

Adaptive Work-Centered and Human-Aware Support Agents for Augmented Cognition in Tactical Environments

Martijn Neef¹, Peter-Paul van Maanen^{1,2}, Peter Petiet¹ and Maartje Spoelstra¹

¹ TNO Defence, Security and Safety, The Netherlands

² Vrije Universiteit, Amsterdam, The Netherlands

{martijn.neef, peter-paul.vanmaanen, peter.petiet, maartje.spoelstra}@tno.nl

Abstract. We introduce a support system concept that offers both work-centered and human-aware support for operators in tactical command and control environments. The support system augments the cognitive capabilities of the operator by offering instant, personalized task and work support. The operator obtains support by entering into a collaborative agreement with support agents. Such an agreement creates opportunities to establish adaptive capabilities and human-aware support features. We describe the concept in and propose an experimental design to evaluate its effectiveness in tactical environments.

Introduction

Tactical command and control environments present demanding challenges to operators because of their task complexity, time-criticality and the risk associated with poor decisions. Operators will have even more difficulty in handling their tasks with the increasing complexity of defense missions and the ongoing pressure to reduce manning in defense environments. Operators will continue to face new challenges when it comes to properly focusing attention, keeping correct situation awareness and preventing overload situations. Conventional technology-driven support systems are not capable and adaptive enough to fulfill such support demands, and there is a distinct need for support systems that cope with those demands.

Support systems for tactical environments come in many flavors, but many recent developments show support systems that resemble active companions rather than conventional decision aids. The key feature of such companions is adaptivity. Adaptive capabilities for support systems come in two varieties: (a) adaptivity based on the behavior and performance of the human operator, and (b) adaptivity based on the state of the environment. The first variety could be called *human-aware* adaptivity, because it requires the support system to have a certain degree of awareness about the behavior and motivations of the human operator. The second variety could be called *work-centered* adaptivity, because it revolves about events in the environment, and thus mainly requires knowledge about the work domain. Both types of adaptivity result in changes in the interaction between the human operator and his support systems. This includes every option from a change in notification method to a change in the division of labor between operator and support system. We believe that a support system should include both types of adaptivity, and be readily configurable, so that the operator can benefit from adaptive capabilities at the right moment, and in the right form.

Based on previous work on human aware agents (Maanen, 2008), adaptive autonomy (Van de Vecht, 2007), and work-centered support systems (Scott, 2005), we propose an extended support system that specifically targets the attention allocation problem and task overload issue of tactical officers by combining both human-aware and work-centered adaptivity. Maanen (2008) describes a generic adaptive human-aware mechanism for assisting humans in appropriately allocating attention. This mechanism has been tailored and evaluated in the domains of air traffic control (ATC), computer games, and in tactical picture compilation (TPC). In this paper, we extend this mechanism with the possibility of adjusting the agreements between the human operator and the support agent. The system gives the officer options to arrange and configure support capabilities at will. These support options are embodied by autonomous software agents that act as versatile companions, but only after explicit creation and instruction by the operator. It is the responsibility of the human user to instantiate support agents, and to tune them to the task at hand. By making it the responsibility of the operator to create his own support functions, we lessen the possibility of ‘automation surprises’ (Sarter, 1997), and make the operator better aware of his system surroundings, goals, and states. Configuring and managing adaptivity is an important issue when dealing

with support systems that have adaptive capabilities. Adaptive features make it harder for an operator to anticipate on system behavior and thus may cause serious usability issues. Moreover, more intensive training and instruction may be necessary in order to fully control and anticipate on the behavior of the adaptive support system.

In this paper, we will first briefly describe adaptive autonomy in software agents, work-centered support systems, and previous work on human-aware support. Next, we propose a set of design requirements for adaptive, work-centered and human-aware support agents, and discuss their use in practice. In addition, we propose an experimental setting for validation and evaluation of the concept. We conclude with a discussion and thoughts on future work.

Work-centered support systems

Work-centered support system design uses methods from cognitive system engineering (cognitive work and task analysis, work-centered design) to understand and support operators in complex work environments. It adds agent technology to create practical support tools that help to reduce work complexity and keep the user in control of his work (Scott, 2005). Traditional user interface design is primarily focused on the capabilities of the system, and forms the interface so that system features are as accessible as possible. Work-centered support system design, on the other hand, seeks to shape the working environment in such a way that it agrees with the user's work, and make relationships between elements, constraints and affordances in the environment easier to observe. Support systems built on this premise are context sensitive, easily adaptable and tuned to the antics and ontologies of the human way of working.

