Modelling tasks: a requirements analysis based on attention support services

Joona Laukkanen
+358 (0)44 3651 777
joona.laukkanen@aup.fr

Claudia Roda
+33 (0)65 01 42 75
croda@aup.fr

Inge Molenaar
inge@ontdeknet.nl

ABSTRACT

In this paper we discuss the relevance of task models in the definition of Contextualised Attention Metadata and claim that task models indeed are important and useful in this context. When actions peformed by a user can be interpreted in the context of what the user is doing when that action occurs, the task he is trying to complete, a lot of reasoning can be based on this context. In particular, we will discuss what task oriented services the use of task models makes possible and concentrating on a few of these services, investigate what requirements these services inflict on the task model. Also, we present the work we have done so far on the task model that we have implemented for the Atgentive system.

Categories and Subject Descriptors

General Terms

Keywords

context, attention, contextualized attention metadata, learning, task modelling, intelligence, reasoning

1. INTRODUCTION

Task models represent a very important element in the definition of Contextualised Attention Metadata (CAM). As CAM aims at tracking resources usage, identifying the specific context in which such usage takes place enables a much better understanding of the value of each resource [10]. The task within which a resource is used is a very important element of such contextual definition. For example, in order to truly understand resource usage it would be important to distinguish whether a user accesses a *book review* because he is writing a research paper, or because he is preparing a reading list for a course, or because he is selecting gifts from a wedding list. These three types of usages correspond to access to the resource *book review* in the context of different tasks. The definition of users' tasks is therefore one of the essential elements for the identification of the context of resource usage.

Modelling user tasks in a manner that is both complete and operational is far from being an easy undertaking. Based on the work done in the AtGentive project [2, 16, 17, 19], this paper discusses how task may be modelled in order to support the implementation of attention management services. In the process we will also highlight another important relation between CAM and task modelling, i.e. the fact that not only (as mentioned above) tasks may be associated to resource access, but also resources may be associated to task descriptions.

In the context of the AtGentive system a task represents the target of an attentional focus (e.g. writing a paper, accessing some resource, ...). Since we aim at applicability in combination with a number of different types of applications, the key design issue with the definition of tasks has been to make it as application independent as possible. In particular, the questions of task granularity, task structure, and task attributes, have been addressed.

In section 2 we give a brief description of the AtGentive system. We summarize the goals of the project, introduce the different modules of the system, and explain how the Reasoning Module provides functionalities supporting users' attention allocation. The analysis of such functionalities has provided us with the most critical requirements for the AtGentive task model. Whilst the AtGentive System aims at providing many task-oriented services, in this paper we concentrate only on those that support interruption management and task switching. In section 3 we discuss the requirements that these services impose on task modelling. Section 4 briefly overviews the issues commonly encountered in task modelling, and section 5 details the AtGentive task model.

2. THE ATGENTIVE SYSTEM

The objective of the AtGentive project is to investigate the use of artificial agents for supporting the management of the attention of young or adult learners in the context of individual and collaborative learning environments.

The Atgentive system observes the user's activity and generates *interventions* aimed at supporting his/her attentional choices. Such interventions may either be designed to help users sustaining their current focus of attention (e.g. help user to find the best way to complete a task), or they may be designed to shift the user's attention to a different focus (e.g. communicate important information that has become available).

The main components of an Atgentive system include: (1) one or more applications that communicate with (2) a *reasoning module*, and (3) one or more *user tracking* components providing information about the users activity – see figure 1. Applications, users, and tracking modules inform the reasoning module about the state of the user and the environment by generating *events*. The reasoning module supports the user in his attentional choices

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Conference'04, Month 1–2, 2004, City, State, Country.
Copyright 2004 ACM 1-58113-000-0/00/0004…$5.00.

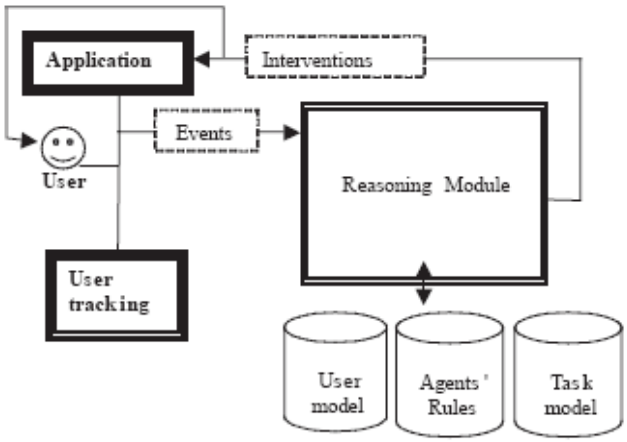

Figure 1 – A simple schema of an Atgentive system

by generating interventions that are then sent to the user through the application.

