

Adaptive Scheduling in Wireless Sensor Networks

A.G. Ruzzelli, M.J. O’Grady[†], and G.M.P. O’Hare, R. Tynan
{ruzzelli, gregory.OHare, richard.tynan}@ucd.ie

Adaptive Information Cluster (AIC),

[†]Practice & Research in Intelligent Systems & Media (PRISM) Laboratory,
michael.j.ograde@ucd.ie

Department of Computer Science, University College Dublin (UCD),
Belfield, Dublin 4,
Ireland.

Abstract. As the number of Wireless Sensor Networks (WSNs) applications is anticipated to grow substantially in coming years, new and radical strategies for effectively managing such networks will be needed. One possibility involves endowing the network with an autonomic capability to dynamically adapt itself to the prevailing network operating conditions, even while communications sessions are active. This may involve the network adapting itself either partially or completely. The approach suggested in this paper proposes that a suite of intelligent agents autonomously monitor the various network nodes and, depending on the status of certain parameters, actively intervene to alter the scheduling mechanism used, thus ensuring continuous operation and stability of the network together with an improved performance yield.

1 Introduction

Wireless Sensor Networks (WSNs) technology consists of a large number of small electro-mechanic devices (in the order of hundreds or thousands) called sensor nodes and a low number of gateways or sinks (in the order of units or tens). Sensor nodes are of low-cost and of low-memory capacity; their task is to collect data from the surrounding and to relay them to the gateway in multi-hop fashion. Gateways are usually considered more powerful and they act as medium between the network and the user. A sensor network is that it should remain active, functional and unattended for long period of time. Hence the need for a strategy to minimize the energy consumption of the nodes. MAC and Routing protocols have a big impact on the overall performance of the system in terms of energy saving. As suggested from the ISO/OSI architecture, network protocols are normally treated separately. Such an approach helps to reduce the descriptive and organizational complexity and is very useful in terms of solving problems independently and autonomously. However, a difficulty with this approach is the increased computational overhead of managing the system that can

lead to an increase both in latency of messages from source to destination, as well as overload on the low memory of the sensor node processor. Ruzzelli et al. [1], proposed the lightweight MERLIN protocol to integrate MAC and Routing into one simple architecture. MERLIN has been designed to reduce the energy consumption of nodes by trading off its intrinsic low-latency properties. In this paper, the ability of the network to adapt in response to node failures or application requirements is studied. In particular, the solution proposed encompasses the use of mobile intelligent agents as a basis for realizing dynamic adaptivity in situations of unexpected or erratic network behavior. In such cases, an appropriate opportunistic scheduling mechanism will be adopted to reduce the energy consumption according to the prevailing circumstances. The paper is structured as follows: Section 2 provides a further reflection on the need for adaptivity and includes a review of other work in this area. In Section 3, a description of MERLIN, an energy efficient MAC and routing integrated protocol for WSNs is described. The use of intelligent agents for realizing dynamic adaptivity is discussed in Section 4.

2 Motivation

As the deployment of WSNs increases, the number of applications dependent on their reliable operation likewise increases. Thus the need for a strategy for handling the irregularities and anomalies that will undoubtedly arise. As an example, consider the case of a network of sensors that notify a variation of temperature above a certain threshold. In case of fire, all nodes in the vicinity will continuously send messages, leading to possible network overload as well as increasing the energy consumption. Furthermore, another application may be required to effect a dynamic change in the sampling rate, depending on the frequency of incoming measurements. To address such scenarios and ensure that the WSN continues to operate in an optimum manner, it is proposed that the network be initially subdivided into virtual sectors for node localization. Using a lookup table, intelligent mobile agents can migrate to the relevant sector, or indeed migrate to as many nodes as necessary to ascertain the status of the network, and then proceed to identify and take the necessary corrective action. Although the literature offers many relative energy-efficient MAC protocols [2–5], and routing protocols [6, 7], only few protocols are focused on the integration of different layers and functionalities as in [8]. Moreover, TDMA protocols like TRAMA [3] or EMACS [9] suffer from a very high latency of messages that makes them very slow to adapt to changes in the network.

Traditionally, there has been a strong research focus on the use of mobile agents in the telecommunications area. As well as network management [10], the areas of resource allocation and management have successfully demonstrated the potential of agent technologies for these tasks [11, 12]. The use of agents in WSNs is a logical continuation of this research.

3 The MERLIN protocol

3.1 Assumption

Sensor networks communicate in a multi-hop fashion hence packets are relayed from one node to another until a gateway is reached. Their number is usually very low compared with the total number of nodes in the network. We assume gateways to be synchronized (i.e to know the "perfect time"), to be responsible for data collection from the network and to be the medium, through which the user can both access the network and make changes to the network parameters if necessary.