A good example of a work-centered support system is the work by Eggleston on support agents for weather forecasters in a military airlift organization (Scott, 2002). Their design allows operators to create agents instantly for various types of work support. For example, operators can use visual signs to instruct watch agents to monitor certain events on the interface, such as the formation of thunderstorms. Other types of agents in the system monitor external information sources and manage the presentation of information to the user. By letting the operator self-initiate support agents and providing a high level of observability and directability of automation behavior, there is less chance of automation surprises, which makes the system easier to accept.

We use principles from work-centered system design for our support model. More specifically, we want our support agents to help focus the operator on important issues in this work, and, in line with work-centered design principles, make them able to support both work-domain tasks and meta-cognitive activities.

Human-aware support systems

In previous work, a generic human attention-based task allocation (HABTA) agent component has been described and instantiated in a tactical command and control environment (Van Maanen, 2008). An agent incorporating a HABTA-component is aware of the operator's attentional state (human-aware) and is capable of managing the operator's attention and that of the agent itself. When the complexity of the in- and external environment increases, so will the information volumes for navigation, system monitoring, and tactical tasks (Grootjen, 2006). With expanding information volumes, human management of attention will eventually lead to performance degradation. The idea is that implemented HABTA-components help in decreasing this effect by preventing human error. This can be done through notification or by taking over certain tasks from the human at the appropriate time. Attention management will have a significant positive impact in situations (a) where there is a need for frequent attentional switches between tasks or objects, (b) where novice operators have difficulty selecting the appropriate attentional focus, or (c) where an inappropriate attentional focus may lead to severe operational risks.

A HABTA-component is based on two cognitive models: one that describes the current allocation of a human's attention and one that prescribes the way his attention should be allocated. If there is a discrepancy between the outputs of both models, HABTA reallocates the tasks between the human and the agent, for instance depending on certain rules the human and agent agreed upon. Models of attention or situation awareness have already been developed and used to predict faults in attention allocation (e.g., the

SEEV model (Wickens, 2004), but less is known about how they can be used to initiate agent adaptation, or, more specifically, automatic task reallocation.

The HABTA approach was evaluated in air traffic control (ATC), computer games, and in tactical picture compilation (TPC). Figure 1 shows an example of a tactical environment that is suitable for the application of HABTA.

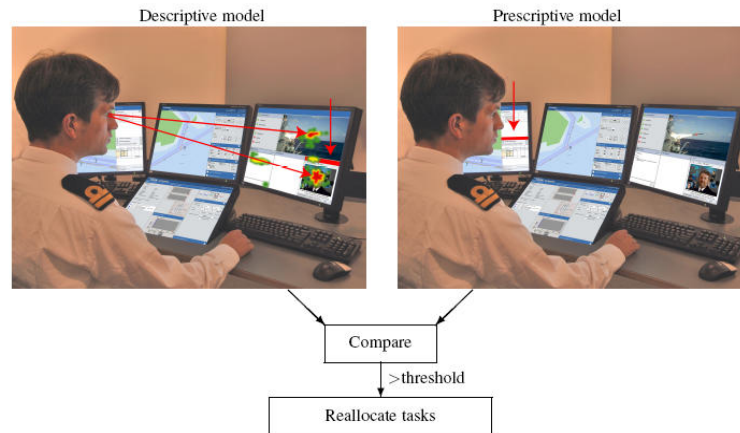


Fig. 1. Design overview of the HABTA-component in a tactical environment (Van Maanen, 2008)

Human-aware support agents augment the meta-cognitive capabilities of human operators. For instance, TPC is a critical task in naval surface warfare, and its main goal is to create a correct tactical picture of the direct surroundings of a ship at sea. In such a dynamic and complex environment, the human operator is likely unable to get an accurate overview of the whole situation without support. A HABTA-based agent presented in (Van Maanen, 2008) is able to monitor the complete environment, and when the agent decides that human attention is needed within a specific area, it draws the attention of the human operator. In other words, it has a high level of human-aware adaptivity (directs human attention), but a low level of work-centered adaptivity (does not perform the task itself).

