Towards Commitments for Situated Agents*

Danny Weyns, Elke Steegmans, Tom Holvoet
AgentWise, DistiNet, Department of Computer Science
K.U.Leuven, B-3001 Leuven, Belgium
{danny.weyns, elke.steegmans, tom.holvoet}@cs.kuleuven.ac.be

Abstract – *Traditional architectures for situated, behavior-based agents take the viewpoint of the individual agent to select the most appropriate action. Action selection is typically based on internal stimuli and stimuli from the agent's neighboring environment. As such collaborations between agents have to emerge from the individually selected actions of the agents.*

In this paper we study the research problem of how to enable explicit collaborations between situated agents. Explicit collaborations are reflected in mutual commitments. Contrary to the traditional approaches of commitment that are based on the mutually dependent mental states of the involved agents and a goal-oriented plan, we introduce the notion of a situated commitment that is based on the roles of the involved agents and the local context they are placed in. Activating mutual situated commitments in a collaboration results in more consistent behavior of the agents towards their commitments. The proposed approach fits the general principles of situatedness and robustness of situated multi-agent systems.

Keywords: Situated multi-agent systems, roles, situated commitments, role-based collaboration.

1 Introduction

Dealing with the increasing complexity of developing, integrating and managing distributed applications is a continuous challenge for software engineers. It is generally acknowledged that self-management is the only viable long-term solution to challenge this complexity. In our research we propose situated multi-agent systems as a generic approach to develop self-managing distributed applications.

A situated multi-agent system¹ (situated MAS) models a problem domain as a collection of agents and objects situated in an environment. In a situated MAS, agents and the environment constitute complementary parts of a multi-agent world that can mutually affect each other. Situatedness places an agent in a context in which it is able to perceive its environment and in which it can (inter)act. Intelligence in a situated MAS originates from the interactions of

the agents in their environment rather than from the capabilities of individual agents. Where the approach of situated MASs started from the rejection of classical agency based on symbolic AI, nowadays the original opposition tends to evolve towards convergence with different schools emphasizing different aspects. In recent years, several researchers have shown that a mutual integration of visions can yield a synergetic surplus value, see e.g. [5][11][20]. This paper contributes in this evolution by integrating commitments in situated, behavior-based MASs.

Traditional architectures for situated agents (see e.g. [2][9][14][1][3]) take the viewpoint of the individual agent to select the most appropriate action. Action selection is typically based on internal stimuli and stimuli from the agent's neighboring environment. Situated agents usually communicate indirectly, e.g. by depositing pheromone trails in the environment. As such collaborations between agents have to emerge from the individually selected actions of the agents.

In this paper we study the research problem of how to enable explicit collaborations between situated agents. Explicit collaborations are reflected in mutual commitments. Contrary to the traditional approaches of commitment that are based on the mutually dependent mental states of the involved agents and a goal-oriented plan, we introduce the notion of a *situated commitment* that is based on the *roles* of the involved agents and the *local context* they are placed in. An example of a situated commitment is an agreement between two robot agents to transfer a stack of pallets. After agreeing, the first robot passes the pallets one by one to the second robot who accepts the pallets and constructs a new stack with them. Both robots play a complementary role in this collaboration. The collaboration continues until the stack is transferred, unless one of the involved robots for some urgent reason withdraws from the collaboration, e.g. to recharge its battery. Then the conditions for the situated commitment no longer hold and that finishes the collaboration.

This paper is structured as follows. In section 2, we describe the Packet-World, i.e. a simple multi-agent application we use as an illustrative case in the paper. Section 3 gives a brief overview of our agent architecture. In section 4, we zoom in on free-flow architectures, i.e. the decision

*0-7803-8566-7/04/\$20.00 © 2004 IEEE.

¹Alternative designations are behavior-based agents [2], adaptive autonomous agents [9] or hysteretic agents [6][5].

Figure 1: The Packet-World

structure for an agent to select actions. Section 5 discusses in detail how we have integrated roles and situated commitments in free-flow architectures. We illustrate the extended model with an example of collaborating agents in the Packet-World. Finally we conclude in section 6.