Events generated by the application either describe the user activity (e.g. the user has started working on a certain task) or relevant changes in the environment sensed by the application (e.g. new information is available for the user).

Events generated by the user may describe his/her preferences (e.g. "don't interrupt me when I am working on this task"), or provide a direct feedback on the reasoning module's interventions.

Finally, The tracking devices monitoring the user physical state and activity may generate events describing the user keyboard activity, the level of noise in the room, or the presence (or absence) of the user from the screen.

On the basis of these events the reasoning module (which is implemented as a multi-agent system) tracks what the users current focus is, then creates a list of possible alternative foci, and finally, evaluates those alternative foci and, using *interventions*, communicates those foci (if any) that seem to be most beneficial to support the user attention.

While processing events, the reasoning module maintains an optimized list of foci that have been identified as most relevant for the user. Each focus is composed of a target, a *priority*, and a *state*. Possible *states* are: *current*, *inactive*, or *suspended*. Normally one of the foci is active (this is the user's current focus). Suspended foci are inactive foci that have been previously active. Inactive foci are those that the reasoning module has evaluated as interesting for the user but the user has never activated (e.g. the focus associated to an email that the user has not yet read). The *priority* is an estimate of how important/urgent is the task associated to the focus for the user. The target of the focus is either a *user task* or a *message*. A user task is an instance of a generic task for the specific user in the specific situation (see section 5.1). A message is something that needs to be communicated to the user without any concrete actions related to it, e.g. some motivational feedback for a learner who has completed an assignment.

Whilst the reasoning module is designed as an application independent, general purpose entity capable of generating suggestions about attention management, within the Atgentive project it is being tested in the framework of two different applications: AtgentSchool, and AtgentNet. AtGentSchool is eLearning platforms for elementary school aged children and AtGentNet is a virtual community platform supporting knowledge exchange in knowledge communities.

3. TASK ORIENTED SERVICS IN ATTENTION AWARE SYSTEMS

In the context of the AtGentive project we have identified several task-oriented services aimed at supporting learners and knowledge workers in environments characterised by frequent interruption and multi-tasking. These include: Interruption management, Support to task switching, Orienteering within resources (e.g. searching and ranking), and Self and Community awareness.

For sake of brevity, in this section we only discuss the first two services with the aim of detecting the characteristics that a task model should have in order to enable the implementation of such services.

3.1. Interruption management

Interruption management services may either automatically select the time and mode of presentation of newly available information, or they may provide notification services that help the user making the decision on when to attend newly available information.

With respect to interruption management, the task model should enable reasoning about cost/benefits of interruptions and allow determining the most appropriate time for interruption.

3.1.1. Enabling reasoning about the cost/benefits of interruptions

In order to decide if to interrupt a user in the first place, the system must be able to consider the costs / benefits of an intervention for the user.

For example, the system could decide to intervene and suggest that the user attends the newly available resource if it can be seen to benefit the user, i.e. if:

[a] the resource is *relevant* to the users current focus or

[b] the resource is relevant to an inactive or suspended task with a high priority

Note that a resource may be relevant to a task both if it is relevant to the task or if it is relevant for a sub-task of that task.

Further considerations may intervene if enough knowledge about the user tasks is available. For example, in case [b] above notification may be delayed if the user is about to complete the current task. Observatory studies report that returning to long term projects, is particularly challenging and makes such tasks potentially more vulnerable to the harmful effects of interruptions, compared to more common, shorter tasks, such as writing e-mails [7]. Besides the expected time that a task on average takes to complete, the number of subtasks within a task (or maybe the number of leaf tasks or suspended leaf tasks) and number of windows and resources that need to be available could well aid at determining if a task is a *long term project* and hence, interrupting it is more costly than interrupting some shorter tasks.