In the framework presented here, we combine the human-aware features of such HABTA-based agents and the influence control that allows for work-centered autonomy. For this purpose, we extend the HABTA-based agent by the introduction of the tasking description and interaction contracts, and a reasoning component for influence control.

Adaptive Autonomy

Adaptivity is an often-sought feature of support systems in many domains. Most recent work in this field use the intelligent agents paradigm to model adaptive capabilities, mainly because intelligent agents self-governing by definition. Autonomy is a defining characteristic of intelligent agents (Jennings, 2000) and it usually interpreted as the amount of freedom an agent has from intervention by other agents, including humans (Barber, 2001). Another characteristic of artificial agents is their goal-driven behavior, which directly relates autonomy to agent decision making. The degree of autonomy of decision making can be defined as the degree of intervention by other agents on the decision making process of one agent (Barber, 2001, Castelfranchi, 1995). An agent heavily influenced in its decision making by other agents displays obedient behavior. An agent that does not allow any external influence in its decision making is maximally independent. By altering the amount of external influences on its decision making, an agent can self-adapt its autonomy, and effectively display adaptive autonomy. In this fashion, agents can actively select the level of autonomy that best fits the circumstances and their own objectives.

Van der Vecht (2007) developed a model that implements this perception of autonomy by means of reasoning rules that decide on adoption or rejection of certain influences. These reasoning rules weigh knowledge from the agent's internal state against incoming information, and decide whether external events may affect the agent's internal state, and consequently the decision making process.

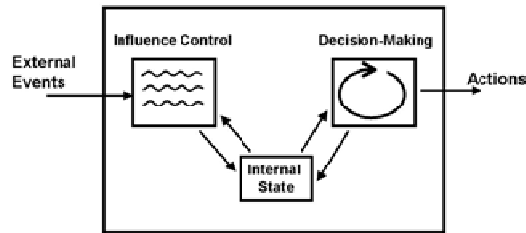


Fig. 2. Illustration how influence control can affect agent behaviour (Van der Vecht, 2007).

By giving the agent self-control over these rules, the agent can filter out certain events and communications and consequently adjust his own level of autonomy. Van der Vecht illustrates his model by giving example rule sets for organizational influences (how the agent should deal with influences from other actors), for information (how to determine information relevance) and environmental knowledge (how to let the availability of resources and actors influence autonomy).

The approach of Van der Vecht to adaptive autonomy offers interesting perspectives for use in support system design, because it provides a comprehensive way to represent the rules that produce adaptive behavior. It also returns the responsibility for adaptive behavior itself to the agent, while giving the operator straightforward means to convey preferences (i.e. by manipulating the influence control settings of the support agent so that the boundaries of the autonomy of the agent become clear). Van der Vecht uses social contracts, as proposed in the OperA framework (Dignum, 2004), to convey organizational knowledge into the influence control rules. Upon entering an organization, the agent signs a contract that specifies its role and hierarchical relations to other actors. This contract directly translates to reasoning rules for the influence control. For example, messages from commanding officers should always be allowed to pass. In a similar manner, we propose to use interaction contracts between operator and support agent to specify adaptive behaviors in a support setting.

Combining work-centered and human-aware adaptivity

We combine elements from work-centered design, human-aware support systems and adaptive autonomy to form a novel approach to tactical C2 support system design. Our basic stance is that the operator should be responsible for creating his own set of support agents. The operator creates agents by means of tasking descriptions and interaction contracts. These instructions give the agents a basis for their behavior. Because the operator actively creates an agent for a specific task, and explicitly communicates his expectations about the agent's behavior, it will be easier for the operator to remain in control over his suite of support tools.

There are two types of agents that the operator can create, namely work-centered agents and human-aware agents. Work-centered agents deal with events in the work domain, such as the monitoring of a certain geographical map for dangerous events. Human-aware agents deal with the meta-cognitive aspects of the operator, such as workload management and attention strategies. Most human-aware tasks will require a normative model for assessing the operators' state or actions. Therefore, the use of human-aware task agents will often give rise to the use of work-centered agents. Jointly, human-aware task agents and work-centered task agents make it possible to create comparative support features, such as human attention assessment. Figure 3 shows the interactions between user and agents.