2 The Packet-World

Throughout the paper we use the Packet-World [15] as an illustrative case. Figure 1 depicts an example of the Packet-World. The goal of the agents in the Packet-World is to bring colored packets (squares) to correspondingly colored destinations (circles). Agents can observe the environment, however only to a limited extent. Agents are allowed to make one step at a time to a free neighboring field or pick up a packet from a neighboring field. An agent is able to carry one packet at a time, which it can put down at any free neighboring field or at the destination. Agents can also send messages to each other either for exchanging information or for setting up collaborations. Performing actions requires energy. Therefore agents are equipped with a battery. Energy is of vital importance for the agents. The battery can be charged at one of the available charge stations. To find the way to a charge station, each charge station transmits a gradient field. Agents can follow such a field and “climb up” the gradient towards the station. In the example of Figure 1 there is one charge station which is indicated by an electric socket symbol. The gradient field is marked by concentric rounded squares around the charge station. The intensity decreases proportional to the distance to the charge station.

3 Generic agent architecture

In this section we give a brief overview of the agent architecture we have developed during the last two years of our research. The agent architecture fits in a generic architectural model for situated MASs we have described in [16]. This generic model formally describes an abstract architecture for situated MASs. The model builds upon Ferber’s theory for action, described in [5]. In essence, this theory separates what an agent wants to perform from what actually happens.

Agents produce influences into the environment and subsequently the environment reacts by combining the influences to deduce a new state of the world from them. The reification of actions as influences enables the environment to combine simultaneously performed activity in the MAS.

An overview of the generic architecture with a focus on the functional decomposition of an agent’s behavior is depicted in Figure 2. We touch briefly on the different modules. The *Perception_i* module takes care of the perception of the environment of agent a_i , i.e. it maps the local state of the environment σ onto a percept, denoted as p_i . The *Consumption_i* module selects from the set of consumptions χ a consumption c_i for agent a_i . A consumption is an effect of the reaction of the environment to the most recently produced influences for a particular agent. When an agent “consumes” a consumption, the consumed effect can be absorbed (e.g. food that is turned into energy), the agent may simply hold an element (e.g. an object it has picked up) or the consumption may affect the agent’s state (e.g. the arm of a robot is wrenched through an external force). Perception and consumption depend on an agent’s ontology [18].

The *KnowledgeIntegration_i* module uses the most recent percept and consumption to update the internally registered state and to produce the *current knowledge* κ_i . The current knowledge is determined by the actual context the agent is situated in. Contrary to knowledge-based agents, situated agents do not build up an internal model of the environment. Instead, they favor to employ the environment itself as a source of information. Internal representations are described in terms of the immediate environment, known as indexical-functional representation [10][5]. Situated agents use representations to direct their decision making process, however this is done “here and now”. Such representations do not oblige the agent to keep track of a hypothetical future state or investigate the implications of it on a plan.

The *Decision_i* module is responsible for action selection. We developed the decision module as a free-flow activity tree. The tree consists of nodes that collect activity from internal or external stimuli and feed it down in the tree. At the lowest level, action nodes receive the activity. The action node that collects the most activity is selected for execution. Free-flow architectures are discussed in section 4. Since existing free-flow architectures lack explicit support for social behavior, we introduced the concepts of a role and a situated commitment. A role is defined as a subtree that covers a logical functionality of the agent. A commitment is defined as a relationship between one role (the goal role) and a non-empty set of other roles (the source roles) of the agent. Activating a situated commitment results in more consistent behavior of the agent towards the commitment. We elaborate on the concepts of a role and a situated commitment in section 5. To decide about its next action, the decision module takes the agent’s current knowledge κ_i , the set of available roles ρ_i and the activated situated commitments γ_i and selects an operator o_i for execution. *Exec_i* executes the selected operator producing an influence f_i into the environ-

Figure 2: Generic model for a situated agent

ment. The environment collects the influences of simultaneously acting agents [17] and calculates the reaction, i.e. state changes in the environment and consumptions for the acting agents, according to a set of domain specific laws.

