The Reconciler: supporting actors in meaning negotiation

Marcello Sarini[1] and Carla Simone[2]

[1]*Dept. of Computer Science, University of Torino, Corso Svizzera 185,*

I-10148 Torino, Italy. e-mail: sarini@di.unito.it

[2]*Department of Informatics, System and Communication (DISCO),*

University of Milano-Bicocca, Via Bicocca degli Arcimboldi 8, 20126 Milano, Italy.

e-mail: carla.simone@disco.unimib.it

Motivations

Organizations are increasingly evolving toward a network of autonomous, distributed, specialized local components involved in cooperative activities. This trend triggers, and is triggered by, the evolution of technology that supports communication and cooperation in different work settings. Despite the richness and powerfulness of current technologies (in terms of functionality, connectivity, integration and so on), there is no adequate support to the work of articulating such kinds of cooperative distributed activities. With *articulation work* (Schmidt and Simone, 1996) is intended the additional effort needed in cooperative settings in order to align, coordinate, mesh, etc. the distributed cooperative activities characterizing the specific field of work.

The lack of technological support to articulation work can be related to many different aspects (see for example (Simone and Sarini, 2001) for a discussion about the issues in designing technologies supporting collaborating actors in the construction and use of classification schemes). Here, we focus on the kind of support that should be provided in terms of what is called *semantic interoperability* (Simone et al., 1999). Since part of the activities related to the articulation among different communities is expressed in terms of communication among the involved actors, communication has to be understood in order to guarantee articulation. One kind of misunderstanding in communication can be related to the different terminologies actors use during their collaboration. Since each of them communicates in terms of her professional domain, the construction of a *common space of understanding* is needed to work jointly. Another kind of misunderstanding can arise during inter-community collaboration since actors describe the shared space in which they will collaborate in very different ways. In this case descriptions serve like "maps" drawn using a set of concepts in which different levels of granularity, different names and structures represent the different but not irreconcilable perspectives the communities have on the same shared "territory" (see (Kent, 1978) for the metaphor of maps as artifacts giving different descriptions of a single common territory). Furthermore, since each local view represents consolidated conventions, each community wants to maintain it without "being enforced" to create and use a unique standardized global view. As a consequence of the above facts, the use of local descriptions in communication may generate misunderstandings. In fact, if a term used in a message is unknown or differently used by the receiver, misunderstandings arise since the receiver

interprets the term in her context. Semantic interoperability aims at supporting this interpretation reducing the risk of misunderstandings.

Possible solutions

There are mainly two possible approaches to provide semantic interoperability: one refers to the integration of local views in a unique global representation in order to let the actors belonging to different communities collaborate thanks to their mutual understanding based on the common language defined by the integration. In order to avoid imposing this unique vision and give flexibility to users, systems usually provide the opportunities to define local customization of the global view according to the user needs and preferences when they represent local activities. Along this view, Macadam (Dourish et al, 1999) is a prototype allowing users to construct personalized local views of a global classification scheme organizing information in an engineering environment.

An alternative approach to semantic interoperability is related to mapping. This implies first to establish correspondences among elements belonging to the different local views in order to let the involved actors maintain the different local descriptions and then to solve the eventual (semantic) conflicts related to the defined correspondences. Different approaches are proposed according to the role of technology in supporting the different ways to realize mappings and to solve the related conflicts.

In EDAMOK (Bonifacio et al, 2001), a project about building Distributed Architecture for KM Systems, semantic interoperability is based on two steps: making explicit each community context, and creating mappings from context to context. Mapping is realized automatically thanks to algorithms of context matching. However, the authors claim that providing support to context matching is not enough. For human beings knowledge sharing is often the result of a social process, in which many different cooperative strategies are used. In that case, the system should aim at reproducing this social process, called *meaning negotiation*. This is realized in the architecture through communication protocols between autonomous agents reproducing the dynamic of social interaction at system level.

What we call reconciliation (Simone et al., 1999) is in the line of emphasizing social processes. To re-concile means to bring into agreement; hence, reconciliation is a process where a common territory is provided to let people discuss to make comparable different points of view, still leaving them the possibility to use their local languages for communication. Since each community describes locally the common space of collaboration in very different ways, a tool supporting the reconciliation process is needed. The goal of supporting the reconciliation of different perspectives in cooperation can be achieved in two ways: either incorporating in the support a predefined model of reconciliation or proposing a light support and let users define their own way to handle the reconciliation process. In the first case, models based on argumentation and explanation could be adopted as well as on the structure of the dialogue (e.g. (Conklin and Begemann, 1988), (Fischer et al, 1991), (Karsenty and Brezillon, 1995), (Winograd and Flores, 1986), (Zacklad and Rousseaux, 1995)). According to the second approach, we implemented a light support to experiment its usage and to base a possible set of richer functionalities on the result of the experimentation.