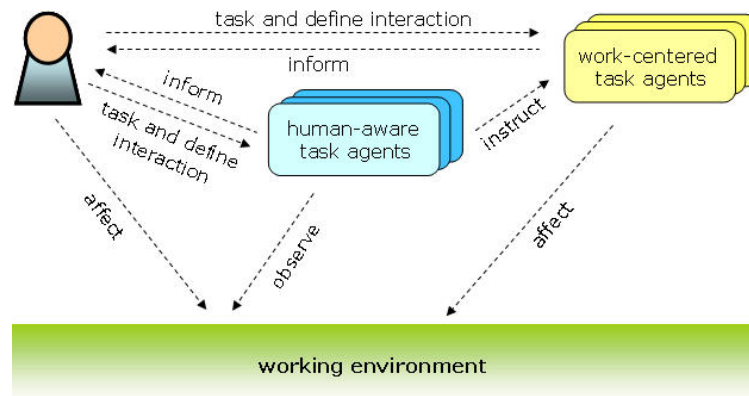


Fig. 3. Interactions between the human operator, human-aware agents and work-centered agents.

Creating support agents

The creation of support agents consists of three steps for the operator: *defining the support task*, *defining the interaction*, and *instantiating the agent* (see Figure 4). In the first step, the operator expresses a functional task description for the support agent. This description includes essential aspects such as objectives, start- and stop-criteria, and pre- and post-conditions. How the operator actually communicates this tasking description is of less importance, as long as the process uses visualizations and terms from the work-domain itself. Tactical C2 officers usually have a map-based work environment, and most of their tasks have geographical attributes ('monitor this area', 'follow this track'). It would make sense to use a tasking method that uses objects and terms from that environment, which seamlessly integrates with the operators' own way of working. For example, for a work-centered agent, the operator could mark out an area using his input device, and make a selection from a predefined set of typical support tasks (e.g. 'monitor this area for suspect tracks', see Table 1). This would be a straightforward manner for the operator to request help, without having to leave his work domain to enter a 'programming process'. On the side of the agent, the visual tasking would be translated in a set of actions and pre- and post-conditions.

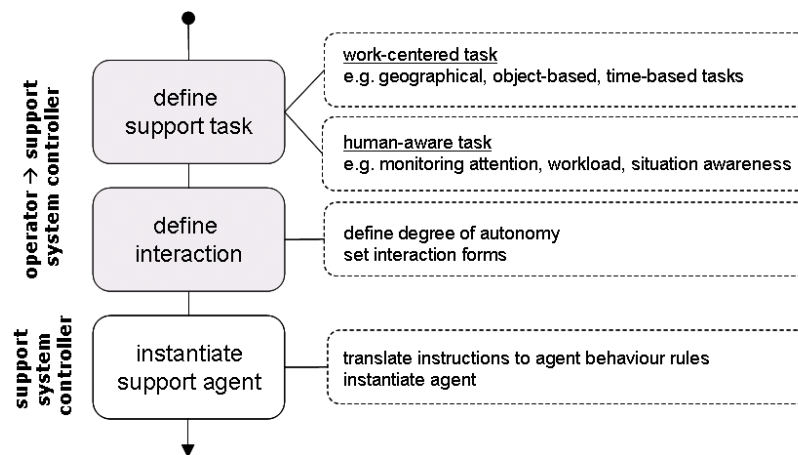


Fig. 4. Definition and instantiation of support agents.

The second part of the agent creation process deals with defining the interaction between operator and support agent by means of an interaction contract. The interaction contract defines the desired behavior of the agent towards the operator. It delimits the extent of its autonomy (in how far the agent can take control

of its own behavior), it defines communication requirements towards the operator (what should the agent communicate and under which conditions), and sets collaborative aspects (what is the division of labor between agent and operator). It defines the degree of autonomy that the agent is granted, and states in how far, and under which conditions the agents may perform its task independently from the operator.

