TASK MODEL REPRESENTATION OF A COLLABORATIVE SYSTEM USING HYPERMEDIA NETWORKS

Beatriz Jiménez, Miguel Sánchez, Francisco L. Gutiérrez and Nuria Medina
Department of Computer Languages and Systems, University of Granada
Periodista Daniel Saucedo Aranda s/n, E-18071 Granada, Spain

Keywords: Task models, hypermedia networks, collaborative systems.

Abstract: This paper presents a method for representing the task models of collaborative systems using hypermedia networks. The aim is to represent these models flexibly and extensibly so that they may be managed in web environments. This approach considers three relevant aspects of collaborative systems: the organization's structure (roles), the tasks to be performed by organization members, and the resources necessary for carrying out these tasks. The AMENITIES methodology will be used to represent the initial models which describe these three aspects in collaborative systems. These initial models will be the basis for representing the transformation process for achieving our hypermedia network.

1 INTRODUCTION

One important feature of current systems (Vernadat 1996) is the existence of collaborative processes where different users communicate and cooperate in order to achieve common aims. Unfortunately, when we decide to specify a collaborative system we find that the actor's behavior and interaction to carry out different tasks cannot be defined statically as a result of the changing context of such systems. These context changes have their effect on task management, and while the development of a system is complex, it is also necessary to pay special attention to dynamic task management.

Our research group GEDES is currently working on the development of a Web Service Architecture (WSA) for collaborative systems (Gutiérrez et al., 2006a), the main objective of which is to enable the adaptation and evolution of system behavior and the security and coordination of task management. While task models allow us to pinpoint the problems of such systems and to find solutions for them at a conceptual level, they cannot be executed directly on the proposed architecture and so another representation is necessary. In this paper, we will represent cooperative task models with hypermedia networks to obtain a flexible, extensible and easy-to-integrate representation with the different architecture services at the execution level. Our idea is to use hypermedia networks to control the workflow generated between different data in our system.

Section 2 of this paper presents an existing approach for workflow collaboration. Section 3 describes a task model approach that makes up a complete model for collaborative system specification. Section 4 details our approach for representing task models using hypermedia networks with an example. Finally, Section 5 outlines our conclusions and our future lines of research.

2 RELATED WORK

Web services are components which are based on WSDL (W3C-WSDL), UDDI (OASIS-UDDI), and SOAP (W3C-SOAP) industry standards, and enable different organizations to be connected independently of the platforms and languages used in each one (Leymann, 2003). In recent years, many workflow approaches have been developed and these allow different processes in a system with complementary skills to cooperate. These approaches divide an application into two strictly separate layers: the top layer (the business process) which is written in the Business Process Execution Language for Web Services (BPEL4WS or BPEL) (Leymann & Roller, 2000) and represents the application's flow logic; and the bottom layer (the web services) which represents the application's

Jiménez B., Sánchez M., L. Gutiérrez F. and Medina N. (2007).
TASK MODEL REPRESENTATION OF A COLLABORATIVE SYSTEM USING HYPERMEDIA NETWORKS.
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Society, e-Business and e-Government / e-Learning*, pages 113-120
DOI: 10.5220/0001271801130120

function logic. This structure has several advantages over more conventional approaches: firstly, both the underlying business process and the invoked web services can be changed with no impact on the other web services within the application or on the web services that the business process represents; and secondly, the application can be developed and tested in two separate stages: the business process is developed and tested independently of the development and testing of the individual web services. As a result, it is therefore possible to easily make changes to the application.

The advantages provided by web services have been recognized by the UML community, and have been taken advantage of for UML to BPEL mapping and for the relations corresponding to Model-Driven Architectures (MDA) (Gardner, 2003). None of these web service standards and approaches, however, provides support for the coordination of activities where the sequence of activities to be carried out is undefined - a typical situation in collaborative systems. Correspondingly, our work presents an approach to enable navigation within a hypermedia structure by means of the flows of aims and subaims which have not been pre-established, in order to obtain effective coordination within collaborative systems where the uncertainty common to task models makes it impossible for a pre-established sequence to be followed.

