A Multiple-Perspective Architecture for CSCW Applications

Enio Emanuel Ramos RUSSO [a, b, 1], Alberto RAPOSO [a],
Terrence FERNANDO [c] and Marcelo GATTASS [a]
[a] *Tecgraf, Department of Computer Science, PUC-Rio, Brazil*
[b] *PETROBRAS Research and Development Centre (CENPES), Brazil*
[c] *School of Construction and Property Management, University of Salford, UK*

Abstract. We present a mutiple-perspective collaboration metamodel, which mixes Place-Centred and People-Centred perspectives. It allows instances of the metamodel to be derived and experimented until the more adequate to a particular situation is found. It also allows parametric changes in run-time, enhancing the flexibility of the metamodel. The motivation for this work was extracted from the necessity of developing for a global oil & gas company a collaborative virtual workspace for disaster management of oil & gas offshore structures.

Keywords. collaboration modeling, metamodel, collaboration architecture, multiple perspectives, oil & gas

Introduction

Many companies have been creating virtual teams that bring together geographically dispersed workers with complementary skills, increasing the demand for CSCW applications. In order to make the development of a wide range of these collaborative applications more effective, we should offer a general architecture that is adaptable to different situations, tasks, and settings in a flexible way.

CSCW research to date on how to address the architecture characteristic mentioned above has largely focused on issues concerning differences between: *(i)* co-located work and working across distance; or *(ii)* work with people from the same culture or common ground and work with people from different cultures. The previous perspectives have been named, respectively: *Place-Centred* and *People-Centred* [1].

We propose to adopt a different view on the problem based on the activities carried out by the teams participating in the collaborative work. We name it an *Activity-Centred* perspective, which may be seen as a multi-perspective concept since it not only encompasses the Place-Centred and the People-Centred perspectives, but also allows adopting each one or both of them (in a hybrid way) to the desired extent, and admits seamless change from one perspective to another.

The motivation for this work has been the necessity of developing a collaborative virtual workspace for disaster management of oil & gas offshore structures for a global company [2].

[1] Corresponding Author: Centro de Pesquisas e Desenvolvimento da PETROBRAS (CENPES), Cidade Universitária Quadra 7, Fundão, Rio de Janeiro, RJ, 21949-900, Brasil; E-mail: enio@tecgraf.puc-rio.br.

1. Activity-Centred Metamodel

Dewan's generic collaborative architecture [3] structures a groupware application into a variable number of layers from the domain-dependent level to the hardware level, where a layer is a software component corresponding to a specific level of abstraction. Similarly, the Clover architectural metamodel [4] also structures a groupware application into a variable number of layers, decomposing each layer into three functional sub-components dedicated to production, communication and coordination.

Our proposed metamodel adopts a similar multi-level approach, accordingly to Leontjev's [5, 6] activity theory version in which a three-level scheme describes the hierarchical structure of activity. Orthogonally to this approach, similarly to the Clover metamodel, the Activity-Centred metamodel also allows the breakdown of the components correspondent to a specific level. These two orthogonal approaches applied together contribute to the generality of the metamodel.

1.1. Metamodel Abstraction Levels

The top-most level is represented by a complex *node* which encompasses the whole activity. This level can be as diverse as the elaboration of this paper or the disaster management of an oil & gas offshore structure. The level immediately below contains the main actions that should be performed in order to accomplish the activity. These actions are the result of the interactions of groups, with each group represented by a complex node and the interactions among them represented by *edges*.

Splitting downwards each complex node of the upper abstraction level in more elementary nodes, we reach a *leaf node*, which will typically be a *person* or a *software agent*. To those leaf nodes we then associate implementation and hardware attributes such as the application to be executed and the host in which it should be run.

