The Design of Convivial Multiagent Systems

Patrice Caire and Leendert van der Torre[1]

Abstract. In this paper, we consider the design of convivial multi-agent systems. Conviviality has recently been proposed as a social concept to develop multi-agent systems. In this paper we introduce temporal dependence networks to model the evolution of dependence networks and conviviality over time, we introduce epistemic dependence networks to combine the viewpoints of stakeholders, and we introduce normative dependence networks to model the transformation of social dependencies by hiding power relations and social structures to facilitate social interactions. We show how to use these visual languages in design, and we illustrate the design method using an example on virtual children adoptions.

1 Introduction

The focus of this paper is the social/organizational structure of a multiagent system. In particular, we are interested in the design of *convivial* multiagent systems, which is directly related to well studied issues such as groups and teams, norms and normative behavior, and coalition formation. First, we discuss the determining factors and the decisions we have to make concerning the actual convivial characteristics of the system. Following the TROPOS methodology, this process leads us to our dependence network model. A crucial step in this phase is to manage conflicting requirements such as reconciling freedom with exclusion and missing or incomplete specifications such as implicit agents goals. Second, we propose a representation of our model and present our formalism, initially expressing dependencies with static dependence network. We then express the sequence of different actors point of views, temporal dynamic networks. Third, we define the actors interactions and model a protocol.

We study the following research questions:

1. How to design the evolution of convivial social relations?
2. How to combine viewpoints from stakeholders?
3. How to incorporate normative aspects of conviviality?

The description level of this paper is methodologies and languages. To answer these questions we develop temporal dependence networks to model the evolution of dependence networks and conviviality over time, we introduce epistemic dependence networks to combine the viewpoints of stakeholders, and we introduce normative dependence networks to model the transformation of social dependencies by hiding power relations and social structures to facilitate social interactions.

The inspiration source of our work is political and social science. Empathy and reciprocity were foregrounded by Polanyi in 1964. "Individual freedom realized in personal interdependence" was tooled up by Illich in 1974 [12]. And in 1988, Putnam considered conviviality as a condition for civil society and social capital, a concept re-

ferring to the collective values of all social networks. One of the four themes of the European Community fifth framework program was entitled the "societe de l'information conviviale" (1998-2002) [18], which was translated as "the user-friendly information society." Today, a number of research fields such as computer supported cooperative work and social software aim at supporting users to interact and share data. Conviviality has recently been proposed also as a social concept to develop multi-agent systems [5].

As a running example, we use the design of a virtual adoption agency for instance on Second Life (SL). Adopting virtual children is a successful experience and a flourishing business on SL. Parents wishing to adopt a child must pay a fee to the adoption agency. The procedure typically involves that parents list themselves to advertise their profile to prospective children who can select them. The agency then matches children and parents and organizes a try-out period. There is no pressure. Once parents and children have made their decision, they simply come back to the agency to cancel the adoption if unhappy or otherwise to confirm it and get their adoption certificate and a ceremony. The experience must be convivial.

The conviviality literature discusses many definitions and relations with other social concepts, which we do not introduce in the formal model in this paper, referring to qualities such as trust, privacy and community identity. Also, in this paper we do not consider Polanyi's notion of empathy, which needs trust, shared commitments and mutual efforts to build up and maintain conviviality.

The layout of this paper is as follows. In Section 2 we discuss the social focus of this paper by explaining how the social concept "conviviality" can be used to develop multiagent systems in general, and their design in particular. In the following four sections we answer the research questions. In Section 3 we introduce temporal dependence networks to model the evolution of dependence networks and conviviality over time. In Section 4 we introduce epistemic dependence networks to combine the viewpoints of stakeholders. In section 5 we introduce normative dependence networks to model the transformation of social dependencies.

2 Convivial multiagent systems

In this section we discuss the use of social concepts in general, and "conviviality" in particular, for the development of multiagent systems.