3 TASK MODELS

Task models are defined as logical descriptions of the activities for obtaining the user's aims and goals and for showing how the user perceives the system (Van der Veer & van Welie, 2000; Paterno, 2001). Taking into account this definition and the need to describe tasks in collaborative organizations, task models are an ideal means of specifying what information is used in task management and how these tasks interrelate with each other.

Figure 1 shows a conceptual model diagram for describing tasks in an organization. This model is part of a complete model defined in the ADACO research project for modeling collaborative systems (Gutiérrez, 2006b). Tasks are classified into abstract, individual, and team tasks. While abstract tasks are tasks that gather other tasks, team tasks are one or more tasks that are carried out by user groups sharing common aims.

Both abstract tasks and team tasks eventually consist of individual tasks. In team tasks, the individual tasks will be performed by different team members (always under the influence and dynamics of teamwork) with a shared but not clearly assigned responsibility. Individual tasks are considered to be application tasks, interaction tasks, or user tasks .In order to establish this classification we follow Paternó's CONCURTASKTREE approach (Paternó, 1999) for modeling collaborative tasks.

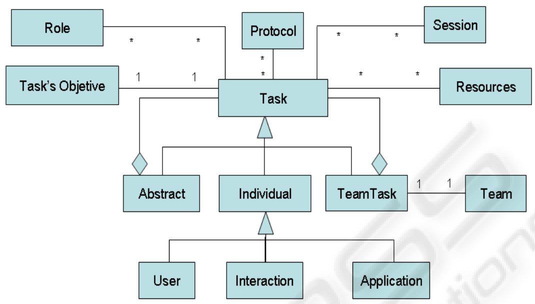

Figure 1: Task model.

By analyzing the conceptual model, we extract a set of considerations to obtain a hypermedia representation. Firstly, it is necessary to bear in mind that the reason why a process is successful or not will depend not only on its logic of application but also whether the aims associated with each task have been completed. Secondly, we must bear in mind the relation between actors and tasks: a task is not directly related with the actor, but the roles they carry out show their responsibilities in the tasks to be performed. There is therefore an indirect relation between actors and tasks through the roles. Finally, but just as important, another aspect in task coordination in collaborative processes is the control and presence of resources associated with actions in a task. Protocol and session concepts, which are part of the task model, will not be taken into account in this work. In the following section, we will present and provide examples of our approach for representing task models using hypermedia networks.

4 HYPERMEDIA REPRESENTATION

The advantages of a hypermedia representation are that information structuring, navigation facilities, and simultaneous use of resources (for example) allow us to organize task model information in an extensible and flexible way so that these models can be managed in the web environment. From the observations presented in the previous section, in order to carry out a hypermedia representation of task models, it is necessary to consider and represent three relevant aspects: the organization's structure (roles), the tasks that must be performed by the organization members, and the resources necessary to carry out these tasks. For each aspect, we create a particular hypermedia structure and by joining all these structures together we obtain the global hypermedia network which represents the collaborative system.

In order to make the representation process more comprehensible, all the steps will be illustrated by studying a particular example which illustrates the web version of the board game *Pictionary*. The game is carried out as a collaborative task with different users taking part and playing different roles. In the game *Pictionary*, a user can either play or watch other users play. It is necessary for users to register in the system if they want to play a game but not if they want to watch one.

A game will begin when there are at least four registered users, divided into two or more teams of two or more players. The player whose turn it is throws the die, moves a counter and picks a card from the pack. The drawing phase then begins and players must then draw pictures for their team mates (in the case of a "*TeamPlay*" card) or all the players (in the case of an "*AllPlay*" card) to guess the word. This stage finishes when a player guesses the picture or the timer rings, and it is then another player's turn. The game continues until one team reaches the finish and the team with the highest score wins. A moderator watches the game throughout to ensure that the rules are followed correctly.

The following sections 4.1, 4.2 and 4.3 will describe the representation of the organization's structure, the tasks, and the resources. In the three cases, AMENITIES (Garrido, 2003), a structured method for cooperative models, is used to represent the initial models that describe the system.