Consequently in order to reason about the cost/benefit of interruptions, the task model should make it possible to:

[REQ 1]identify the user's current task

[REQ 2]identify the priorities of the current task and of other (inactive or suspended) tasks

[REQ 3]identify the resources that may be relevant to a task

[REQ 4]identify the state of advancement in execution of tasks

3.1.2. Determining the most appropriate time for interruption

Several studies have demonstrated that the exact time when an interruption is presented may make a very significant difference on both how easily the information presented is acquired by the user, and on how much disruption it generates in the task being interrupted [1, 6].

In order for the system to determine the most appropriate time for interruption,

[REQ 5]the task model should support the description of task hierarchies.

As noted by Bailey et al. [4] when tasks are organized into hierarchies the task model can be used to infer "breakpoints" i.e. times when interruptions are less disruptive for the user. Bailey and his colleagues [1,5] represent tasks as two level hierarchies composed of coarse events further split into fine events (for example, a coarse event would be the selection of the email application, which would then be further decomposed in selecting the email application, typing in the username, and typing in the password). The authors then measure the impact of interruptions as they occur at various points within these hierarchies and demonstrate that the best times for interruptions correspond to coarse breakpoints. The availability of such a hierarchical task model enables the system to infer the best time for interruption. In the Atgentive system, when there is a switch in the users current task, the magnitude of the break in attention is evaluated on the basis of the depth of the task in the task hierarchy (see below). Further, a shift to the next subtask can be identified as a low strength break in the users attention whilst a jump to a task that is not a child or a parent of the current task may be interpreted as marking a stronger break in the users attention.

If tasks are organized so, that lower level tasks divide a higher level task into logical sub-steps, the level at which a task switch to a next subtask happens could perhaps be used to infer the magnitude of the break in the users attention, possibly with a more accurate value. A switch at a lower level would mark a smaller change in attention than at a higher level. This would however need the task model to allow specifying if a task does indeed refine the parent task or if the parent task exists just to group subtasks, as could be in the case of a math exercise in a learning environment authored as a task hierarchy like [Task T1, “exercise 1”, T1.1, “exercise 1.1”, T1.2, “exercise 1.2”, ..., Task 1.2.n, “3 + 9 / 3 = ”]. In the latter case a task switch to a next subtasks would actually mark a smaller break in user attention on a higher level task than when the switch to the next subtask happens at the level of the concrete leaf tasks with the actual cognitive work.

3.2. Support to task switching

Major motivation for services supporting task switching comes from the observation that people can only focus on one thing at a time and as several authors have indicated [e.g. 18], switching from a task to another is costly. Services supporting the user with task switching operations such as restoring task context, task reminders, and support for *task continuation* could well be introduced if a comprehensive task model exists..

3.2.1. Restoring task context

When task switches and interruptions are frequent, the activities required to restore the task context of a resumed can be expected to result in a significant increase in cognitive load. A diary study tracking the activity of knowledge workers to investigate these effects, reported that participants rated switching to tasks that were previously interrupted to be significantly more difficult than to others, that the resumed tasks were in fact twice as long as other, more short-term projects and that they required significantly more resources than other tasks [7]. Automatically providing access to such resources when a task is resumed would represent a significant help to users. Providing such service requires that:

[REQ 6]The task model associates to interrupted tasks information describing the resources in use when the task was interrupted.

3.2.2. Task reminders

Another problem related to switching tasks is one encountered frequently when a task needs to be performed at a specific moment (at an absolute time or in response to some event). Prospective memory failures, which occur when something cannot be remembered at the right time, may account for up to 70% of the memory failures in everyday life [14]. This has been shown to have a very eminent effect on performance in work and learning environments. Also, these memory failures intervene differently in different age groups.

Providing services that remind users of important dates and deadlines, or notify them of certain events could be used to alleviate this threat of prospective memory failures daunting to so many activities planned to take place in the future. Further, task reminders could prove particularly useful to help users remember tasks that they have suspended earlier as a study has reported that in fact over 40% of tasks that have been interrupted, are not resumed again [15]. For example if a user suspends his current activity and starts to work on another more urgent task, the system could remind him of the task that was interrupted once the task has been completed.