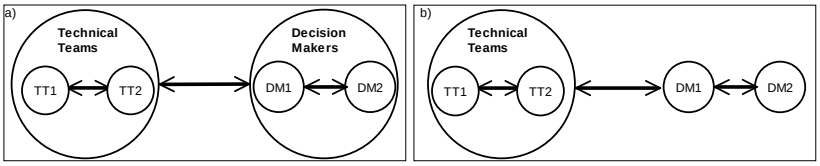

Figure 1. a) The first downward level of the oil & gas company from the disaster management collaborative application. b) Now Decision Maker 1 is placed between the Technical Teams node and Decision Maker 2.

Orthogonally to the top-down process, the Activity-Centred metamodel also allows the breakdown of the components correspondent to a specific level. Let's consider one level of the disaster management example, namely the first one downward of the oil & gas company (Figure 1a). We can observe two main groups: Technical Teams (TT) and Decision Makers (DM). TT is decomposed into sub-groups, and DM, also decomposed into sub-groups. In Figure 1a, both DMs have the same background and level of interaction with TT, while in Figure 1b DM2 has a higher organisational level, with DM1 making the link between TT and him.

1.2. Metamodel Components

1.2.1. Nodes and Edges

Nodes are essential components of our metamodel, going from the top-most node representing the whole activity through many nodes of different levels representing

groups and sub-groups until the *leaf nodes* representing a *person* or a *software agent*. Nodes have a set of *attributes* such as user interface preferences and language used, which are applied using a hierarchical class concept.

Nodes also have an attribute called *artifacts* defined as "all objects on which users can operate" [7]. Examples of artifacts are drawings, physical models, prototypes, and documents. Following the class concept, an artifact associated with a group node is shared by all members in the group, unless otherwise explicitly stated. In this case, a mechanism such as an access control list will determine who share access to the artifact.

Edges in our metamodel represent the *interaction paths* among nodes, which can be uni or bi-directed. When an edge is represented by a thin arrow, this means that the nodes on its extremities are co-located. When the arrow is thick, the nodes are placed remotely to each other. Edges have one important element, *channel*, which represents the electronically mediated channel that allows communication between two nodes.

1.2.2. Edge Especialisation Elements

We have identified the need for additional *edge especialisation elements*, namely *pre-* and *post-communication processings*, which are separated into two different classes. The first class is constituted by the ones directly associated with the leaf nodes. They represent the processings to be executed particularly onto a specific message being passed between two nodes and are stored in a especial table with key (message_id, receiver). The second class is constituted by the ones associated with groups on different levels of the metamodel hierarchy, representing the policies of these groups when respectively sending (*out-policies*) and receiving (*in-policies*) messages.

In Figure 2, we show possible pre- and post-communication processings that could be executed while sending a message from a Computer Science Researcher CR1 of the Computer Science Dept. CD1 of University U1 to Researcher CR2 of University U2.

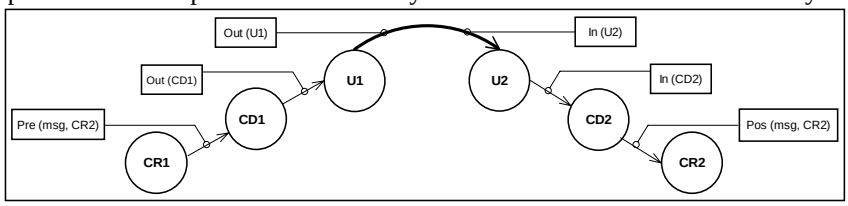

Figure 2. Activity-Centred metamodel: pre- and post-communication processings.

At the sender side, the natural candidate to execute the pre-processings is the leaf node who is sending the message. At the receiver side, this could be accomplished adding an attribute to the first group node pointed by the edge (in our example, U2) corresponding to the leaf node of this group to execute the post-processings.

Regarding the algorithms to be executed when sending a message, it is important to note that each message has one initial sender, which is necessarily a leaf node, and one or more final receivers, which can be either leaf or group nodes. The algorithms to be executed at either side are shown in Table 1.