Tasking description (work-centered task)		
task	Monitor (area) for contacts	
type	work-centered	
agent	WCA_1	
subtasks	MONITOR (area) CLASSIFY (object) REACT (object) INFORM (actor, form, status)	observe area classify object activate countermeasures provide information
rules	MONITOR (area) for contacts IF status(contact) = new, THEN CLASSIFY (contact) IF status(contact) = suspect THEN REACT (contact)	
preconditions	main: IF operator permission THEN do (main_task) for all subtasks: interpret interaction contract determine permission and information requirement	

Tasking description (human-aware task)		
task	Attention support (area)	
type	human-aware	
agent	HAA_1	
subtasks	OBSERVE (actor) COMPARE (actor1, actor2) RESPOND ASSIGN (objects, actors) INFORM (actor, form, status)	observe actions of actor compare actions of actors actions in case of disagreement allocate actions to actor provide information (warnings)
rules	O1 = OBSERVE (operator) O2 = OBSERVE (WCA_1) IF difference (O1, O2) > threshold THEN INFORM (operator, form, status) ASSIGN (contacts, WCA_1)	
preconditions	main: IF operator permission THEN do (main_task) for all subtasks: interpret interaction contract determine conditions and information requirements	

Table 1. Examples of tasking description for a work-centered en human-aware tasks in pseudo code.

The interaction contract describes per task, when it is permitted. For example, if the agent has been instructed to monitor an area and act upon certain events, then the interaction contract may specify whether the operator needs to be consulted before acting. In this case the contract can contain conditions related to the agent's awareness of the user's task execution of current attention division and based on these conditions determines either to interrupt the user, wait until a more appropriate moment, or execute his task anyway. In the tasking description, this actually sets the pre-condition.

Interaction description	
parties	operator, WCA_1
task	Monitor (area) for contacts
rules	relevant trigger: workload MONITOR: allow(agent, MONITOR) CLASSIFY: allow(agent, CLASSIFY) IF status(contact) = suspect THEN require(agent, INFORM (suspect), warning) REACT: conditional IF workload (operator) > threshold2 THEN allowed(agent, REACT) required(agent, INFORM (action), notification) IF workload (operator) > threshold1 THEN required(agent, INFORM,(warning), warning)

Table 2. Example of an interaction contract in pseudo code. The contract relates to the first example in table 1.

The support agents use a reasoning model to decide on their behavior. In this model (Van der Vecht, 2007) there is a distinction between 'influence control' and 'decision making' (see figure 2). The autonomy of an

agent is determined by the amount of influence that other actors have on the agent’s decision making. The influence-control function actively regulates which events and commands reach the decision-making process. It filters out those external events that should not influence the decision making, and thus it indirectly controls the agent’s behavior. Similarly, the influence-control function can adapt the agent’s behavior by altering its settings. For example, if the agent should take the workload of the operator into account in its decision making process (human-awareness), the influence control could adjust its settings so that cues pertaining to the state of the operator are admitted.

<p>“Whenever status-suspect then agent is obliged to do inform-operator-about-status”: <code>observation(status-suspect) ← TRUE AddGoal(send(operator, inform, new-suspect-contact))</code></p> <p>“Whenever workload-high then agent is permitted to do propose-operator-contact-takeover”: <code>observation(workload-high) ← TRUE AddBelief(permitted(propose-operator-contact-takeover))</code></p> <p>“Whenever takeover-request from operator then agent is obliged to do accept-request”: <code>message(operator, request, raise-support-level) ← TRUE AddGoal(takeover(Task))</code></p>

Table 3. Three examples of reasoning rules for setting influence control.

Table 3 show examples of rules that govern the influence control. If the former part of the rule (the head) of the rule agrees with the condition (the guard), then the latter part (the body) is performed. Adding a ‘goal’ boils down to obliging the agent to perform a certain action. Adding a ‘belief’ gives the agent a notion of permissible actions. By giving the agents simple rules like these, the operator can communicate his preferences, and easily change agent behavior.

Experimental validation

There are many open issues to be answered when putting work-centered and human-aware agents into practice. Especially the design of controls that allow an operator to convey preferences to support agents will require further attention. In this section, we describe an experimental design for a first-order validation of the concept. We draw inspiration from earlier work on active support systems such as the TANDEM experiments in the TADMUS project (Sycara, 2004), and the aforementioned studies on HABTA-based agents (Van Maanen, 2008), and propose an experimental setting in which we evaluate the operator performance under different saturation conditions. When stressed enough, the attention of the operator will become more and more saturated and will need support to achieve his objectives. We propose to subject the operator to several saturation conditions using three dimensions of the problem space: complexity, time-criticality, and volume. We use multiple dimensions, because each dimension calls for a different way for working, and we need to ensure that the operator benefits from his support agents under all circumstances.