In order to provide support with task reminders, it will be necessary to allow:

[REQ 7]Associating to tasks information either about the time when the task should be executed, or about the events that should trigger the execution (or resumption) of the task

3.2.3. Supporting task continuation and prioritisation

When there are several tasks that the user is working on in parallel or there simply are several tasks to choose from for example when a task has been completed, it could be beneficial for the user if there were services that could take off some of the cognitive load that is related to choosing the next activity. Especially when the user might not have much knowledge of the relevant properties of the different tasks that are available (how long a task is expected to last, for example).

Already using the task structure it is possible to find some potential and logical, yet arguably more or less simple

continuation options for the user. More complex and useful guidance can be achieved by applying some timing strategy in the evaluation, maybe preferring tasks that can expectedly be completed before their deadline. If the evaluation also considers priorities of different tasks or gets otherwise more sophisticated, the reasoning could be expected to have a real effect on the cognitive load of the user.

When a user completes a task, enabling a smooth transition to the next activity may entail restoring the context where the choice to start the now completed task was made. This could amount to reminding the user of the task that was suspended when the user moved to the current task or, reminding him of the current task sequence (e.g. the next subtask, the next required task or the parent of the task in the hierarchy).

In both situations the requirement for a hierarchical task structure ([REQ. 5]) is reinforced.

Further, elements that will intervene in the evaluation of valid continuations include prioritisation (already listed as requirement [REQ. 2]) and timing:

[REQ 8]Task model should allow the definition of task deadlines

On a more sophisticated level also expected duration of tasks, is required, this is listed below as [REQ 14].

4.ISSUES IN TASK MODELLING

Diaper quotes Shepherd [20] as saying that "'Task' is seldom defined satisfactorily" and continues suggesting that this might actually never be the case [8]. Some difficulties in defining tasks, such as the specification of application independent task taxonomies, have been repeatedly encountered and seem inherently difficult. Some other issues may be easier to address but need a comprehensive approach. For example whilst it would not be difficult to provide adequate contextual information for tasks, this information is often missing from task models. This section briefly overviews what we consider the main open issues in task modelling.

4.1. Task taxonomies

One clear problem when modelling tasks is the difficulty of defining a sufficient taxonomy. It would be useful to classify tasks, for example, by type of operation. Finding generic actions or operations independent of application types has however proven very difficult [8]. One of the few generic tasks that Diaper & Johnson [9] were able to identify in their work on TAKD (Task Analysis for Knowledge Description) was *insert*. TAKD is a method capable of modelling tasks in a wide range of applications and within this work *insert* was found common for a number of different objects in different application domains (namely microelectonics, automated office applications and computer programming). Inserting could here mean either inserting text in a word processor or a program editor or alternatively inserting components on a Printed Circuit Board. Whilst it could be possible to identify some actions possibly totally independent of application domains, such as insert, the set of such actions seems to be simply too small. Whilst Diaper [8] does not see the development of task taxonomies as totally impossible, it is obvious that we are far from having such a tool and probably the definition of ontologies allowing the integration of several such taxonomies is the most promising direction of research.

4.2. Task descriptions

Traditionally tasks have been described at the level of the application, i.e. tasks correspond to very specific users' actions within a specific application, e.g. create document, attach document, submit form. In order to support the user in his attentional choices tasks should be described at a level that better corresponds to the user's objectives, (e.g. write a paper, complete an exercise). This type of task description has been suggested by some researchers [11, 13] and corresponds to the one used in the Atgentive project. In order to achieve this objective we require that:

[REQ 9]The task model should allow different types of applications to define their own tasks and task structure

[REQ 10]The task model should allow describing tasks at any level of granularity

4.3. Task attributes

Failing to provide contextual information within a task is another pitfall of several task modelling efforts. Contextual information such as relevant resources and users, deadlines, complexity, priority, state of advancement, and location of the task in a task hierarchy is something that is clearly needed for many services supporting attention management. The inclusion of some of these attributes is represented by several requirements already listed above, further task attributes we have identified include:

[REQ 11]Keywords may be associated to tasks.

Keywords provide a way to relate tasks to resources (e.g. by keyword matching)

[REQ 12]Maximum allowed idle time may be associated to tasks

The Maximum allowed idle time specifies the time limit within which the user is expected to act to avoid being recognised as idle by tracking devices. This information is used both to identify breakpoints and to provide help or solicitations to users who seem to have difficulties continuing a task.