4.1 Representation of the Organization's Structure

The first step in our hypermedia representation is to analyze the actor's behavior in the system: role specification and role change in the organization. Figure 2 shows the organization of the system by means of an *Organization Diagram* used by AMENITIES to describe organizations using a special type of UML state diagram, where roles in the organization are treated as states and role changes as changes of state. The system that we want to represent is structured as a general organization (which we call *Pictionary*) and a more specific suborganization (which we call *Team*).

In the *Pictionary* organization, we can identify four roles: *Moderator*, *URegister*, *UGuest*, and *Observer*. The rhombus joined to the initial state (black dot) shows the initial possibility of acquiring two different capacities. Any actor acquiring the capacity *guest?* can carry out the *UGuest* role, but if an actor acquires the capacity *authorized?*, they will carry out the *URegister* role. The remaining roles performed by an actor will depend on the capacities acquired as the game evolves.

Actors carrying out the initial role *URegister* could change to the *Moderator* role if they decide to moderate a game (*moderate?* capacity), or to the *Observer* role if they decide to observe a game but not to play (*observer?* capacity), or to the *Player* role in the *Team* organization if they decide to play a game (*play?* capacity). Actors carrying out the *UGuest* initial role could only change to the *URegister* role when they acquire *registered?* capacity. The cardinality *2..NMax* in the *Team* flowchart indicates that a game begins when there are at least four registered users, divided into two or

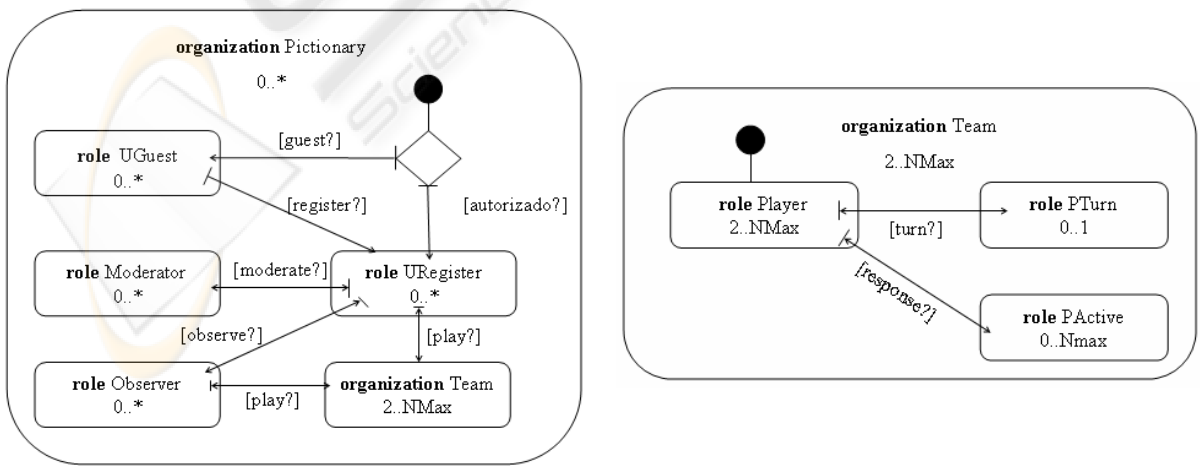

Figure 2: Organization Diagram for *Pictionary* and *Team*.