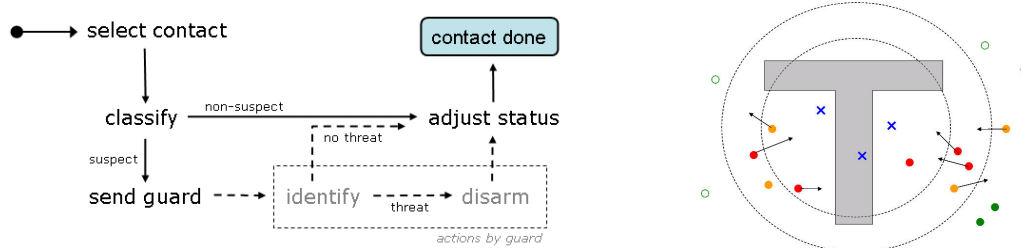


Fig. 5. Left: Task sequence in the experiment. Right: Basic scenario environment layout.

We propose a common tactical scenario: the protection of a High Value Target (HVT). The HVT is an airport, threatened by hostile entities. The airport is defended by mobile guards, who can intercept individuals that pose a security threat. A commander with global overview of the HVT area instructs the guards. Sensor networks around the HVT reveal movements of individuals, but cannot reveal whether their intent is malicious. If the commander has reasons to believe that a certain individual poses a threat to the airport, then he can instruct guards to identify, and, if necessary, disarm that individual. The challenge for

the operator is to keep track of all contacts on the screen, while commanding his guards in such a way that they do not miss out on any threats to the airport. This implies two vital decisions on the part of the commander: (a) deciding whether a contact is suspect or not, and (b) deciding whether guards should intercept the contact. Figure 5 shows the sequence of tasks. On the right side of figure 5, we can see a schematic game view of the T-shaped airport, with three guards present at the junction of the runways. Surrounding the runways are several entities at different distances from the airport, indicated by rings around the airport. One way to classify entities around the airport is by using this distance: entities within the inner circle could then be classified as “hostile”, the ones between the circles as “possibly hostile”, and the entities outside the outer circle as “neutral”.

The operator will have a limited number of options to create support agents. Work-centered agents can do the exact same tasks as the operator (*observe*, *select contact*, *classify*, *send guard*, *adjust status*), but the operator decides which tasks the agent may perform. The operator can opt to have a work-centered support agent to help monitor a certain area (*observe* and *classify*), or assist by also actually instructing guards (*observe*, *classify* and *send guard*). This selection equals setting the degree of autonomy. In addition, the operator can include a limited number of human-aware features, such as *attention monitoring* (based on comparing classification assessments between operator and a work-centered agent) and *workload management* (based on the time it takes for a contact to be classified by the operator). Furthermore, the tasking interface will give the operator a number of ways to divide the labor between himself and the agents. The most reasonable options would be via spatial arrangements (sectors, perimeters or free-form areas that the operator can draw on the screen), based on perceived threat (the operator deals with all suspect contacts, the agent is responsible for monitoring unknown or safe contacts).

By varying three scenario parameters, we can challenge the operator, and evaluate the behavior of the support agents. By increasing the number of contacts on the screen (*volume*), or increasing the movement speed of contacts (*time*), we can drive the operator into a work overload situation. We can also make the classification task harder (*complexity*) by letting contacts follow paths that are more diverse. A contact that takes a long detour along the area before turning towards the airport will be harder to assess than a contact that follows a straight line towards the airport. Another option to increase complexity would be to include more non-threatening entities. Varying settings of these parameters will likely lead to a different use of the support system, and thus give an interesting insight in the value of such a support system in tactical environments. For instance, a higher level of complexity could possibly lead to more reliance on human-aware support, while the operator will most likely benefit more from work-centered agents under higher volume circumstances. Exploring the range of use scenarios is essential given the many different types of missions in tactical command and control.