The final building block in the model of a situated agent is the *Communication_i* module. The communication module processes incoming messages and produces outgoing messages according to well-defined communication protocols. Agents typically modify their state (current knowledge and/or situated commitments) implied by the communicative interactions. For a detailed study of the communication module, we refer to [19]. Messages are exchanged through the communication medium that is part of the environment.

4 Free-flow architectures

Open multi-agent systems are characterized by dynamism and change: new agents may join the system, others may leave, the environment may change, e.g. its topology or its characteristics such as throughput and visibility. To cope with such dynamism the agents must be able to adapt their behavior according to the changing circumstances. A well-known family of agent architectures for adaptive behaviour are free-flow architectures.

Free-flow architectures are first proposed by Rosenblatt and Payton in [12]. In his Ph.D thesis, T. Tyrrell [14] demonstrated that hierarchical free-flow architectures are superior to flat decision structures, especially in complex and dynamic environments. The results of Tyrrell's work have been very influential, for a recent discussion see [3].

A simple example of a free-flow architecture is depicted in Figure 4. The hierarchy is composed of *nodes* which receive information from internal and external stimuli in the form of *activity*. The nodes feed their activity down through the hierarchy until they arrive at the *action nodes* where a winner-takes-it-all process decides which action is selected. The main advantages of free-flow architectures are:

- Stimuli can be added to the relevant nodes avoiding the “sensory bottleneck” problem.
- Decisions are made only at the level of the action nodes; as such all information available to the agent is taken

into account to select actions.

- Since all information is processed in parallel the agent can take different preferences into consideration simultaneously. E.g. consider an agent that moves to a spotted object but is faced with a neighboring treat. If the agent is only able to take into account one preference at a time it will move straight to the spotted object or move away from the threat. With a free-flow decision tree the agent avoids the treat *while* it keeps moving towards the desired object, i.e. the agent likely moves around the treat towards a spotted object.

Figure 4 depicts a free-flow tree for action selection of a simple agent in the Packet-World. The agent collects packets and brings them to a destination. The agent is supplied with a battery that provides energy to work. The agent has to maintain its battery, i.e. when the energy level of the battery falls below a critical value the agent has to recharge the battery at a charge station. The left part of the depicted tree represents the functionality for the agent to search, collect and deliver objects. On the right, functionality to maintain the battery is depicted. The *System node* feeds its activity to the *Work node* and the *Maintain node*. As a default, the full amount of activity is passed to each child node. However, when a link is marked with a number, a corresponding proportional part of the activity of the parent node is fed to the child node (see e.g. the link between the *Work node* and the *Explore node*). The *Work node* combines the received activity with the activity from the *energy level* stimulus. The “+” symbol indicates that the received activity is added up. The negative activity of the *energy level* stimulus indicates that little energy remains for the agent. As such the resulting activity in the *Work node* is almost zero. The *Maintain node* on the other hand combines the activity of the *System node* with the positive activity of the *energy need* stimulus, resulting in a strong positive activity. This activity is fed to the *ToStation node* and the *Charging nodes*. The *ToStation node* combines the received activity with the activity level of the *not at station* stimulus (the “*” symbol indicates they are multiplied). In a similar way the *Charging node* combines the received activity with the activity level of the *at station*

Figure 3: An example of a free-flow architecture

stimulus. This latter is a binary stimulus, i.e. when the agent is at the charge station its value is positive, otherwise it is negative. The *ToStation* node feeds its positive activity towards the action nodes it is connected to. Each moving direction receives an amount of activity proportional to the value of the *gradient* stimulus for that particular direction. *gradient* is a multi-directional stimulus. The value of this stimulus (for each direction) is based on the sensed value of the gradient field that is transmitted by the charge station. In a similar way, the *Charging* node and the child nodes of the *Work* node (*Explore*, *Collect* and *Deliver*) feed their activity to the action nodes they are connected to. Action nodes that receive activity from different nodes combine that activity according to a specific function. The action nodes for moving actions use a function f_m to calculate the final activity level. A possibility definition of this function is the following:

$$A_{moveD} = \max [(A_{Node} + A_{stimulusD}) * A_{freeD}]$$

Herein is A_{moveD} the activity collected by a move action, D denotes one of the four possible directions, i.e. $D \in \{N, E, S, W\}$. A_{Node} denotes the activity received from a node. The move actions are connected to four nodes: $Node \in \{Explore, Collect, Deliver, ToStation\}$. With each node a particular *stimulus* is associated. $stimulus \in \{random\ direction, see\ object, see\ destinat, gradient\}$ are multi-directional stimuli with a corresponding value for each moving direction. Finally, *free* is a multi-directional binary stimulus that indicates whether the way to a particular direction is free for the agent to move to.

When all action nodes have collected their activity the node with the highest activity level is selected for execution. In the example, the *ToStation* node is clearly dominant

over the other nodes connected to actions nodes. Currently the East and West directions are blocked (see the *free* stimulus), leaving the agent two possibilities to move towards the charge station: via North or via South. In the depicted situation, the agent will move northwards according to the values of a guiding gradient field.

Evaluation of the example brings up the following considerations. (1) Although the behavior of the agent in the example is rather simple, the free-flow tree is already fairly complex. For a non-trivial agent, the overall view of the tree quickly becomes very cluttered. When a change is made in one part of such a tree it becomes unclear how this affects the other parts. Although free-flow trees are at best developed with a focus on a particular functionality of the agent, the architecture in itself does not support any structure. From our experiences we learned that it is unrealistic to assume that software engineers build a complex free-flow architecture for complex applications, where agents can perform many actions. For such applications, the architecture quickly becomes unmanageable, it is no longer possible to have an overall view of the architecture. (2) The free-flow architecture does not support explicit social behavior. Actions are selected based on internal stimuli (e.g. the energy level of the battery) or external stimuli (e.g. the perception of a packet). In the subsequent section, we introduce the notions of a role and a situated commitment as a means to structure the behavior of an agent on the one hand, and to enable explicit social behavior between agents on the other hand.

5 Roles and situated commitments

Explicit collaborations between agents are reflected in mutual commitments. The attitude of a commitment has been studied extensively, however always from the perspective of

Figure 4: Extended free-flow architecture with a detail of the activity flow

cognitive agents, see e.g. [4][7][8]. These traditional approaches take a *psychological* viewpoint on commitment, i.e. a commitment is essentially based on the mutual beliefs of the involved agents. We introduce the notion of a *situated commitment* as a *social* attitude of situated agents. Contrary to the traditional approaches on commitment which are based on the mutually dependent mental states of the involved agents and a goal-oriented plan, a situated commitment is based on the *roles* of the involved agents and the *local context* they are placed in. We share the sociological viewpoint on commitment proposed in [13], however Singh’s work focuses on cognitive agents in information rich environments while our focus is on situated, behavior-based agents.

As indicated in Figure 2, roles (ρ_i) and situated commitments (γ_i) are related to action selection as well as to communication. In this paper we study the concepts of a role and a situated commitment from the viewpoint of action selection, which is the responsibility of the $Decision_i$ module. [19] elaborates on the impact of roles and situated commitments in the agent’s communication.

Figure 4 shows a free-flow tree for agents in the Packet-World. We have omitted the details of stimuli etc. Contrary to the example discussed in the previous section, the free-flow tree depicted in Figure 4 allows an agent to collaborate explicitly with another agent by forming a chain to pass packets to one another.

We define a role as a subtree in the hierarchy that covers a logical functionality of the agent. The root node of such a subtree is denoted as the *top node* of the role. A role is named as its top node. A role may consist of a set of sub-roles, and sub-roles of sub-sub-roles etc. All roles of the agent are constantly active and contribute to the final decision making by feeding subsets of actions with activity. However the contribution of each role depends on the activity it has accumulated from the affecting stimuli of its nodes. In the example, there are three roles demarcated by dotted triangles. In the role *Individual*, the agent searches for packets and brings them to the destination. The role of *Chain* is composed of two

sub-roles: *Head* and *Tail* denoting the two roles of agents in a collaboration to form a chain to pass packets to each other. Finally in the role of *Maintain* the agent recharges its battery.