Illich defines conviviality as "individual freedom realized in personal interdependence" [12]. We therefore model it using dependence networks [6, 16], representing on which agents and agent depends to fulfill its goals. An agent depends on a set of agents to fulfill one of its goals, when the set of agents has the power to fulfill the goal.

We define conviviality masks based on Taylor's idea that conviviality "masks the power relationships and social structures that govern

societies." [17, 5, 3]

A conviviality mask is a transformation of social dependencies by hiding power relations and social structures to facilitate social interactions.

2.1 Conviviality requirements

Requirements for multiagent systems say that systems must be convivial, whereas system researchers and developers use other concepts. To model the requirement, the developers may interpret the conviviality requirement as being autonomous to make suggestions, to react the discussion in the meeting to reach their goals, being proactive to take the initiative and being goal-directed, and most importantly being social by interacting with others to reach their goals.

When writing down requirements for user friendly multiagent systems, it is crucial to understand the inherent threads of conviviality, such as deception, group fragmentation and reductionism [5]. Whereas conviviality was put forward by Illich as a positive concept, also negative aspects were discussed. People are often not rational and cooperative to achieve conviviality [15] and unity through diversity [11] may lead to suppression of minorities. Taylor explores the contradiction that conviviality cannot exist outside institutions: i.e., the question "whether it is possible for convivial institutions to exist other than by simply creating another set of power relationships and social orders that, during the moment of involvement, appear to allow free rein to individual expression. Community members may experience a sense of conviviality which is deceptive and which disappears as soon as the members return to the alienation of their fragmented lives."

2.2 Conviviality ontology

The use of conviviality as a computer science concept ensures that considerations on the user-friendliness of multiagent systems get the same importance and considerations on the functionality of the system. For example, our experience with the development of a digital city in Europe is that computer engineers are focussed on filling in forms and developing menu structures and other interface issues, and do not take into account that a digital city should be a meeting place for human and artificial agents.

Conviviality is a useful high level modeling concept for organizations and communities, emphasizing the social side of them rather than the legal side. Erickson and Kellogg [9] say: "In socially translucent systems, we believe it will be easier for users to carry on coherent discussions; to observe and imitate others' actions; to engage in peer pressure; to create, notice, and conform to social conventions. We see social translucence as a fundamental requirement for supporting all types of communication and collaboration". Taylor studies conviviality in British pantomime and observes that: "conviviality masks the power relationships and social structures that govern societies."

2.3 Design of convivial systems

In this paper we study how convivial multiagent systems can be designed using our operationalized concept of conviviality. We illustrate our arguments and contributions with a running example on multiagent systems for virtual adoptions, where typically physical reality such as multiagent technologies interact with virtual and social realities.

The aim of social scientists to create conviviality by creating the desired conditions for social interaction, coincides with the aim of designers of multiagent systems. For example, Illich defines a convivial learning experience in which the teacher and the student switch roles, such that the teacher becomes the student and the student becomes the teacher. This role swapping emphasizes the role of reciprocity as a key component for conviviality. Parallelely the importance of reciprocity in conviviality was shown for instance in [10]. As a result, such role swapping scenarios can directly be used in multi-agent systems.

3 Temporal dependence networks

In this section, we propose a design methodology for convivial multi-agent systems based on the agent-oriented software development process, Tropos [1]. Key ideas in Tropos are first, that throughout the process phases, e.g. from early requirements to implementation, agents are endowed with intentionality. Second, the importance of very early phases of requirement analysis to allow for a profound understanding of the environment and of the interactions for the software to be built. This methodology guides designer through an incremental process, from the initial model of stakeholders, to refined intermediate models that, at the end, becomes the code.

3.1 Dependence networks

Multiagent systems technology can be used to create tools for conviviality. Illich defines conviviality as "individual freedom realized in personal interdependence" [12]. Dependence network is a tool that allows us to model this interdependence [6, 16]. In a recently published paper [5] dependence networks were formally defined as in Def. 1.

