its main problem is that interoperability between two specific types of components requires a specific mediator. So it is necessary to provide an environment to accelerate the translation components creation. The use of metadata is very helpful in such environment. It allows working at a more abstract level (meta-level), which means to define translation operations for types of data. So meta-data will still be a key element of our proposition. We are convinced that interoperating systems will be constrained to provide metadata about their data. However, such constraint is not a matter due to trend in creating and using new metadata standards, like RDF (Swick 1999) or the Meta-Object Facility (MOF) we use and describe in next part.

4.2.2 The Meta-Object Facility

Our choice of metadata standard between existing ones is based on two reflections :

1. DARE can interoperate with all sorts of systems. It means that specialized metadata standards like Dublin Core or PICS (Manola 1998) do not suit.
2. In our interoperability context, the problem is not only sharing information, like in database federation (Hull 1997), but also sharing functionalities. This implies two statements :

- As functionalities of a system will be used and system metadata are parts of the system framework, metadata have to include these functionalities. So metadata must be objects.
- Information, functionalities and so metadata have to be accessible through a standard protocol.

The Meta-Object Facility (OMG) is the only one to provide such qualities. Its process of metadata creation is based on meta-modelling which enables creating any type of metadata. Its meta-modelling process is object oriented and created metadata are CORBA objects. In addition, as it manipulates meta-models, the MOF works at a more abstract level than common metadata level. So it is more generic and thus, the MOF is the base of our proposition.

As the metadata standard is selected, the next step is to find MOF tool(s) that is adapted to our goal. Requirements of such tools are to provide a saving of time and an easy use in constructing bridges between systems. In (Le Pallec 2001), we show that minimal requirements for such tools are to score high in prototyping and to enable working simultaneous on different meta-models (i.e. different MOF metadata servers). Unfortunately, no existing MOF tool has these characteristics.

To go further in our interoperability problem for supporting emerging needs, we have developed RAM3 that is a MOF tool prototype meeting previous requirements. We aim at using RAM3 to quickly develop two tools : a graphic link editor between MOF metadata, and a facility to easily make links between metadata and data. These two tools will achieve our bridge creation environment.

Our first construction using them will have to unite DARE and a workflow system.

5. CONCLUSION

Drawing from the fact that the systems we traditionally design lack in supporting the inevitable users emerging needs, we have presented our new approach founded on the Activity Theory and some advanced software design techniques. AT teaches us that a computer system should support both the activity it has been designed for and its closely linked meta-activity. We strongly believe that this can be achieved by allowing the users to access and manipulate the meta-level of their computer systems in the context of their use. Our main focus here has been to show that meta-level architectures, meta-modelling or even meta-data are fruitful approaches in future software design. The DARE system applies and implements these principles, thus proposing a reflective groupware supporting its own redefinition activity. Actually, our main results are about tailorability and there is more work to be done for interoperability, even if our advances in meta-modelling, MOF and RAM3 seem promising.

More generally, our work tries to understand software design techniques from the human activity viewpoint and we aim at identifying how they can be developed in order to help better design software architectures supporting end-users activities. Human sciences teach us that the best persons to develop a computer system are those that are using it. Today, systems users are generally domain specific specialists but are not computer scientists. Our assumption is that we have to further understand the essence of human activity and to propose software better supporting its expansiveness and reflectivity properties, because computer scientists are the only ones that are able to use computer technologies to do so.

REFERENCES

Bodker S., 1991, Activity Theory as a challenge to system design, dans "*Information systems Research: Contemporary Approaches and Emergent Traditions*", Nissen H., Klein H., Hirschheim R. (eds), Elesevier Science Publishers, BV (North Holland), pp. 551-564.

Dourish P., 1998, Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications, ACM Transaction on Computer-Human Interaction, vol. 5, n°2, pp 109-155.

Engeström Y., Brown K., Christopher L., Gregory J., 1997, Coordination, cooperation and communication in the courts, dans Cole M., Engeström Y., Vasquez O. (eds), *Mind, Culture and Activity*. Cambridge University Press, Cambridge, UK.

Frankel D., 1998, Technology-independent business object – the concept of a meta-model, Java Report, pp 71 –78.

Hull R., 1997, Managing Semantic Heterogeneity in Databases : A Theoretical Perspective, PODS 97, Tucson Arizona USA, p51-61

Hummes J., Merialdo B., 1999, Design of extensible component-based groupware, soumis pour être publié par Kluwer dans le Journal of CSCW. http://www.eurocom.fr/~hummes/docs/JCSCW/JCSCW.html .

Kiczales G., 1996, Beyond the black box: open implementation, IEEE Software.

Kiczales G., Bobrow D.G., Des Rivieres J., 1991, *The Art of the Metaobject Protocol*, MIT Press, 335 p.

Kuutti K., 1991, *The concept of activity as a basic unit of analysis for CSCW research*, Proceeding of the second ECSCW'91 conference, Kluwers Academics Publishers, pp 249-264.

Le Pallec X., Bourguin G., 2001, RAM3 : un outil dynamique pour le Meta-Object Facility, proceedings of LMO'01, Le Croisic, FRANCE, L'Objet, Hermes, vol. 7 - n°1-2/2001, pp 79-94.

Maes P., 1987, *Computational Reflection*, Ph.D. Thesis, V.U.B, Brussels.

Manola F., 1998, *Towards a Web Object Model*, http://www.objs.com/wom.html

Morch A. , 1997, Method and Tools for Tailoring of Object-oriented Applications: An Evolving Artifacts Approach, part 1, Dr. Scient. Thesis Research Report 241, University of OSLO, Department of Informatics.

Paepcke A., Chang C-C K., Garcia-Molina H & Winograd T., 1998, *Interoperability for Digital Libraries Worldwide*, Communications of the ACM, vol 41 n°4, p33-43.

Pree W., 1997, Component-based software Development – A new paradigm in software engineering ?, Software-Concepts and Tools, Springer-Verlag, n° 18, pp 169-174.

Suchman L. , 1987, *Plans and Situated Actions*, Cambridge University Press, Cambridge, UK.

Swick R., 1999 *Putting it together: RDF: weaving the web of discovery*, netWorker: The Craft of Network Computing, Volume 3 , Issue 2, pp 21-25.

Weinstein P. C. , 1998, *Ontology-based metadata: transforming the MARC legacy*, International Conference on Digital Libraries, June 23 - 26, Pittsburgh, PA USA, pp 254-263.