1.2.3. Role Rules and Message Attributes Table

Role rules for coordination structure have been employed in CSCW for more than one decade [8, 9]. According to the majority of these studies, we adopted the strategy of separating the coordination structure and the computational program, using a logic-based specification language for specifying coordination policies.

Table 1. Activity-Centred metamodel: sender and receiver algorithms.

sender (sender, receiver, flag)

- until the receiver is found repeat
  - at the current level, search for the sub-tree that contains the receiver
  - if the receiver is found (and all the path from the sender to the receiver is determined)
    - if flag = in_table
      - execute the pre-processing associated with the pair (message, receiver)
    - else
      - create a new line in the message attributes table with pair (message, receiver) indicating the post-processing to be executed
    - execute all the out-policies associated with groups on levels in the path beginning at the sender until the communication edge is reached
    - send the message with the receiver to the leaf node which is assigned to the post-processing attribute of the receiver group node, or to the receiver itself
  - else
    - go to the upper level

Sender side of the communication edge (executed by the initial_sender leaf node):

- for each final_receiver associated with the message
  - sender (initial_sender, final_receiver, in_table)

Receiver side of the communication edge:

- receive the message
- execute all the in-policies associated with groups on levels in the path beginning at the present node until the final_receiver node is reached
- if the final_receiver is a leaf node
  - if it is equal to the post-processings execution node:
    - execute the post-processing associated with the pair (message, final_receiver)
  - else
    - send the message to the final_receiver
- else
  - execute the post-processing associated with the pair (message, final_receiver), which in this case should determine the leaf node(s) or group node(s) to receive the message
  - for each of the node(s) determined above (current_node)
    - sender (final_receiver, current_node, not_in_table)

We declare a *collaboration bus*, used to connect all participants, having at least one *channel* declaration. Different collaborations may be executed at the same time, each with its correspondent collaboration bus. The set of participants who are governed by the same set of coordination policies is playing the same *role*. When these policies also define the order in which the events occur, they can be considered workflow rules. Communication among participants occurs through one or more message channels associated with one collaboration bus. Similarly to COCA [9], the basic tasks of receiving messages and sending out messages are: *(i)* for receiving messages, an active rule named *on-arrive* with arguments *channel*, *receiver*, *message_id* (and *sender*); *(ii)* for sending out messages, a *send* formula with arguments *channel*, *sender*, *message_id* (and *receiver*).

We also build a *message attributes table* to enhance the flexibility of the coordination program, separating coordination rules from data related specifically to each message. This table provides an indirection that enables dynamic reconfiguration.

2. Instanciating the Activity-Centred Metamodel: Activity-Centred Models

Sometimes the most important aspects of our collaborative application are related to the place where people are effectively working. A model using this *Place-Centred*

perspective for a paper elaboration collaborative application is shown in Figure 3a. There, we have three main nodes: PUC University, Salford University and Petrobras (BR, Brazilian oil & gas company). The central node, playing the main role in writing the paper, is PUC, which communicates remotely with both Salford University and Petrobras. Within PUC, we have two sub-groups: one is the Computer Science (CP, C for Computer and P for PUC) Department, which has two co-located researchers, and the other is the Engineering department, which has one single Engineer (EP1). The two departments, being in different buildings, also communicate remotely. In Salford, there is only one Computer Science researcher, and in Petrobras, two co-located Engineers.