Definition 1 (Dependence networks) *A dependence network is a tuple $\langle A, G, dep, \geq \rangle$ where:*

- *A is a set of agents*
- *G is a set of goals*
- *$dep : A \times 2^A \to 2^{2^G}$ is a function that relates with each pair of an agent and a set of agents, all the sets of goals on which the first depends on the second.*
- *$\geq: A \to 2^G \times 2^G$ is for each agent a total pre-order on goals which occur in its dependencies: $G_1 \geq (a)G_2$ implies that $\exists B, C \subseteq A$ such that $a \in B$ and $G_1, G_2 \in depend(B, C)$.*

Nevertheless, this representation of conviviality is static and therefore has a limited field of application. In the next sub-section, we present our extension to encompass the temporal aspect of conviviality.

3.2 Temporal dependence networks

Before proposing our definition, we introduce our virtual adoption running example. The procedure typically involves that parents list themselves to advertise their profile to prospective children who, if they like the parents, can select them. The agency then matches children and parents and organizes a try-out period. Once parents and children have made their decision, they simply come back to the agency to cancel the adoption if unhappy or otherwise to confirm it and get their adoption certificate and a ceremony.

We start by informally listing critical stakeholders. We then identify the relevant goals and the social dependencies of the stakeholders represented as actors. In particular, the actor **Parent** is associated with the goal: adopt child, while the actor **Child** is associated with the goal: get adopted and **Virtual Agency** with the goal: provide adoption service.
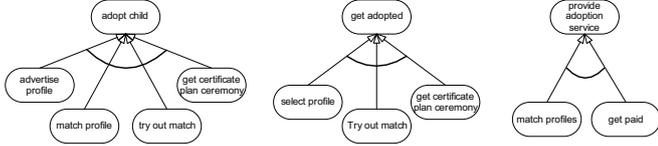

Figure 1. Decomposition of goals.

To enrich the model with a finer goal structure and elicit dependencies, we decompose each root goal into sub-goals. For instance, **Child** goal: get adopted, is decomposed into three sub-goals: select profile, try out match and get certificate - plan ceremony. In Fig. 1, a graphical representation of goal modeling is given through a goal diagram; AND decomposition only are shown, no OR decomposition, e.g. no alternate sub-goals.

The UML sequence diagram (Fig. 2), illustrates the interactions among the stakeholders and how operations are carried out. The diagram shows time incrementing vertically. In particular, the diagram models the interaction among the three Users: **parent**, **agency** and **child**. The interaction starts with the advertise profile request by the **parent** to the **agency** and ends with the pay fee by the **parent** to the **agency**. We note that the match ok sent by both **parent** and **child** can be asynchronous. Moreover, the **agency** sends the adoption certificate and the plan ceremony to both **child** and **parent**.
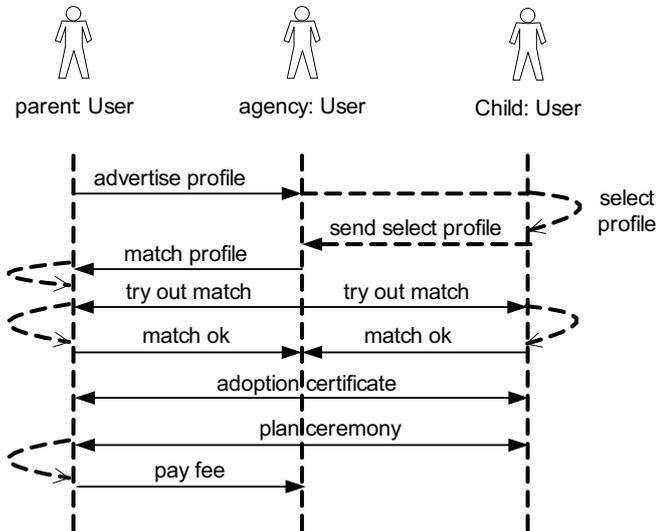

Figure 2. Actor diagram modeling the stakeholders for the virtual adoption domain.