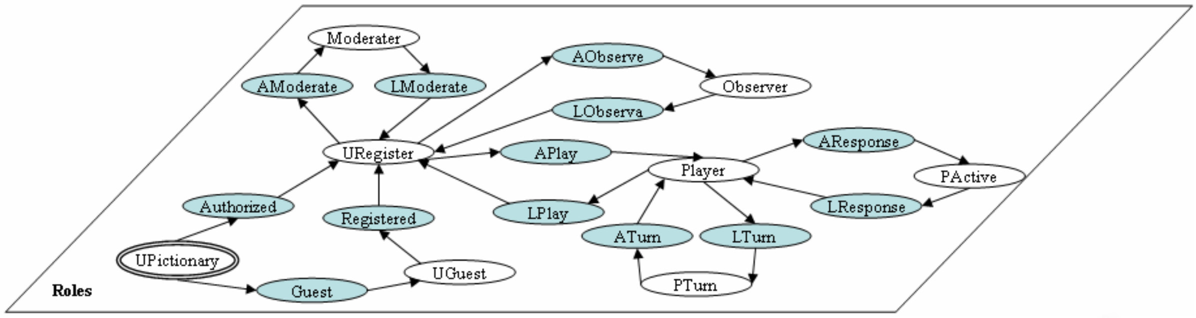

Figure 3: Hypermedia network for organization *Pictionary* and *Team.*

more teams of two or more players. Actors with the *Player* role could then change to the *PTurn* role after they acquire the *turn?* capacity. This role allows the actor to throw the dice, pick a card and draw a picture. The remaining actors with the *Player* role will change to the *PActive* role if they acquire the *response?* capacity to guess a picture.

In order to represent the flowchart information by means of a hypermedia structure, an ontological representation of roles and transitions is necessary to represent role changes. Consequently, two types of nodes are defined in our hypermedia representation: nodes to represent roles, and nodes to represent capacities and/or laws that produce role changes.

There are also two types of transitions between the states in the flowchart: non-additive transitions (original role is left and cannot be recovered later) which are represented by a single arrow in Figure 2; and additive transitions (original role is recovered later) which are represented by a double arrow in Figure 2. In order to represent this information in our hypermedia structure, in the case of non-additive transitions, there is a single link from the node representing the origin role to the node representing the destiny role. In the case of additive transitions, however, it is necessary for there to be two links in opposite directions between the origin and destiny roles with two intermediate nodes to represent the changes to *acquire* the destiny role (see node *AModerate* in Figure 3); and the changes to recover the origin role and *leave* the destiny role (see node LModerate in Figure 3). For organizations with no single initial state (in the example there are two different access modes *URegister* or *UGuest* depending on the capacity that has been acquired), we must include an additional node (see *UPictionary* in Figure 3) to represent these initial role changes. This initial node is represented by a double ellipse node.

Figure 3 shows the complete hypermedia graph for the flowchart in Figure 2. The hypermedia model obtained allows us to navigate through the possible role sequence in the system. At execution time, it can be used to control the user's authorization to change from a certain role to another.

4.2 Task Representation

The second step in our hypermedia representation is to analyze task management in the system: tasks carried out by a role, concurrent tasks, event triggering, etc. In AMENITIES, the *Role Diagram* is the connection between the organizational structure and the tasks to be carried out by members in the organization. Figure 4 shows the association between tasks and roles for the specific role *JTurn*.

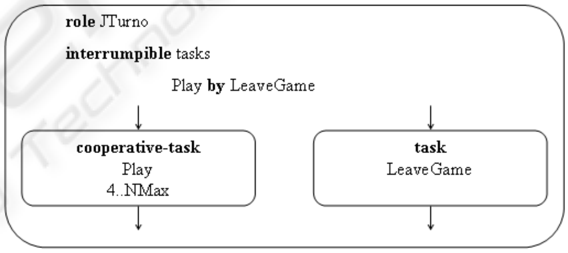

Figure 4: Role Diagram *JTurn.*

This role can carry out two tasks: a collaborative task, *Play,* which we will describe later, and an individual task, *LeaveGame*. The cooperative task *Play* can be interrupted by the task *LeaveGame* at any time. The cardinality of the cooperative task *Play* establishes that a minimum of four players are needed for a game to start. The representation of this diagram produces a new graph (level *Role-Task* in Figure 5) that will be the connection between the organizational hypermedia graph (level Roles in Figure 5) and the tasks to be carried out by members in the organization (level *Tasks* in Figure 5). Each task is represented by a node on the hypermedia graph. A link can represent two different concepts: execution or interruption. More specifically, there will be a link from each task node to the role node that executes it; there will also be a link from each interrupted task node to the task node or nodes which interrupted it (see *Play* and *LeaveGame* nodes on the *Role-Task* level in Figure 5.