A situated commitment defines a relationship between one role, called the *goal* role, and a non-empty set of other roles, i.e. the *source* roles, of an agent. Situated commitments have a well-known name. Explicitly naming the commitments enables agents to set up mutual commitments in a collaboration [13]. However, a single agent can also commit to itself. The connector *Charging* in Figure 4(a) denotes the situated commitment of an agent to itself to recharge its battery. The connectors *HeadOfChain* and *TailOfChain* denote the mutual situated commitments of two agents to collaborate in a chain. With each link between (the top node of) a source role and the commitment, a *weight factor* is associated that determines the amount of influence of the source role on the goal role via the situated commitment.

Besides a name, each situated commitment is characterized by a *relations* set, a *context*, an *activation condition*, a *deactivation condition*, a *status* (activated or deactivated) and an *addition function*. The relations set contains the identity of the related agent(s) in the situated commitment. The context describes contextual properties of the situated commitment such as descriptions of objects in the local environment. Activation and deactivation conditions can be boolean expressions based on internal state or perceived information, or event occurrences such as sending or receiving a message. When the activation condition becomes true, the situated commitment is activated. The situated commitment then injects an additional amount of activity in the goal role defined by the addition function. The weight factors of the links from the source roles determine the fraction of activity of each source role that is taken into account by the addition function. The top node of the goal role combines the additional activity of the situated commitment with the regular activity accumulated from its stimuli. As soon as the deactivation condition becomes true, the situated commitment is deactivated. Then the situated commitment no longer influ-

Figure 5: Detail of the free-flow architecture of Figure 4

ences the activity level of its goal role.

In general one agent can be involved in different situated commitments at the same time. Activity received through different situated commitments is then combined with the regular activity received from stimuli into one result.

As an example, consider agent 8 (see Figure 1) that commits to be *HeadOfChain* in a collaboration to pass packets to agent 1, see Figure 5. The activation condition for the situated commitment is the receipt of the confirmation message from agent 1 to collaborate, indicated as *request for Tail accepted*. The relation set and the context of the commitment are specified as $\langle \{1\}, \text{gray packets} \rangle$, indicating that the relations set for the commitment is the singleton agent 1, and the context of the situated commitment are *gray packets* that are passed along the chain. The deactivation condition is specified as a change in the environment that indicates that the collaboration has finished, in particular *nb of remaining packets = 0* or *chain broken*. *nb of remaining packets = 0* indicates that all packets have been passed, while *chain broken* indicates that the collaborating agent has left its post. In the example, the agent currently does not carry a packet, and, as its next action, will pick up the packet located on its East side.

Summarizing, agents agree on mutual situated commitments in a collaboration via direct communication. Activating situated commitments and initiating their properties (relations set and context) are typical responsibilities of the *Communication_i* module. Once activated, the situated commitment will affect the selection of actions in the *Decision_i* module. The situated commitment induces extra activity in the hierarchy, favoring the goal role of the commitment relatively to the source roles. Contrary to traditional approaches of commitment (e.g. a joint commitment [4]) where

the agents have the obligation to mutually communicate with each other when the conditions for a committed cooperation no longer hold, for a situated commitment it is typically the local context in which the involved agents are placed that regulates the duration of the commitment. This approach fits the general principles of situatedness and robustness of situated MASs.

6 Conclusion and future work

In this paper we extended free-flow architectures with the concepts of roles and situated commitments. A role is defined as a subtree in the free-flow hierarchy that covers a logical functionality of the agent. A situated commitment defines a relationship between one role, called the goal role, and a non-empty set of other roles, i.e. the source roles, of an agent. On the one hand, roles and situated commitments enable a designer to structure the agent's behavior, improving the design of free-flow architectures. On the other hand, roles and situated commitments enable explicit collaborations between situated agents. The followed approach fits the general principles of situatedness and robustness of situated multi-agent systems.