Based on actor diagrams and goal decomposition, we proceed with a goal analysis taking each actor point of view. The objective is to

obtain a set of strategic dependencies among the actors. We therefore perform an iterative analysis on each goal until all are analyzed. We build a succession of dependence networks from each actor point of view.

With temporal dependence networks, we aim at analyzing the evolution of dependence networks and conviviality over time. We identify the most relevant interactions in our running example and build a model with the key succession of dependence networks.

Definition 2 (Temporal dependence networks) *A dependence network is a tuple $DP = \langle A, G, goals, dep \rangle$ where:*

- *A is a set of agents*
- *G is a set of goals*
- *T is the set of natural numbers*
- *$goals : T \times A \to 2^G$ is a function that relates with each pair of a sequence number and an agent, the set of goals the agent is interested in.*
- *$dep : T \times A \times 2^A \to 2^{2^G}$ is a function that relates with each triple of a sequence number, an agent and a set of agents, all the sets of goals on which the first depends on the second if the third creates the dependency.*

We use this structure to model our example (Fig. 3). Note that the set of agents does not change, but the goals of the agents and the dependencies among them, changes over time.

Agents $A = \{P, C, VA\}$ and
Goals $G = \{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}\}$
We thus have the following sequence of dependence networks:
$DP_4 = \langle A, G, goals_4, dep_4 \rangle$, where:

- $goals(4, VA) = \{\{g_5, g_6, g_7\}\}$: In dep_4, the goals of agent *VA* are to provide adoption service, to get paid and to match parent-child profiles.
- $goals(4, P) = \{\{g_1, g_{10}\}\}$: In dep_4, the goals of agent P are to adopt a child and to try out match.
- $goals(4, C) = \{\{g_8, g_{10}\}\}$: In dep_4, the goals of agent C are to get adopted and to try out match.
- $dep(4, VA, \{P, C\}) = \{\{g_7\}\}$: In dep_4, agent *VA* depends on agents P and C to achieve goal g_7: match parent-child profiles.
- $dep(4, P, \{C\}) = \{\{g_{10}\}\}$: In dep_4, agent P depends on agents C to achieve goal g_{10}: try out match.
- $dep(4, C, \{P\}) = \{\{g_{10}\}\}$: In dep_4, agent C depends on agents P to achieve goal g_{10}: try out match.

In our notation, dep_i refers to the temporal dependence network where $i \in T$ and denotes the ith sequence, P refers to agent **Parent**, C to agent **Child** and *VA* to agent **Virtual Agency**.

4 Epistemic dependence networks

In our running example, we use the Tropos methodology [1], with the difference that we include neither plans nor resources. However similarly to Tropos, we identify actors which depend on each other to achieve their hardgoals, simply referred to as goals, and softgoals, the latter being typically used to model non-functional requirements and "having no clear -cut definition and/or criteria for deciding wheter they are satisfied or not" [1]. In Fig. 4, we show an *actor diagram* for the virtual adoption. In particular, **Parent** is associated with the goal: adopt child, and the softgoal: get nice child. Similarly, **Child** is associated with the goal: get adopted and the softgoal get nice parents while **Virtual agency** wants to provide adoption service and has the
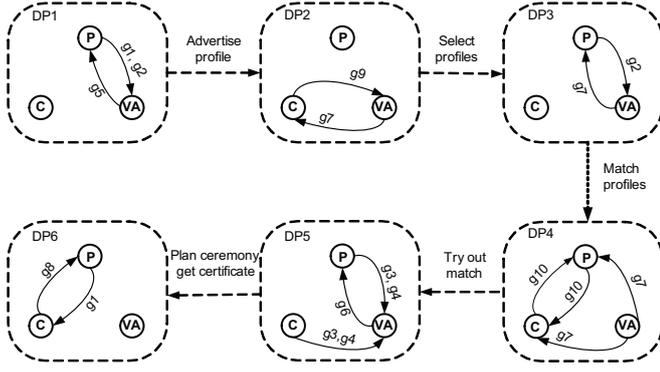
Figure 3. DP sequences