The next step will be to complete the hierarchical task level by considering the detailed specification of the execution of each task. Figure 6 shows the AMENITIES specification for the cooperative task *Play* by means of a *Task Diagram* and the subactivity *TurnGame* (which belongs to the task *Play*) by means of a *Subactivity Diagram*. AMENITIES uses the *Task Diagram* to describe the action sequences for an activity. The cooperative task *Play* comprises two actions (*BeginPlay* and *FinalizePlay*) and one subactivity (*TurnGame*), which is represented by two states joined by a transaction in the bottom left-hand corner. AMENITIES uses the subactivty term to define work units composed by a set of actions and other subactivities to structure tasks, The action term to describe an atomic work unit. The *times no-winner* clause shows us that *TurnGame* will be instantiated while winner is not found. The subactivity *TurnGame* is defined with two main concurrent constructions (between solid lines).

The execution of the first set of *actions/subactivities ThrowDice, MoveCardGame*, PopCard, *ThrowCrono DrawCard* and *ResponserCard* finish when all the actions/subactivities have finished (transition with no event), and the subactivity *ModerateTurnGame* finishes when the event *FinalizeTurnGame* occurs. The subactivity *ThrowDice* is executed by the role *PTurn*. The subactivities *MoveCardGame* and *PopCard* will be executed by the system. After *PopCard* we can obtain two possibilities: if an "*AllPlay"* card is picked, the event *AllPlay* is triggered; if a *"TeamPlay"* card is picked, the event *TeamPlay* is triggered. The rest of the sequence is defined with a new concurrent construction: the subactivity *DrawCard* which is executed by the role P*Turn* and the subactivity *ResponseCard* which is executed by the role *PActive*. Both subactivities finish when all their actions/subactivities have finished (transition with no event) or when the *CronoZero* event is triggered.

In order to represent the information shown in a *Task Dagram* by means of a hypermedia graph, an ontological representation of actions/ subactivities and the transitions between them is necessary. In this case, we consider three types of nodes: nodes which represent actions/subactivities; nodes which represent sending of signs and events; and nodes which represent the global aim associated with a task.

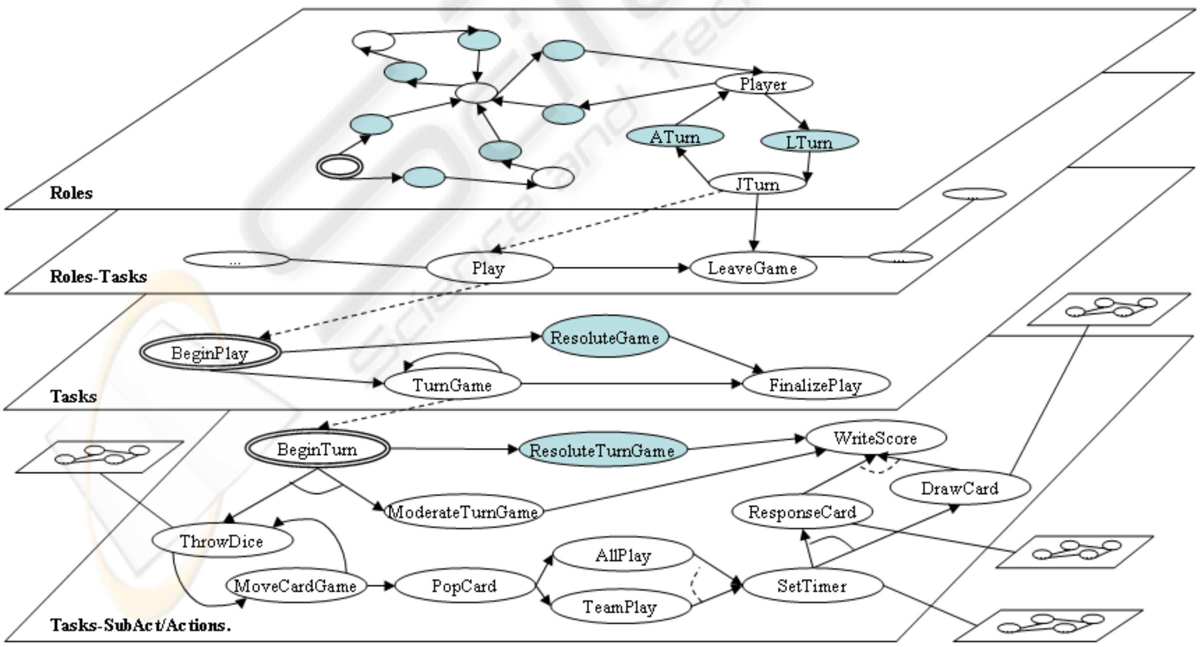

Figure 5: Hypermedia network for the role-task connection *PlayerTurn-Play* and for the task-subactivity connection *Play-TurnGame.*