[REQ 13]Task difficulty may be associated to tasks

Indications on the difficulty of a task may help in the evaluation of the cost/benefits of interruptions, as well in the selection of the help to be provided to users.

[REQ 14]Expected duration of the task may be associated to tasks.

This attribute specifies the average expected time to complete the task. Combined to the task advancement indication ([REQ. 4]) enables a better evaluation of the best time for interruption. Further, task continuation services may implement strategies in which, under certain conditions, tasks with certain durations (e.g. tasks that can be completed quickly) are preferred over other tasks.

[REQ 15]Actors relevant to the task may be associated to tasks

Relevant actors could for example include a teacher in the case of a learning environment or the creator of a resource when the task is simply to attend some resource. In general actors relevant to tasks will be defined within a social network associated to the user model. This information is both useful to evaluate the relevance of

newly available information, and to provide community awareness services.

[REQ 16]Support tasks may be associated to tasks (see section 5)

Currently we assume that most of these parameters are manually entered (e.g. by the user himself, or by a teacher setting up a learning sequence - as is done in the AtgentSchool application), in the future we expect that the system may be capable to generate estimates of parameters such as maximum allowed idle time, task difficulty, expected duration time, etc. by observing how several users act on the task, and by inferring the possible behaviour of a specific user.

4.4. Recognising tasks

Whilst defining tasks, their structure and resources presents, as described above, a series of challenges, a further, possibly more complex challenge is represented by the automatic recognition of tasks. This requires that, on the basis of the observation of user's actions, the system is capable of matching actions sequences to specific tasks. The problem here is that if simple sequences of actions are observed (such as typing some characters on the keyboard) the system may not have enough semantic information to associate the action sequence to a specific task. In fact a very large number of higher-level tasks may be associated to simple action sequences. Within Atgentive we base task recognition on three possible inputs. First, an application, which has a much better knowledge of the semantics associated to simple user actions may recognise that the user is working at a specific task and communicate this information to the reasoning module. Second, Atgentive may use its knowledge about a small subset of all possible user tasks that are most likely to be performed by the user at a given time, and use this information to recognise that a simple action sequence is actually contributing to a task. Third, the user may explicitly indicate that he is performing a certain task.

5.ATGENTIVE TASK MODEL

The task model implemented in Atgentive's Reasoning Module distinguishes between two different categories of tasks: main tasks and support tasks. *Main tasks* are in essence anything the user may decide to do. *Support tasks* are aimed at helping the user perform a given main task and manage his attention within that task.

5.1. Generic Tasks versus User tasks

Both *main tasks* and *help tasks* represent abstract task properties. Whenever main tasks, or help tasks are activated concrete instances are created as *user tasks*. This results in creating a hierarchy of *user tasks* corresponding to the hierarchy of the main tasks and support tasks. User tasks instantiate all the properties for the concrete execution of that task, such as a deadline, progression etc, for one particular user. For example, one may have a main task "prepare lecture" which has abstract properties such a *title*, and an average expected duration, and is organised in a hierarchy of sub-(main)-tasks such as "collect resources", "create draft", etc. each having their abstract properties. For each user, there would then be a corresponding user task structure to actually execute the task, with for example individual deadlines for those users.

5.2. Main tasks

Main tasks (and the user tasks that correspond to them) represent actions the user might perform, e.g. write a paper, prepare for a meeting, complete an assignment. These tasks can be formed into hierarchies as pleased as all main tasks could have other main tasks as subtasks.

A main task can then be described to consist of a number of finer level tasks. Task T1, Writing a paper, could for example consist of the more concrete tasks T1.1 (do research) T1.2 (write abstract), ..., T1.n (discuss future work). The hierarchical organization of main tasks allows for varied granularity when defining tasks; nothing forces one to define tasks at a finer level so for example writing a paper could in some environments be modelled as a single high level task if the task is, perhaps, known to be already well understood by the target users. In another environment the same task could be represented as one with a number of subtasks (possibly on several levels). Besides allowing granular description of task execution, subtasking can be used to specify the level at which support needs to be provided for the user. This could in fact be one way to author tasks; first identify how the entire task needs to be supported (e.g. support for doing research, support for writing the abstract, ...) and divide the task in subtasks accordingly.