softgoal to provide a good service. Finally, the diagram includes one softgoal dependency where **Parent** depends on **Virtual agency** to fulfill the softgoal: adoption fee well spent.
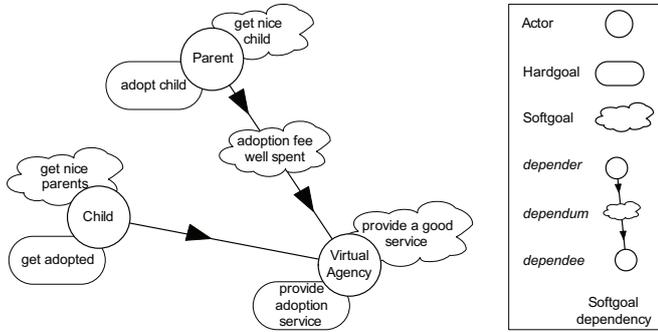

Figure 4. Actor diagram modeling the stakeholders for the virtual adoption.

Temporal dependence networks allow us to capture a relation from a specific point of view and at a specific time. Unfortunately, it is not sufficient for the situation we want to model, so in the next section, we try to answer this question by introducing a new model that will allow us to capture a more global view from the system point of view.

In order to model such system, we use the epistemic dependence network formally defined as Def. 3.

Definition 3 (Epistemic dependence networks) *An epistemic dependence network is a tuple $DP = \langle A, G, T, goals, dep \rangle$ where:*

- *A is a set of agents*
- *G is a set of goals*
- *T is the set of natural numbers*
- *goals : $T \times A \to 2^G$ is a function that relates with each pair of sequence number and an agent, the set of goals the agent is interested in.*
- *dep : $A \to T \times A \times 2^A \to 2^{2^G}$ is a function that expresses from the point of view of an agent $a \in A$, the dependence relation between another agent $b \in A$ and a set of other agents regarding the goals of agent b in a sequence $t \in T$.*
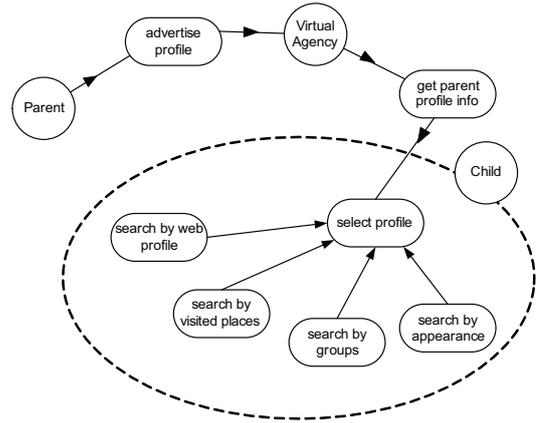

Figure 5. Goal diagram for the goal select profile and dependencies between the actor Child and other environment's actors.

If we consider Fig. 5 the starting *goal diagram*, the three steps of this design process are:

1. Goal delegation: Each goal of any actor may be delegated to any other actor, already existing or new. It proceeds with the analysis of goals from the point of view of each actor. This generates a network of delegation between stakeholders, external actors and the system. The inclusion of new actors and sub-actors and subsequently, the delegation of sub-goals to sub-actors continues until all goals have been analyzed. Actors that contribute to the requirements are also included.
2. Goal decomposition: Goals and softgoals are further decomposed into sub-goals or found not reachable. Through this refinement process a goal hierarchy is created where leaf goals represent alternatives to root goals. Moreover, some identified sub-goals become reasons for new dependencies with new actors. Therefore, dependencies in actors diagrams must be revised.
3. When all actors fulfill their goals, all the goals have been analyzed and the root goals are satisfied then, this design process is complete.

4.1 Example

In our running example, let's consider the set of agents
$A = \{P, C, VA, AS\}$, where *AS* is the **Adoption System**.
$dep(P) = (2, VA, \{C\}) = \{g_9\}$: **Parent** believes that in sequence 2, **Adoption System** depends on **Child** to achieve goal g_9: select profile.