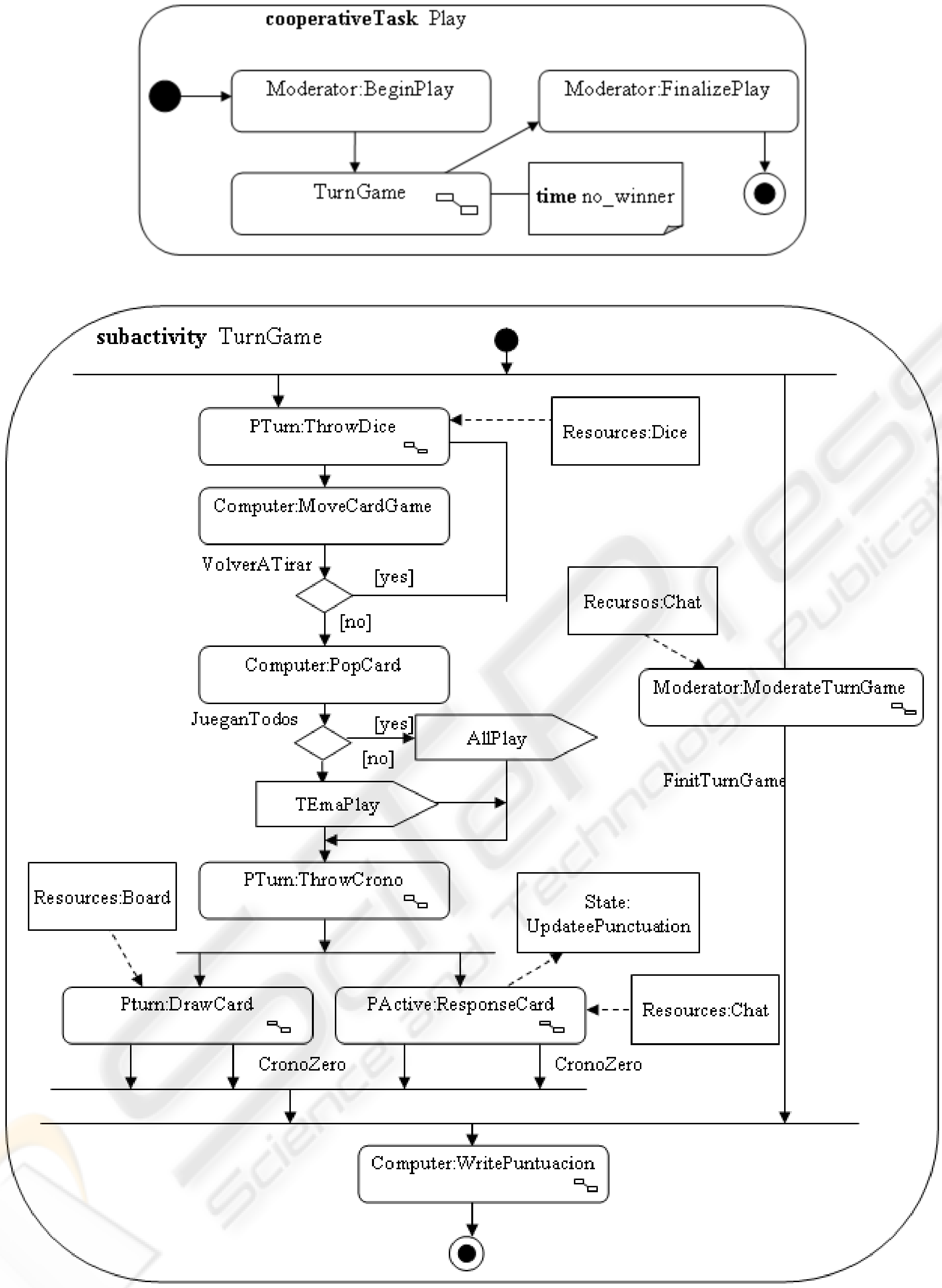

Figure 6: Task Diagram Play and Subactivity Diagram *TurnGame.*

In the previous sections, we have mentioned that the success of a process will depend not only on its logic of application but also on whether the aims associated with each task have been completed. There might be situations where rather than a task being executed on the basis of a formal execution of the tasks comprising its workflow, achievement of the aims and subaims associated with the tasks will be sufficient. Such a situation is typical in collaborative processes where the sequence of actions to be carried out by actors is not clear but the desired aim is.