Our approach

In our view, reconciliation is the explicit additional effort of articulation that involves actors with different local cultures and skills and different work conventions in order to achieve the *alignment of meanings* (Sarini and Simone, 2002) for cooperation to occur. The idea that articulation aiming at alignment of meanings has to be explicitly considered was derived from empirical investigations in the framework of the Politeam project (Wulf, 1997). In that experience, the effort to recompose dis-alignments was mainly performed during workshops where users, with the help of technologists, discussed the linguistic misunderstandings specifically induced by the introduction of a document management technology supporting their cooperation. Moreover, it was recognized that the outcomes of that re-alignment were not naturally incorporated in this technology, and users were in charge to manage the established clarifications outside the system. From that experience, we derived the idea that the alignment of meanings can be obtained by the combination of two phases: the first one, *the definition phase*, where users are active in clarifying the problems and negotiate partial solutions; the second one, *the communication phase*, where this effort is paid back since the supportive technology alleviates the communication problems using the collected information. On this basis, we implemented a specialized module, called Reconciler (Simone et al., 1999), which is a technology to interactively build correspondences to solve conflicts related to concepts used in communication during cooperation. The basic idea is to provide users with a framework where they can cooperatively establish correspondences between entities, attributes with their domains, and relations (the language of the Reconciler to describe the local views) and discuss the various kinds of semantic conflicts arisen to (partially) solve them. Taking inspiration from the conceptual modeling approach (Batini at al, 1986) the considered conflicts are: *Naming conflicts* (including Synonym and Homonym), *Category conflicts*, *Structural conflicts* (including Type and Dependency conflicts) and *Unit conflicts*.

To support the two different phases of alignment of meanings the Reconciler module provides two kinds of interfaces: *the Definition and the Communication Interfaces*.

The Definition interface

The *Definition interface* is split in two parts: the one related to the first scheme, let's say A, and the other related to the second scheme, let's say B (Figure 1).

In the current implementation the interface is unique for the two schemes. The scenario of use of the reconciliation process supported by the Reconciler is supposed to be a negotiation involving representatives of the two communities where a facilitator is in charge to record the negotiated correspondences through the above interface.

Each scheme is presented as a set of entities, the related attributes and relationships among entities. Furthermore for each scheme there are sections devoted to show the conflicts the users have recognized. At the beginning these sections are empty. As the use of Reconciler proceeds, they are filled in with the correspondences among concepts in the scheme. The first kinds