We expect that this experimental setting will reveal critical usability and efficiency indicators, and give us a better understanding of the relationship between scenario-type and most favorable type of support, in terms of adaptive capabilities and user interaction.

Summary and conclusion

In this paper, we have introduced the approach of combining work-centered and human-aware adaptivity in support agents for operators working in tactical command and control environments. The support agents augment the cognitive capabilities of the human operator by providing task support adapted to the operator’s need for assistance, a management task formerly done by the operator himself. This notion of human-awareness is extended with the possibility to adjust work agreements between operators and agents in order to result in less automation surprises. This entails the use of a work-centered component of the agent, able to take over (parts of) the task of the human operator. We described the basic aspects of the approach, and proposed an experimental design to evaluate its value in practice. We can summarize the approach in the following set of design requirements:

- Support functions are embodied by autonomous software agents.
- It should be possible for the operator to create a support agent for a specific task at his will. This includes work-centered tasks (tasks that relate to the domain), and human-aware tasks (tasks that relate to the behavior of the operator), or the combination thereof.

- It should be possible for the operator to observe and configure the behavior of the support agents, including the limits of its autonomy and specific interaction requirements via suitable controls.
- The support agent should have a suitable normative model of user behavior and (part of) the work domain. These models may be available by design, or given to the support agent by the operator through simple instructions (for instance, performance indicators, geographical cues).
- Human-aware agents must be able to instruct existing work-centered agents, or create such agents themselves. This allows a human-aware agent to autonomously reallocate tasks from the human operator user to the support suite in the case of suboptimal performance by the operator.
- Work-centered agents instantiated by a human-aware agent should behave according to the interaction contract set between the operator and the human-aware agent.

The main objective of the suggested approach is to augment cognition by providing assistance in meta-cognitive processes (human-aware) and lessening decision making complexity by offering work-centered task support. We expect that this strategy is going to be very effective in domains where there is limited time and resources for the execution of multiple tasks, but where the operating environment provides high volumes of information. The separation in work-centered and human-aware task agents makes it easier to develop interesting adaptive support features, such as the three examples below.

<p>The operator is in discussion with a colleague, and loses but there are too many tasks to be completed. A support agent watches the environment. Seeing that some tasks need urgent attention, the agent decides to take over tasks within his permission range, and notifies the operator to refocus his attention (<i>attention- and priority based adaptivity</i>)</p> <p>There is an incoming message, which the operator does not respond to. A support agent has been set up to monitor for relevant messages. The agent interprets the message as relevant with respect to active tasks and alerts the operator (<i>information relevance</i>)</p> <p>The operator has communicated a list of tasks to a support agent for progress monitoring. The next task needs to be finished shortly. The current set of activities leads the agent to believe that the operator will fail his upcoming task deadline if no action is undertaken. The agent generates a cue to make the operator aware of this impending issue. When the operator fails to show any signs of action, then the agent will activate a work-centered agent to take on this task. (<i>time-based and workload-based adaptivity</i>)</p>

Table 4. Three sample situation where human-aware and work-centered agents work in unison to support the operator.

This approach is the first step towards a framework that can cater both technical and operational demands, and is open enough to accommodate all sorts of cognitive requirements.

We believe that our approach is novel and worthwhile, because it joins several established lines of thought on support system design. It augments cognition by providing assistance in meta-cognitive processes (human-aware) and lessening decision making complexity by offering work-centered task support. By keeping a distinction between human-aware tasks and work-centered tasks, it becomes easier to design tailored, adaptive support. Clear interaction contracts between operator and support agents define the boundaries of agent behavior, and minimize the chances on automation surprises. The practice of interaction contracts agrees with most cognitive system engineering proposition, and help to maintain observability and directability of system functions (Klein, 2004). By giving the operator agent tasking controls that use visualizations and ontologies from his work-domain, user acceptance and benefit will easier to achieve.

The approach needs practical evaluation, and this will most likely reveal many implications. We may inadvertently introduce many new issues. By effectively shifting the agent design process from designer to user, we lessen many design challenges that stem from user modeling. At the same time, this implies that the user becomes partly responsible for system management as well. The management of a large number of autonomous support agents will require extra attention and take up valuable work capacity. The use of contracts to instantly create support agents is a fascinating concept, but it will require an extensive exploration into user tasking controls, interaction formats and management procedures.