The links in the hypermedia structure represent the flow between the subactivities of each task taking into account the different laws of application:

- The sequential execution of instances of a same activity is represented by means of a cyclical link from/to the node representing the subactivity (see the node *TurnGame* in Figure 5).
- Initial subactivities in a concurrent flow are represented by means of an arc which joins their respective links with the previous subactivity prior to the concurrent section.
- We therefore indicate that it is necessary to visit all the concepts, i.e. all the concurrent subactivities must be executed (see the node *BeginTurn* in Figure 5).
- For conditional flows, the initial subactivity is the subactivity prior to the condition in the diagram, and the destination subactivities are each possible subactivity after the evaluation of the condition (see the node *PopCard* (origin), and *AllPlay* and *TeamPlay* (destinations)). We use a dashed arc to join the links from the possible origin concept to the destination concept in order to indicate that this concept can be reached from any link, i.e. only an origin subactivity must be executed (see the nodes *ThrowCrono* and *WriteScore)*.

The shaded nodes on the *Task* and *Task-Subactivity* level represent the aims associated to a task (see the concept *SolvedGame* in Figure 5*)*. The navigational links between activity nodes and aim nodes allow us to coordinate activities where the sequence of activities is not clear, and we can navigate through the sequence of activities or the aims. This double path allows a more effective coordination in collaborative systems where uncertainty in task models is an important aspect. The initial nodes for each level are represented by a double ellipse and the links between the nodes at different levels are represented with dashed lines.

If a task has more than one initial subactivity (in the example, the subactivity *TurnGame* is initiated with a concurrent section comprising two subactivities), we must include an extra node (in the example, the node *BeginTurn*) for play to start.

4.3 Resource Representation

The third step in our hypermedia representation is to analyze the use of resources in the system. Next to the task description it is possible to perform a parallel description about the objects used by the different tasks under different roles. In AMENITIES, these objects can be of two types: artifacts, i.e. technological units which support the carrying out of certain collaborative actions - for example the drawing board (see *Resources:Board* in Figure 6); and information objects, i.e. entities with information for carrying out certain actions - for example to register the score (see *State:UpdatePunctuation* in Figure 5). Both concepts appear in the Activity Diagram that describes the tasks and are represented using UML class diagrams.

In Figure 6 we can see how the subactivity *ThrowDice* uses the resource *Dice* in the cooperative task *Play*. The subactivity *DrawCard* uses the resource *Board*, and *ResponseCard* and *ModerateTurnGame* use the resource *Chat* concurrently. For hypermedia resource representation, it is necessary to include on the task/subactivity level a node for each resource and a link to represent the use of that resource by a task. The hypermedia graph on the *Task-Subactivity* level is shown in Figure 7. The shaded nodes represent resources nodes.