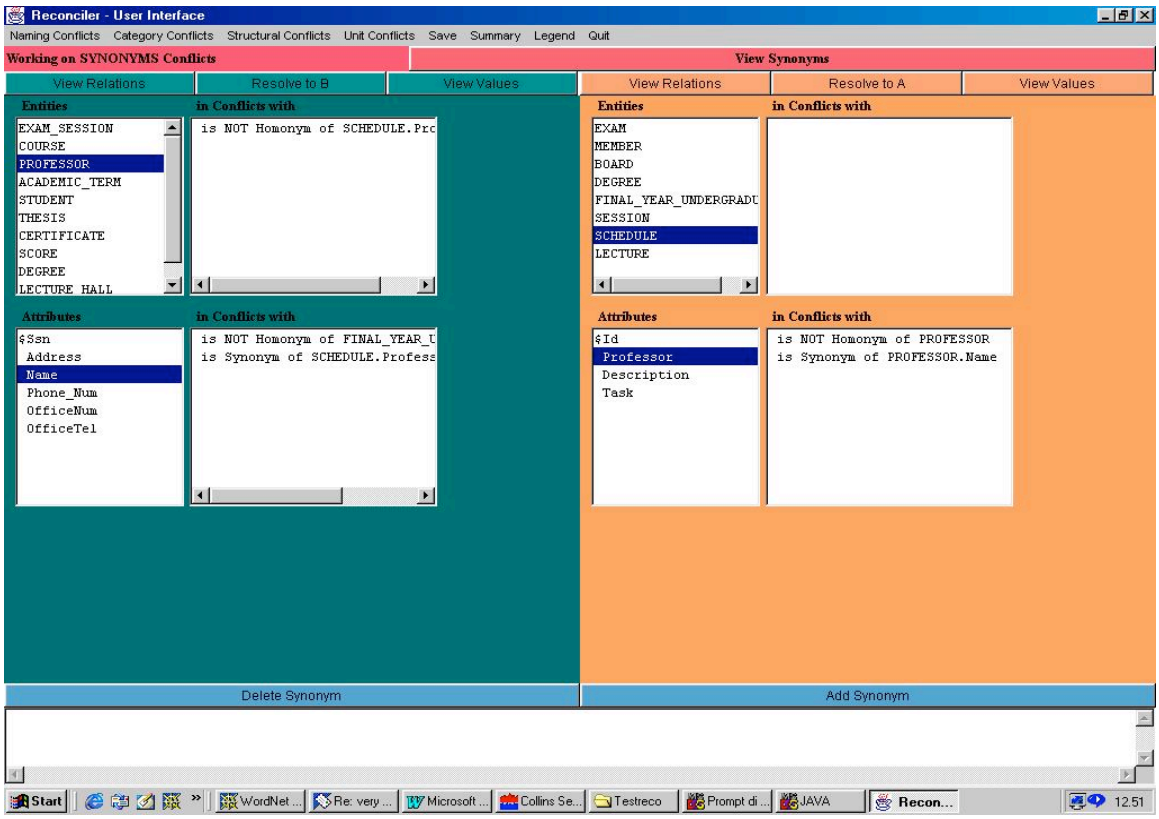

Figure 1: the Definition interface

of correspondences to consider are those related to Naming Conflicts: this involves the identification of homonyms and synonyms. Once the involved actors have determined this kind of correspondences, the system automatically detects the other kinds of conflicts, possible generated by the uncovered correspondences. Then the involved actors are called to solve them according to the various functionalities the interface provides them with.

The Communication interface

Once the schemes and the correspondences are constructed, they can be used in the communication phase. An algorithm uses the established information to elaborate communication messages exchanged through a *Communication Interface* (Figure 2) in order to improve their understandability by the receiver. This algorithm is composed of three steps. 1) The algorithm parses the content of the message in order to recognize the concepts used by the sender. Taking into account the semi-structure of the message, the recognized concepts are grouped into entities, attributes, relationships and conditions. 2) Then the algorithm searches for the correspondences involving the elements inside every single group, by using the information established in the previous reconciliation process. 3) The algorithm substitutes the concepts recognized during the first step with the corresponding concepts identified during the negotiation phase.

Implementation and future work

From the architectural point of view, the Reconciler was implemented as a standalone module characterized by the high modularity of its internal structure and by well defined information interfaces between sub-modules. This architectural choice makes the Reconciler module