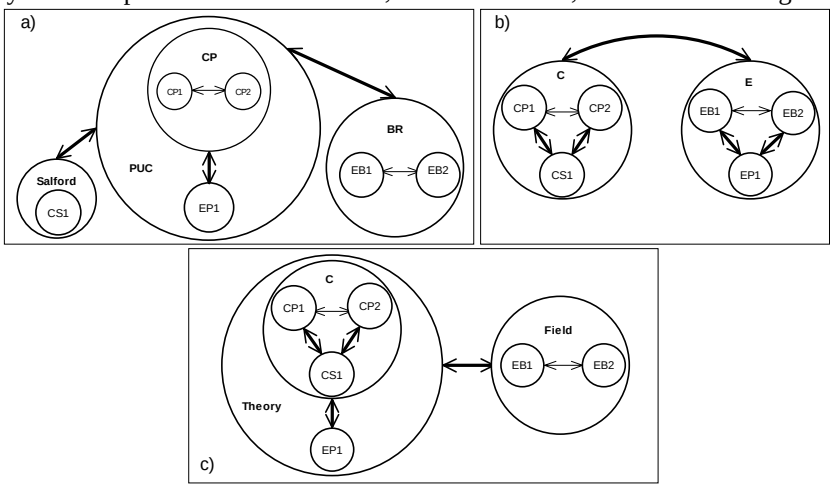

Figure 3. The paper elaboration collaborative application: a) a Place-Centred perspective; b) a People-Centred perspective; c) an Activity-Centred perspective.

Now consider that the main concerning issues of our collaborative application are related to culture and common ground barriers. In this case, we should derive a model with a *People-Centred* perspective (Figure 3b). We now have only two main nodes: the Computer Science researchers' (C) group and the Engineers' (E) group, communicating remotely. Within C group, we have three researchers: CP1 and CP2 work co-located and CS1 works remotely. It is important to note that, although CS1 is from a different university than CP1 and CP2, their common ground is so intense that they belong to the same sub-group. The same reasoning is applied to the E group.

We now mix the two previous perspectives in what we call an *Activity-Centred* perspective. In the present collaborative application, it seems more adequate to focus on the whole activity being performed – the paper elaboration – and then derive the groups to be formed. To elaborate the paper, authors CP1, CP2, CS1 and EP1 try to derive a new theoretical model based on the requirements' identified through field study, working together with Engineers EB1 and EB2. So we aggregate those people in two main groups formed based on their main activity: the Theory group and the Field group, which communicate remotely (Figure 3c).

3. Case Study

We now focus on the case study that motivated the creation of our metamodel: the development of a collaborative virtual workspace for disaster management of oil & gas

offshore structures for the Petrobras Research and Development Centre.

The disaster management of an oil & gas offshore structure is a complex operation involving three main groups: the oil & gas company, the Rescue Team and the Health Care Centre. This is an inter-organisational complex activity led by the oil & gas company, whose node will be detailed. An overall picture of the disaster management collaborative application is depicted in Figure 4.

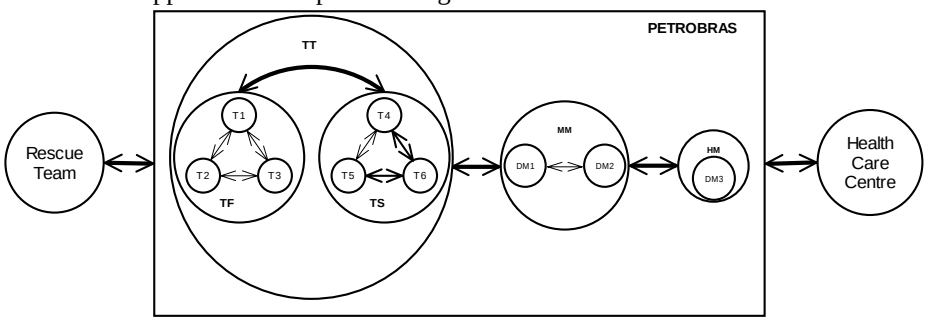

Figure 4. The disaster management collaborative application: overall picture.

Within Petrobras node, we identify three main groups: the Technical Teams (TT), the Middle-level Managers (MM) and the High-level Managers (HM), each one remotely located to the other. TT is formed by two technical sub-groups: the Task Force (TF) team and the Technical Support (TS) team, also remotely located.

TF plays the main role, leading the make-decision process. It is constituted by three co-located technicians, such as naval engineers, structural engineers, risers analysts or oceanographers. TF runs different simulators to derive the best solution to save the offshore unit, permanently communicating with TS. They also maintain contact with MM informing about their work evolution and asking for approval for their derived solution. Once their solution is approved, they pass the sequence of commands to be executed to the unit operator (not represented in our picture).