In defining task structure we have identified further requirements for the task model these include:

[REQ 17]The task model may include a requirement level for a task

[REQ 18]The task model may include task ordering

[REQ 19]The task model may include task visibility

These properties are tightly related to the execution of tasks at a given moment and are useful to support *task continuation* (see 3.2.3) and are briefly described below.

Task requirement level

Task may be defined as *optional* or *required*. Required sub-tasks are necessary (i.e. they must be executed) for the completion of the parent task. Tasks defined as optional allow the user to skip certain sub-tasks in the execution of a main task. In a learning environment some exercise for example, may be marked as optional.

Task ordering

The order in which a task's subtasks need to be performed could either be specified as free for the user to choose, or mandated. In a learning environment an assignment might for example consist of reading a book and then writing a summary about it. Here it would make sense to mark the ordering of the assignments subtasks to be mandated.

Note that, if ordered execution is required, optional subtasks can still be skipped.

In the current implementation going backwards in the execution, even to an optional task, is currently not possible.

Task visibility

Tasks may either be visible or invisible. Invisible tasks are always inner nodes in the task hierarchy and allow describing abstract tasks that, although not executable, are useful to conceptualise a certain grouping in sub-tasks. A group of root tasks that are not

related to each other could for example be grouped under one common invisible root task. Invisible tasks could also be useful if there is a need for a more complex ordering than what otherwise would be allowed by the task model (without adding mundane tasks that only include selections between subtasks).

The task model does not support certain task sequencing constraints. For example its is not currently possible to specify the requirement that the user completes a certain number of subtasks (say 2 tasks out of 3).

5.3. Support tasks

Support tasks are tasks aimed at supporting the user in performing various types of activities that the user might attend at different stages of a tasks execution. For example, a support task might help a confused user gaining a better understanding of the task at hand, another support task could provide some motivational feedback such as statistical information about the users time usage after a task is already completed.

Support tasks differ from main tasks mainly in two ways. First, they cannot be organized into hierarchies and they do not have further support tasks themselves. Although hierarchies of support tasks might be a valid concept, we don't identify a pressing need for them and do not include the concept in the model. This helps us also avoid introducing unnecessary complexity to the model. Essentially for the same reason we don't consider the concept of support tasks for support tasks. The other key difference to main tasks is the classification of support tasks. Support tasks are classified in two ways. First they are classified based on when they will be relevant to the task that they support. This could either be before (pre-task support), during (on-task support) or after the task (post-task support). In addition, support tasks are classified by the type of support they provide.

Based on scaffolding models of Hadwin, Wozney & Pontin, Zimmerman & Chrunk, Azevedo & Hadwin and Wood, Bruner, and Ross [3, 12, 21, 22] we have divided support tasks into four categories: behaviour, cognitive, metacognitive and motivational support tasks. Cognitive interventions have a focus on mental activities of the user, metacognitive support is directed at orientating, monitoring and evaluation activities, behavioural interventions are focussed on physical activities of the user and motivational support tasks are directed towards feelings of the user [12]. The term scaffolding was introduced by Wood, Bruner, and Ross [22] and it is defined as providing assistance to a student on as-needed basis, fading the assistance as the competence increases. The general idea behind scaffolding is that some of the control within the learning environment is temporally transferred from the learner to another more experienced actor to support the learner to acquire all abilities to fully self sustain his learning. The scaffold help supporting the execution of a task that the student could not have done on its own and it is removed when it is no longer necessary. Especially in innovative learning arrangement where student are provided with more control of both learning content and learning procedures scaffolds can support them to deal with this increased responsibility. The task support model allows specifying and selecting the support tasks that assiste the learning process of specific learners based on an assesment of their attentional states.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have identified the major requirements for a flexible and operational task model supporting the implementation of attention management services. We have also indicated how most of these requirements have been implemented within the Atgentive system. We consider the work presented in this paper only a starting point for attention oriented task modelling and the definitions provided will need to be both extended and further detailed. We are currently in the process of evaluating the performance of the Atgentive system in the two pilot environment and we trust that such evaluation will significantly contribute to the further development of the reasoning module as a whole and of the task model in particular. While the task model we present does not have the same objectives as many of the task models presented in the field of human-computer interaction, some of them may be used to guide future development of our model.