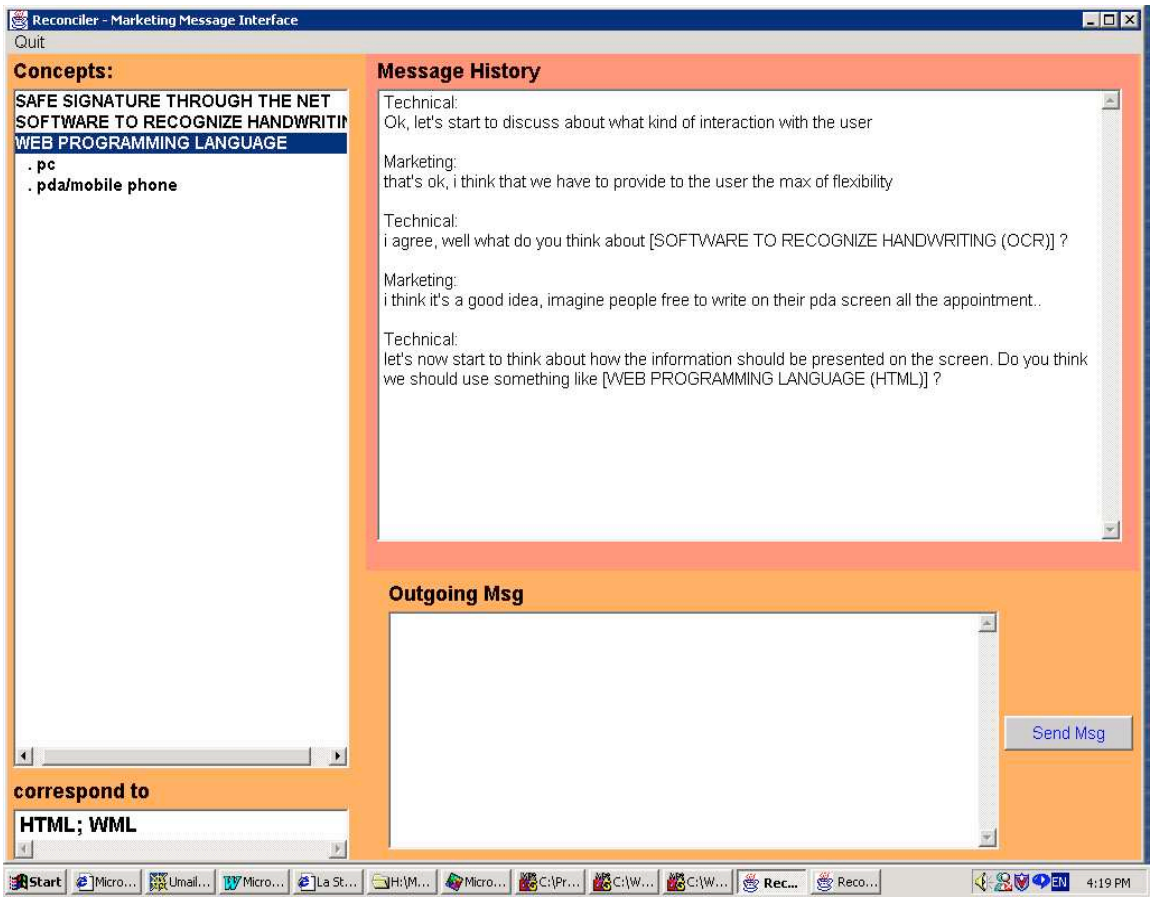

Figure 2: the Communication interface

easy to integrate in different contexts. This possibility is valuable for two reasons: first, to achieve adaptability to different user needs and technological environments both in terms of the different kinds of communication misunderstandings and different technological frameworks supporting collaboration; second, to start the experimentation (Mark et al, 2002) of the Reconciler functionality as a fundamental step to identify features to be incorporated in the user interface and to evaluate the capabilities of the Reconciler algorithm in real communication contexts. This development path is not fully achieved and only partial implementations have been realized. The outcomes of the experimentation encourage us to proceed toward a richer functionality using all the capabilities of the underlying transformation algorithm to be integrated in existing technological supports to cooperation. Moreover, we have to investigate more deeply the design of a Communication Interface that incorporates richer supports of communication without affecting its usability. We are confident that considering the proposed functionality as a support to a learning process for the involved local communities, oriented to increase the mutual awareness of their local points of view and not as a substitutive mediating technology, will allow us to reach a good trade-off between richness and usability.

References

Batini, C., M. Lenzerini, and S.B. Navathe (1986): A comparative analysis of metodologies for database schema integration. *ACM Computing Surveys*, vol. 18, no. 4, pp. 323-364.

Bonifacio, M., P. Bouquet, and P. Traverso (2001): Enabling Distributed Knowledge Management: Managerial and Technological Implications. *UPGRADE*, Vol III, no.1, pp 1-7

Conklin, J.and Begemann, M. L. (1988): gIBIS: A Hypertext Tool for Argumentation. *ACM Transactions on Office Information Systems*, 6(4):pp303-331

Dourish, P., J. Lamping, and T. Rodden (1999): Building bridges: customization and intelligibility in shared category management. In *ACM-Group99, Phoenis AZ*, ed. S. C. Haynes. ACM Press, pp. 11- 20.

Fischer, G., Lemke, A. C., McCall, R., and Morch, A. I.(1991):Making Argumentation Serve Design. *Human-Computer Interaction*, 6, 3&4 , 393-419

Karsenty L. and Brezillon P. (1995): Cooperative problem solving and explanation. *International Journal of Expert Systems With Applications*, 8(4), pp.445-462

Kent, W. (1978): *Data and reality: basic assumptions in data processing reconsidered*. New York: North Holland.

Mark, G., V. Gonzalez, M. Sarini, and C. Simone (2002) Reconciling different perspectives: an experiment on technology support for articulation. *Proceedings of the COOP2002 Conference*, Saint Raphaël (FR), 4-7 June 2002 (to appear).

Sarini M.,and C. Simone (2002): Recursive Articulation work in Ariadne. *Proceedings of the COOP2002 Conference*, Saint Raphaël (FR), 4-7 June 2002 (to appear).

Schmidt, K. and C. Simone (1996): Coordination Mechanisms: Towards a conceptual foundation for CSCW systems design. *Computer Supported Cooperative Work (CSCW). An International Journal*, vol. 5, no. 2-3.

Simone, C., G. Mark, and D. Giubbilei (1999): Interoperability as a means of articulation work. In *WACC'99, San Francisco*. ACM Press, pp. 39-48.

Simone, C. and Sarini, M. (2001). Adaptability of Classification Schemes in Cooperation: what does it mean? *Proceedings of ECSCW'01*, Bonn, Germany, September 18-20, Dordrecht: Kluwer.

Winograd, T. and Flores, F. (1986): *Understanding Computers and Cognition: A new foundation for design*. Addison Wesley, Reading, MA 1986

Wulf, V. (1997): Storing and retrieving documents in a shared workspace: experiences from the political administration. *Proc. of INTERACT'97*, London: Chapman & Hall, 469-476.

Zacklad, M., and Rousseaux F. (1995): Modelling Co-operation in the Design of Knowledge Production Systems: the MadeIn'Coop Method - An example in the field of C3I systems, in *Proceedings of COOP'95*, Juan-les-Pins, 24-27 January 1995, pp 1-19.