TS team, with technicians working in the same fields as TF team, can be invoked by TF team to perform especialised simulations focusing on some particular issues that would not be possible to be done by TF, or to obtain another opinion about the problem.

MM is constituted by middle-level managers working co-located in a company office, with one of them usually being the responsible to make the final decision. They have an overall knowledge about the technical issues and work constantly interacting with the TT group. They also communicate with the HM group, informing about the work evolution and eventually when they need to make a more critical decision.

3.1. Prototype

After investigating the activities involved in this disaster scenario, identifying their requirements in terms of ICT, we decided to concentrate on the Technical Teams group to develop a prototype of collaborative application implementing a particular model of our Activity-Centred metamodel. This prototype is particularly related to the work performed by the Task Force group (TF), including the simulators they run, their mutual communication and their interaction with the Middle-level Manager group.

We first investigate how TF runs the different simulators and what are the relationships among them. During a crisis situation, Petrobras typically uses three simulators. The first simulator to be run is SSTAB [10], the Floating Units Stability

system. The second simulator is called WAMIT and uses as input the results from SSTAB. The third simulator is DYNASIM [11], for Dynamic Stability. It uses as input the results obtained from WAMIT as well as additional parameters related to environmental conditions. DYNASIM calculates the forces acting on the mooring lines and risers. When these forces are considered extreme, a retrofeedback process is started, performing all the simulations again, beginning with SSTAB, to find another stable condition of the unit.

An Activity-Centred model representing this crisis situation (Figure 5a) can be derived based on the participants' roles. We created two remote groups: Technical Teams (TT) and Decision Makers (DM). TT is constituted by the Task Force (TF) team with members T0, T1 and T3, and the agent S2. DM is constituted by a single manager, a representative of all participants not directly involved with the technical part of the simulation activity such as operators and other managers, who only receive from TT follow-up messages, commands to be executed or approval requests.

Other than the interaction network part of the model just described, we also define rules and the message attributes table in order to represent the following workflow.

The Crisis Pilot T0 plays the main role in this disaster application, coordinating the collaborative session and leading the make-decision process. He asks for the SSTAB operator (T1) to begin his simulation. After receiving a message from agent S2 indicating the end of its simulation, he asks for DYNASIM operator (T3) to begin his simulation. On receiving a simulation conclusion message from T3, he makes a decision based on the force values acting on mooring lines and risers. If he understands that these forces are extreme, he asks for T1 to begin all the process again, in order to find a new stable condition of the unit, and this loop continues until he is satisfied with the force values obtained. In this case, he makes contact with DM1, asking for his approval to their solution.

The basic conceptual level architecture of our collaborative application is shown in Figure 5b.

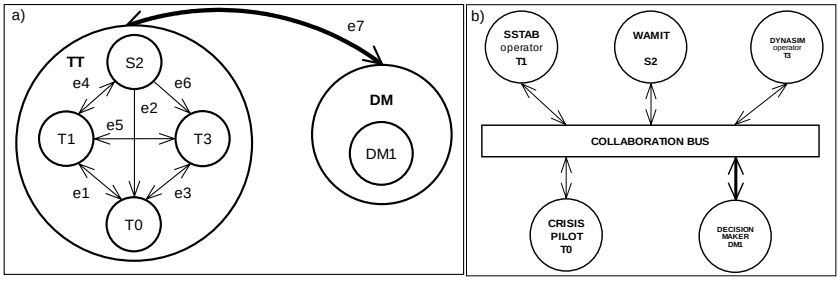

Figure 5. A first model of the disaster management collaborative application (a) and its prototype (b).

In order to map our model into an implementation-level architecture, we investigated different approaches, having in mind two main requirements: real-time support and open-source standard to develop prototypes. We chose HLA – High Level Architecture [12, 13], which not only fulfils our requirements but also is a flexible component-based architecture, in accordance to the principles we have been pursuing.