7. REFERENCES

[1] Adamczyk, P. D. and B. P. Bailey, 2004. If not now, when? The effects of interruption at different moments within task execution. Human Factors in Computing Systems: CHI'04, New York, ACM Press.

[2] Atgentive (2005-2007). *ist-4-027529-stp*.

[3] Azevado, R., Hadwin, A.F. (2005) Scaffolding self-regulated learning and metacognition – Implications for the design of computer-based scaffolds. Instructional Science33: 367–379

[4] Bailey, B. P., P. Adamczyk, et al. (2006). "A Framework for Specifying and Monitoring User Tasks." Computers in Human Behavior **22**(4): *709-732*.

[5] Bailey, B. P. and J. A. Konstan (2006). "On the Need for Attention Aware Systems: Measuring the Effects of Interruption on Task - Performance, Error Rate, and Affective State." Computers in Human Behavior **22**(4): *685-708*.

[6] Czerwinski, M., E. Cutrell, and E. Horvitz. Instant messaging: Effects of relevance and time. in HCI 2000 - 14th British HCI group Annual Conference. 2000. University of Sunderland: British Computer Society.

[7] Czerwinski, M., E. Horvitz, et al. (2004). A diary study of task switching and interruptions. Proceedings of the SIGCHI conference on Human factors in computing systems, Vienna, Austria, ACM Press.

[8] Diaper, D. (2002). "Scenarios and task analysis." Interacting with Computers **14**(4): 379-395.

[9] Diaper, D. and Johnson, P., 1989. Task analysis and systems analysis for knowledge descriptions: theory and application in training. In : Long, J., Whitfield, A. (Eds.). Cognitive Ergonomics and Human-Computer Interaction. Cambridge University Press, Cambridge, pp. 191-224.

[10] Duval, E. (2005). LearnRank: the Real Quality Measure for Learning Materials (Tematic Dossier 06 December 2005), Insight - Oservatory for New Technologies and Education.]

[11] Gonzalez, V. M. and G. Mark (2004). "Constant, constant, multi-tasking craziness": managing multiple working spheres. Proceedings of the SIGCHI conference on Human factors in computing systems, Vienna, Austria, ACM Press.

[12] Hadwin, A., L. Wozney, et al. (2005). "Scaffolding the Appropriation of Self-regulatory Activity: A Socio-cultural Analysis of Changes in Teacher–student Discourse about a Graduate Research Portfolio." Instructional Science 33(5-6): 413-450.

[13] Heath, T., M. Dzbor, et al. (2005). Supporting User Tasks and Context: Challenges for Semantic Web Research. Workshop on End-user Aspects of the Semantic Web (UserSWeb), European Semantic Web Conference (ESWC2005), Heraklion, Crete.

[14] Kvavilashvili, L., D. J. Messer, et al. (2001). "Prospective memory in children: The effects of age and task interruption." *Developmental Psychology* **37**(3): 418-430.

[15] O'Conaill, B. and D. Frohlich (1995). Timespace in the Workplace: Dealing with Interruptions. CHI '95 Conference Companion, Denver, Colorado, United States, ACM press.

[16] Roda, C., Ed. (2006). Atgentive (IST-4-027529-STP) Deliverable D1.3 - Atgentive conceptual framework and application scenarios.

[17] Roda, C. and T. Nabeth (2006). The AtGentive project: Attentive Agents for Collaborative Learners. First European Conference on Technology Enhanced Learning EC-TEL'06, Crete, Greece, Springer.

[18] Rubinstein, J. S., D. E. Meyer, et al. (2001). "Executive Control of Cognitive Processes in Task Switching." Journal of Experimental Psychology: Human Perception and Performance **27**(4): 763-797.

[19] Rudman, P. and M. Zajicek (2006). Autonomous agent as helper – Helpful or Annoying? IAT 2006 - IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Hong Kong.

[20] Shepherd, A., 1998. HTA as a framework for task analysis. Ergonomics 41 (11), 1537-1552.

[21] Zimmerman, B., Schunk, D. (2001). Theories of self-regulated learning and academic achievement: an overview and analysis. In self-regulated learning and academic achievement (2nd ed.) (pp. 1-37). Mahwah, NJ; Erlbaum.

[22] Wood, D., Bruner, J., & Ross, G. (1976). The role of tutoring in problem solving. Journal of Child Psychology and Psychiatry 17, 89-100.