We express Fig. 6 as follows: $dep(AS) = (2, P, \{C\}) = \{g_9\}$: **Adoption System** believes that in sequence 2, **Parent** depends on **Child** to achieve goal $\{g_9\}$: select profile. We note that there is no dependency from **Adoption System** towards **Adoption System** for the goal: select profile.

With Fig. 5 and 6, we explain the iterative design process from the Tropos methodology that are tool supported [14].

To explain what is the delegation process, and as an example, we here give a partial view on goal: select profile.

To start, we have the goal of **Child**: select profile. After analyzing the rational for this goal from each actor point of view, we delegate this goal to the new actor, the system-to-be **Adoption System** . We continue by analyzing each sub-goal.

We then identify the capabilities needed by **Adoption System** to fulfill all the four identified sub-goals: search by web profile, search by visited places, search by groups and search by appearance. In order for this latter sub-goal to be fulfilled, we add a new goal: provide photo/video and a new dependency from **Adoption System** towards **Parent**. Similarly, in Fig. 5 the sub-goal: search by web profile has no dependency while in 6 a new dependency from **Adoption System** towards **Child** has been created to fulfill the subgoal: know web address. Of course, each dependency must be mapped to a capability. We then define a set of agent types and assign each of them one or more capabilities. The specification of agent's goals, beliefs, capabilities and the communication between the agents depends on the adopted platform and the chosen programming language. We therefore leave this part for further work.
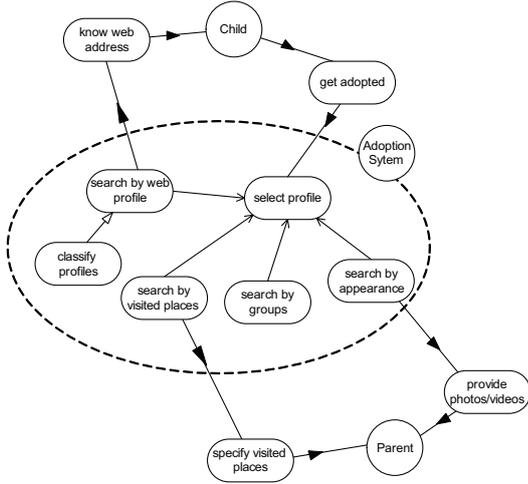

Figure 6. Goal diagram for the goal select profile and dependencies between the actor Adoption System and other environment's actors.

4.2 Nested dependencies

We first mention that by *nested* we simply mean a belief produced and only accessible by an agent a and about another agent b, e.g. inaccessible to all others. For instance, empathy provides a way to know what another agent's preference is, and therefore to better adapt to it, allowing for a convivial relation, whereby agents contribute to each other. In our running example, let's assume that **Parent** believes that **Child** depends on it, **Parent**, for its goal: select profile. Let's further assume that **Child** believes that **Parent** depends on it to advertise parent profile, for example if **Child** first had to publish an announcement on a board that it is seeking parents to be adopted by. We write:

$dep(P) = (1, C, \{P\}) = \{g_9\}$: agent P believes that in sequence 1, agent C depends on it, P to achieve its goal g_9: select parents' profile.

$dep(C) = (1, P, \{C\}) = \{g_2\}$: agent C believes that in sequence 1, agent P depends on it, C, to achieve its goal g_2: advertise its profile.

5 Norms and masks

There are many different kinds of goals, some goals may be considered normative, others personal. Agents do not only have personal goals, they also have normative goals, e.g. goals imposed by the procedures. We propose a further extension of epistemic dependence networks that we call "Normative epistemic dependence networks" in order to take into account the differences in the two kinds of goals as well as obligations and violations.