Acknowledgements

The research reported here is funded by the Dutch Ministry of Economic Affairs (Interactive Collaborative Information Systems (ICIS) project, grant nr. BSIK03024) and by the Dutch Ministry of Defense (Programme number V524).

References

- Barber, K.S., Martin, C.E. (2001), *Dynamic adaptive autonomy in multi-agent systems: Representation and justification*. International Journal of Pattern Recognition and Artificial Intelligence 15(3), pp. 405-433.
- Castelfranchi, C. (1995), *Guarantees for autonomy in cognitive agent architecture*. In Wooldridge, M.J., Jennings, N.R. (eds.): Intelligent Agents: Proceedings of the First International Workshop on Agent Theories, Architectures and Languages (ATAL-94), Springer, Berlin, pp. 56-70.
- Grootjen, M., Neerincx, M.A., Weert, J.C.M. van (2006), *Task-based interpretation of operator state information for adaptive support*. Foundations of Augmented Cognition, 2nd Edition, Strategic Analysis, Arlington, Virginia, pp. 236-242.
- Jennings, N.R. (2000), *On agent-based software engineering*. Artificial Intelligence 117(2), pp. 277-296.
- Klein, G., Woods, D.D., Bradshaw, J.M., Hoffman, R.R. and Feltovich, P.J. (2004), *Ten Challenges for Making Automation a "Team Player" in Joint Human-Agent Activity*. IEEE Intelligent Systems, 19(6), pp. 91-95.
- Maanen, P.-P. van, Koning, L. de, Dongen, K. van (2008), *Design and Validation of HABTA: Human Attention-Based Task Allocator*. In: Proceedings of the First International Workshop on Human Aspects in Ambient Intelligence, Published in: M. Mühlhäuser, A. Ferscha, and E. Aitenbichler (eds.), Constructing Ambient Intelligence: AmI-07 Workshops Proceedings, Communications in Computer and Information Science (CCIS), vol. 11, Springer Verlag, pp. 286-300.
- Parasuraman, R., T. B. Sheridan, et al. (2000), *A Model for Types and Levels of Human Interaction with Automation*. IEEE Transactions on Systems, Man, and Cybernetics, 30(3), pp. 286-297.
- Sarter, N.B., Woods, D.D., and Billings, C.E. (1997). *Automation Surprises*. In G. Salvendy (Ed.), Handbook of Human Factors and Ergonomics (2nd edition), New York, NY: Wiley, pp. 1926-1943.
- Scott, R., Deutsch, S., Kazmierczak, T., Kuper, S., Roth, E., Malchiodi, E., Eggleston, R., and Whitaker, R. (2002). *Using Software Agents in a Work Centered Support System for Weather Forecasting and Monitoring*. Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting (Baltimore MD), Santa Monica CA: Human Factors and Ergonomics Society.
- Scott, R., Roth, E. M., Deutsch, S. E., Malchiodi, E., Kazmierczak, T., Eggleston, R. Kuper, S. R. and Whitaker, R. (2005), *Work-Centered Support Systems: A Human-Centered Approach to Intelligent System Design*. IEEE Intelligent Systems, vol. 20(2), pp. 73-81.
- Sycara, K. and Lewis, M. (2004). *Integrating intelligent agents into human teams*. In E. Salas and S. Fiore (Eds.), Team Cognition: Understanding the Factors that Drive Process and Performance, Washington, DC: American Psychological Association, pp. 203-232.
- Vecht, B. van der, Dignum, F., Meyer, J.-J. Ch., and Neef, M. (2007), *A Dynamic Coordination Mechanism Using Adjustable Autonomy*. In J.S. Sichman, J. Padget, S. Ossowski and P. Noriega (Eds.): Coordination, Organization, Institutions and Norms in Agent Systems III. Lecture Notes in Computer Science (LNCS), Vol. 4870, Springer Verlag, pp. 83-96.
- Wickens, C. D., McCarley, J. S. (2001). *Attention-situation awareness (A-SA) model of pilot error*. Final Technical Report ARL-01-13/NASA-01-6). Savoy, IL: University of Illinois, Aviation Research Lab.