To validate the extended free-flow architecture, we implemented it in the Packet-World. In the paper, we showed how agents in the Packet-World collaborate to pass packets along a chain. The next step is to verify the model in a more complex case. In a current industrial research project we investigate how the multi-agent paradigm can be applied to the control of Automated Guided Vehicle (AGV) warehouse systems. Traditional systems use one central controller that instructs the AGVs to perform jobs based on a calculated plan. By looking at AGVs as agents of a situated MAS the system

can control itself in a distributed manner, improving adaptability and scalability. This real-world application is well suited to verify our model for situated agents in general, and the extended free-flow architecture in particular.

References

- [1] E. Bonabeau, F. Henaux, S. Guerin, D. Sneyers, P. Kuntz and G. Theraulaz, "Routing in Telecommunications Networks with Ant-Like Agents", IATA, 1998.
- [2] R. A. Brooks, "Intelligence Without Representation", *Artificial Intelligence Journal*, (47), 1991.
- [3] J. Bryson, "Intelligence by Design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agents", *PhD thesis*, MIT, 2001.
- [4] P. Cohen and H. Levesque, "Teamwork", *Cognitive Science and Artificial Intelligence*, 1991.
- [5] J. Ferber, *Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, Great Britain, 1999.
- [6] M. R. Genesereth and N. Nilsson, "Logical Foundations of Artificial Intelligence", Morgan Kaufmann, San Mateo, CA, 1987.
- [7] N. Jennings, "Commitments and Conventions: The Foundation of Coordination in Multi-agent Systems", *The Knowledge Engineering Review*, 2(3), 1993.
- [8] B. Dunin-Keplicz and R. Verbrugge, "Calibrating collective commitments", 3th Central and Eastern European Conference on Multi-Agent Systems, CEEMAS'03, Prague, LNAI vol. 2691, 2003.
- [9] P. Maes, "Modeling Adaptive Autonomous Agents", *Artificial Life Journal*, 1(1-2), MIT, 1994.
- [10] P. Maes, "Designing Autonomous Agents" MIT Press, 1990.
- [11] V. Parunak and F. Zambonelli, "From Design to Intention: Signs of a Revolution", 1th International Conference on Autonomous Agents and Multi-Agent Systems, Bologna, AAMAS, 2002.
- [12] K. Rosenblatt and D. Payton, "A fine grained alternative to the subsumption architecture for mobile robot control", International Joint Conference on Neural Networks, IEEE, 1989.
- [13] M. Singh, "Commitments Among Autonomous Agents in Information-Rich Environments", 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Sweden, 1997.
- [14] T. Tyrrell, "Computational Mechanisms for Action Selection", *Ph.D thesis*, University of Edinburgh, 1993.
- [15] D. Weyns and T. Holvoet, "The Packet-World", Demo presented at the 1th International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS, Bologna, 2002.
- [16] D. Weyns and T. Holvoet, "A Formal Model for Situated Multi-agent Systems", *Formal Approaches for Multi-Agent Systems, Special Issue of Fundamenta Informaticae*, to appear, 2004.
- [17] D. Weyns and T. Holvoet, "Regional Synchronization for Simultaneous Actions in Situated Multi-agent Systems", 3th Central and Eastern European Conference on Multi-Agent Systems, CEEMAS'03, Prague, LNAI vol. 2691, 2003.
- [18] D. Weyns and E. Steegmans and T. Holvoet, "Towards Active Perception in Situated Multi-agent Systems", *Special Issue of Journal on Applied Artificial Intelligence*, 18(8-9), 2004.
- [19] D. Weyns and E. Steegmans and T. Holvoet, "Protocol-based Communication for Situated Multi-agent Systems", 3th International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS, New York, 2004.
- [20] F. Zambonelli and N. Jennings and M. Wooldridge, "Developing multiagent systems: The Gaia Methodology", *Transactions on Software Engineering and Methodology*, 12(3), ACM, 2003.