Definition 4 (Normative epistemic dependence networks) *A dependence network is a tuple*
$$DP = \langle A, G, N, O, V, T, goals, dep \rangle \text{ where:}$$

- *A is a set of agents*
- *G is a set of goals*
- *N is a set of norms*
- *T is the set of natural numbers*
- *$O : N \times A \to 2^G$ is a function that associates with each norm and agent the goals the agent must achieve to fulfill the norm; We assume for all $n \in N$ and $a \in A$ that $O(n, a) \in power(\{a\})$;*
- *$V : N \times A \to 2^G$ is a function that associates with each norm and agent the goals that will not be achieved if the norm is violated by agent a; We assume for each $B \subseteq A$ and $H \in power(B)$ that $(\cup_{a \in A} V(n, a)) \cap H = \emptyset$.*
- *goals $: T \times A \to 2^G$ is a function that relates with each pair of sequence number and an agent, the set of goals the agent is interested in.*
- *dep $: A \to T \times A \times 2^A \to 2^{2^G}$ is a function that expresses from the point of view of an agent $a \in A$, the dependence relation between another agent $b \in A$ and a set of other agents regarding the goals of agent b in a sequence $t \in T$.*

5.1 Example 1

We explain with an example how to use our formalism and model normative situations. In sequence 2 of our running example, while **Child**'s obligation to select profiles is a normative goal, **Child**'s desire to select the parents it prefers is a personal goal. In this case, personal and normative goals coincide:

The goal g_9, to select parents' profile, is both a personal goal and a normative goal, that is, $goals(2, C) = g_9 \cup O(2, C) = g_9$, where $g_9 \in PG_C$: in sequence 2, agent C has the goal and the obligation to select parents' profiles g_9, where *PG* is personal goal.

$G_C = \cup O(n, C) \cup PG_C$, where $G_C \in G$ is the set of normative goals of agent $C \in A$, $n \in N$ is an adoption norm, $O(n, C)$ is the obligation for C to respect norm n resulting in its normative goals, and $PG_C \in G$ are the personal goals of C.

5.2 Example 2

In this paragraph, we explain the notions of positive and negative consequences to a norm violation. A positive consequence is adding a goal to the existing ones whereas a negative consequence forbid the realization of a goal. We further explain with our example. Let's assume that the parent believes that, in sequence 2, the child depends on the virtual agency to hide its information to parents. However, the parent violates its obligation to respect it and looks up the child's information. One possible sanction is that the parent cannot advertise its profile at the agency any longer, which means that this goal is unrealizable. In the case of the violation sanctioned by the removal of the goal g_2, the obligation $O(n_2, P)$ is not possible any longer as agent P cannot advertise its profile at the agency, it cannot depend on the agency to get the child information any longer. Moreover, agent P cannot achieve its personal goal g_1: adopt a child, any longer as g_2 is a normative goal needed for agent P to achieve g_1. And the

violations are: $V^-(n_2, P) = g_2$: agent P violating norm n_2 will not be able to achieve goal g_2, advertise its profile, because g_2 is removed.

As a consequence, the parent cannot adopt a child. Another possible sanction is that the parent must make a donation, e.g. pay a fee, in which case a new goal is added to the parent. As a result, until the parent has fulfill this new obligation, it cannot continue the process.

$dep(P) = (2, C, VA) = g_{14}$: agent P believes that in sequence 2, agent C depends on agent VA to achieve its goal g_{14}: no child look up. Where the obligations are:

$O(n_1, C) = g_9$: agent C has the obligation to fulfill norm n_1 to achieve goal g_9, select parent profile.

$O(n_2, P) = g_{14}$: agent P has the obligation to fulfill norm n_2 to achieve goal g_{14}, no look up child.

$V^+(n_2, P) = g_{15}$: agent P violating norm n_2 will not be able to achieve goal g_2, advertise its profile, because a new goal g_{15}, make a donation, is added. Until this new goal is achieved, g_2 cannot be achieved.

In the case of the violation sanctioned with the addition of the goal g_{15}, we note that a mechanism is needed to make sure that the new goal is fulfilled before agent P can further proceeds.