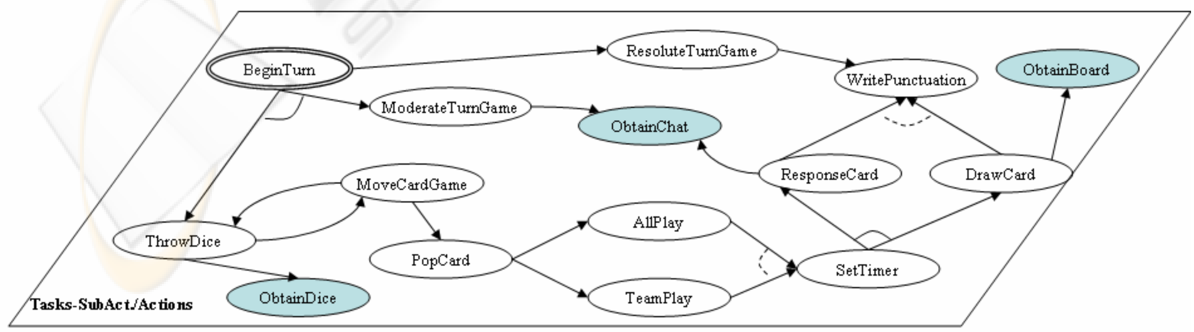

Figure 7: Hypermedia network task *TurnGame.*

5 CONCLUSIONS

In this paper we have presented an approach to represent task models using hypermedia networks. More specifically, we have constructed a hypermedia structure for each of the three fundamental aspects in a collaborative system: the organization's structure (roles), the tasks, and the workflow in each task (subactivities). We have also considered and integrated in these structures two equally important elements: the resources needed to perform the tasks, and the aims and subaims that indicate whether the tasks have been satisfactorily completed in situations where task execution cannot be handled with the simple formal execution of their workflow.

The combination of the structures obtained as a result of this process represents a single global graph which enables us to obtain a dynamic specification that is capable of adapting to changes in context, and thanks to this, we are able to reflect changes in the actors' behavior. It is not necessary therefore for the workflows to be pre-established, but rather the path to be followed at any given time will be determined by the achievement of the aims (conditioned by changes in context).

Our future work will concentrate on the inclusion of the hypermedia network obtained in our web service architecture for the implementation of collaborative systems. Within this architecture, we propose a web service to coordinate workflows which appear in the different existing applications. The hypermedia network obtained will become the information centre which the web service needs in order to carry out all its activities in a flexible way.

Other studies will be focused to analyse the integration of this representation into existing approach such as UML activity diagrams or BPEL specifications.

REFERENCES

Garrido J.L., (2003). *AMENITIES: Una metodología para el desarrollo de sistemas cooperativos basada en modelos de comportamiento y tarea.* Tesis Doctoral. Universidad de Granada, 2003.

Gardner T., (2003). UML modeling of automated business processes with a mapping to BPEL4WS, from http://www.cs.ucl.ac.uk/staff/g.piccinelli/eoows/documents/paper-gardner.pdf

Gutiérrez F.L., Isla J.L., Paderewski P. & Sánchez M., (2006a). Organization Modelling to Support Access Control for Collaborative Systems, In *(IWSSA'06) The 5th international workshop on system/Software architectures, 2006, L*as Vegas, Nevada, USA. 26-29

Gutiérrez F.L., Penichet Víctor M.R., Isla J.L., Montero F., Lozano M.D., Gallud J.A. & Rodríguez M.L., (2006b). Un marco conceptual para el modelado de Sistemas Empresariales. In (Interacción 06) *VII Congreso Internacional de Interacción Persona-Ordenador,*..Universidad de Castilla la Mancha. (2006)

Leymann F & Roller D., (2000). *Production workflow: Concepts and Techniques.* Prentice Hall, NJ, 2000

Leymann F., 2003 Web services: Distributed Applications without Limits. In *Proceedings of the BTW'03*. Leipzig, Germany, February 26–28, GI LNI, 2003

Paterno ,F., (1999). *Model-based Design and Evaluation of Interactive Applications*. F.Paternò, Springer Verlag, ISBN 1-85233-155-0 , 1999

Paterno F., (2001). Task Models in Interactive Software Systems. In *S. K. Chang (ed.), Handbook of Software Engineering & Knowledge Engineering, World Scientific* Publishing, 2001.

W3C, Web Services Definition Language (WSDL) 1.1. http://www.w3.org/TR/wsd.html

OASIS, Universal Description, Discovery and Integration (UDDI).
http://www.oasis-open.org/committees/
uddi-spec/doc/tcspecs.htm#uddiv3

W3C, Simple Object Access Protocol (SOAP) 1.2.
http://www.w3.org/TR/SOAP12

Van der Veer, G.C. & Van Welie M., (2000). Task Based Groupware Design: putting theory into practice. In *Proceedings of DIS 2000*, 17-19 August 2000, New York, United States.

Vernadat F.B., (1996). *Enterprise Modelling and Integration: Principles and Application*, Chapman & Hall, London, 1996