4. Conclusions and Future Work

We propose a multiple-perspective metamodel, which mixes Place-Centred and People-

Centred perspectives. It employs not a technology-driven but a human- and socially-centred approach. Associating pre- and post-communication processings to each of these levels, we could accommodate policy and privacy rules of organisations, even allowing inter-organisational work.

The metamodel allows flexibility in many dimensions. Separating high-level abstraction features from low-level implementation features allows the designer and the application developer to concentrate on their particular domain of expertise. Separating the computational program and the coordination program allows programmers to concentrate on coordination issues with high-level abstraction.

The metamodel is also customisable in the sense that it allows associating pre- and post-communication processings with each message sent. It allows parametric run-time changes such as changing names of pre- and post-communication processings in the message attributes table, or even changing the pre- and post-communication codes before they have been loaded during a collaborative session.

There is still a lot of work to do in order to make our metamodel a fully flexible and evolving collaborative architecture. For example, we should investigate how to promote our metamodel from a customisable category to an adaptable category [14], upgrading from the capability of adjusting parametric controls to the capability of reconfiguring its behaviour according to immediate patterns of use. We could accomplish this using a learning mechanism to monitor the users' activities.

References

[1] Q. Jones, S.A. Grandhi, L. Terveen, and S. Whittaker, People-to-People-to-Geographical-Places: The P3 Framework for Location-Based Community Systems, *Computer Supported Cooperative Work* **13** (2004), 249-282, Kluwer Academic Publishers.

[2] E.E.R. Russo, A.B. Raposo, T. Fernando, and M. Gattass, Workspace Challenges for the Oil & Gas Exploration & Production Industry, in *Proceedings of CONVR 2004 - 4th Conference of Construction Applications of Virtual Reality* (2004), 145-150.

[3] P. Dewan, Architectures for Collaborative Applications, in Beaudouin-Lafon (eds.), *Computer Supported Cooperative Work*, 1999, 169-194, John Wiley & Sons Ltd.

[4] Y. Laurillau and L. Nigay, Clover Architecture for Groupware, in *Proc. of CSCW´02* (2002), 236-245.

[5] A.N. Leontjev, *Activity, Consciousness and Personality*, Prentice-Hall, Englewood Cliffs, USA, 1978.

[6] T. Tuikka, Remote Concept Design from An Activity Theory Perspective, in *Proceedings of CSCW´02* (2002), 186-195.

[7] T. Gross and W. Prinz, Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation, *Computer Supported Cooperative Work* **13** (2004), 283-303, Kluwer Academic Publishers.

[8] M. Cortés and P. Mishra, DCWPL: A Programming Language For Describing Collaborative Work, in *Proceedings of CSCW´96* (1996), 21-29.

[9] D. Li and R. Muntz, COCA: Collaboration Objects Coordination Architecture, in *Proceedings of CSCW´98* (1998), 179-188.

[10] L.C.G. Coelho, C.G. Jordani, M.C. Oliveira, and I.Q. Masetti, Equilibrium, Ballast Control and Free-Surface Effect Computations Using The Sstab System. *8th Int. Conf. Stability of Ships and Ocean Vehicles - Stab* (2003), 377-388.

[11] L.C.G. Coelho, K. Nishimoto, and I.Q. Masetti, Dynamic Simulation of Anchoring Systems Using Computer Graphics. *OMAE Conference* (2001).

[12] IEEE 1516, The Institute of Electrical and Electronic Engineers, IEEE Std 1516-2000, *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-Framework and Rules* (2000).

[13] A. Kapolka, The Extensible Run-Time Infrastructure (XRTI): An Experimental Implementation of Proposed Improvements to the High Level Architecture, *Master's thesis*, Naval Postgraduate School, Monterey, CA, USA, 2003.

[14] P. Dourish, Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications, *ACM Transactions on Computer-Human Interaction* **5, 2** (1998), 109-155.