6 Related work

Castelfranchi [6] introduces concepts like groups and collectives from social theory in agent theory, both to enrich agent theory and to develop experimental, conceptual and theoretical new instruments for the social sciences. For further work on the use of the concept of conviviality in computer science and multiagent system see [3, 2, 4]. A large body of work on design has been produced, to only cite a few: the AOSE methodology [13], GAIA [7], the PASSY methodology [8].

7 Summary

Conviviality has recently been proposed as a social concept to develop multi-agent systems. In this paper we introduce temporal dependence networks to model the evolution of dependence networks and conviviality over time, we introduce epistemic dependence networks to combine the viewpoints of stakeholders, and we introduce normative dependence networks to model the transformation of social dependencies by hiding power relations and social structures to facilitate social interactions. We show how to use these visual languages in design, and we illustrate the design method using an example on virtual children adoptions.

REFERENCES

[1] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[2] P. Caire. How to import the concept of conviviality to web communities. *Web communities*, 2009.

[3] P. Caire and L. van der Torre. Convivial ambient technologies: Requirements, ontology, and design. *The Computer Journal*, 2009.

[4] P. Caire and L. van der Torre. Temporal dependence networks for the design of convivial mul- tiagent systems (short paper). In *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AA- MAS 2009)*, 2009.

[5] P. Caire, S. Villata, G. Boella, and L. van der Torre. Conviviality masks in multiagent systems. In L. Padgham, D. C. Parkes, J. Müller, and S. Parsons, editors, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AA- MAS 2008)*, pages 1265–1268. IFAAMAS, 2008.

[6] C. Castelfranchi. The micro-macro constitution of power. *Protosociology*, 18:208–269, 2003.

[7] L. Cernuzzi and F. Zambonelli. Dealing with adaptive multi-agent organizations in the gaia methodology. In J. P. Müller and F. Zambonelli, editors, *AOSE*, volume 3950 of *Lecture Notes in Computer Science*, pages 109–123. Springer, 2005.

[8] A. Chella, M. Cossentino, L. Sabatucci, and V. Seidita. Agile passi: An agile process for designing agents. *Comput. Syst. Sci. Eng.*, 21(2), 2006.

[9] T. Erickson and W. A. Kellogg. Social translucence: an approach to designing systems that support social processes. *ACM Trans. Comput.-Hum. Interact.*, 7(1):59–83, 2000.

[10] E. R. Gomes, E. Boff, and R. M. Vicari. Social, affective and pedagogical agents for the recommendation of student tutors. In *Proceedings of Intelligent Tutoring Systems*, 2004.

[11] W. Hofkirchner. Unity through diversity.dialectics - systems thinking - semiotics. *Trans, Internet journal for cultural sciences*, 1(15), 2004.

[12] I. Illich. *Tools for Conviviality*. Marion Boyars Publishers, London, August 1974.

[13] J. Odell, P. Giorgini, and J. P. Müller, editors. *Agent-Oriented Software Engineering V, 5th International Workshop, AOSE 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers*, volume 3382 of *Lecture Notes in Computer Science*. Springer, 2004.

[14] L. Penserini, A. Perini, A. Susi, and J. Mylopoulos. High variability design for software agents: Extending tropos. *TAAS*, 2(4), 2007.

[15] M. D. Sadek, P. Bretier, and E. Panaget. ARTIMIS: Natural dialogue meets rational agency. In *International Joint Conferences on Artificial Intelligence (2)*, pages 1030–1035, 1997.

[16] J. S. Sichman and R. Conte. Multi-agent dependence by dependence graphs. In *Procs. of The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002*, pages 483–490. ACM, 2002.

[17] M. Taylor. Oh no it isn't: Audience participation and community identity. *Trans, Internet journal for cultural sciences*, 1(15), 2004.

[18] C. Weyrich. Orientations for workprogramme 2000 and beyond. Information society technologies report, Information Society Technologies Advisory Group, September, 17 1999.