In order to integrate the MAC, routing protocols and to include a possible localization procedure as an upper layer, MERLIN is designed to optimize the following data traffic patterns:

To gateway transmission by node Packets are sent to nodes located in the neighbouring zone closer to the gateway (i.e. lower zone).

Subnet flooding by gateway Gateway packets are forwarded to all nodes in the subnet;

Local broadcast by node Nodes send packets to all of the direct neighbors. No forwarding is performed;

Sector flooding Gateway packets are sent to all the nodes located in the sector of the network specified. The latter data traffic pattern is the result of a new feature of MERLIN and will be described in detail in section 3.4

3.2 Overview

The protocol MERLIN [1] integrates characteristics of MAC and routing in a simple, single architecture. Fundamental to MERLIN is the natural division of the network obtained when gateways start flooding the network simultaneously with an initialization message, for example *init-msg*, containing the transmitting time and the gateway ID number. At the beginning of the session, nodes are in receiving mode waiting for any message. Nodes receiving the initialization message are categorized as being in the first time-zone. The nodes will proceed to update the *init-msg* with their own node ID and forward it to further nodes. Once network flooding is complete, the network may be regarded as being subdivided into subnets each containing one gateway. Usually the gateway of reference is the closest one. Every subnet will then be organized into time zones. All nodes will belong to one time-zone only. Hence they can join the network by using the scheduling table provided. During this procedure, collisions can occur and some nodes may be notified later through an alternative path; this leads to an initial imprecision of the time-zone that will be corrected when the later messages are received. One of the primary benefits obtained from such an approach to flooding the network is the resultant time and space divisions of the nodes. This allows the potential reuse of the medium through effective scheduling tables as described in section 3.5.

3.3 MAC features of MERLIN

In this section, we describe characteristic of Medium Access Control (MAC) of MERLIN, the composition of slots and mechanisms for collided packet recovery. The basic technique of collision avoidance is the carrier sense multiple access CSMA, between nodes in the same zone. In fact, such nodes follow a channel contention procedure before they start transmitting the packet. Every slot starts with a *contention period CP* of about 30 times shorter than the total transmitting period. Any node, willing to transit a packet, firstly follows its scheduling table before picking up a *random time Tr* during the contention period at the beginning of its scheduled slot. The node will monitor the channel until T_r and then start transmitting if the channel is free. If the channel is busy then the packet is rescheduled for the next assigned slot. At the end of this procedure, the node switches to sleeping mode. The slot composition must also make provision for a *collision report period CR*, located between the end of the slot and the next CP. At such a time, the transmitting node switches to receiving mode so as to obtain a possible collision report, which is in the form of a short burst message. The collision report period has a double function:

- A collision can be notified back in order that the transmitter may reschedule the packet;
- Receiving nodes, which will apply CSMA to forward the packet, will implicitly acknowledge back to the transmitter their successful reception.

Finally, as required by some applications like in [13], MERLIN has a buffer where messages are stored while awaiting dispatch.

3.4 Routing properties of MERLIN

The division of the network into time-zones has the advantage of generating an implicit routing to the nearest gateway by means of the data-traffic field contained in each packet. Initially, nodes store their number of hop-counts to the nearest gateway. The hop-count number identifies the node's time-zone. Packets can only be forwarded in two directions: towards the gateway, or away from it. Furthermore, a local broadcast data type is also possible. Gateways are considered to be located in zone 0 with respect to nodes in their subnets. Nodes receiving a packet will forward it to nodes in a zone either one level higher or one level lower according to both the packet type and the scheduling tables.

Virtual Sectors in MERLIN In this section, the concept of *Virtual Sectorization* of the network as a novel feature of MERLIN is introduced. Virtual sectors are generated by simply using the CSMA approach already in use by nodes in the same zone. During initialization, nodes in zone 2, after receiving the init-msg from zone 1 nodes, will generate a *sectorID* based on the parent node ID in zone 1 and then proceed to flood the network with their *sectorIDmsg*. The sectorID is a unique number related to the zone 1 node ID-number. Because of the Contention Period (CP), only some nodes that are far apart can win the channel

simultaneously and then transmit. Receiving nodes in zone 1 will set their sectorIDmsg, then switch to sleeping mode. The rest of receiving nodes will now set both their time-zone as 3, generate their sectorID and then forward the message to further zones. The new init-procedure incorporates a *notify-msg* back from every node containing: node ID, timeZone and sectorID. For further information see Fig. 1. The notify-msg will enable gateways to address the correct location when looking for a specific node or a group of nodes.

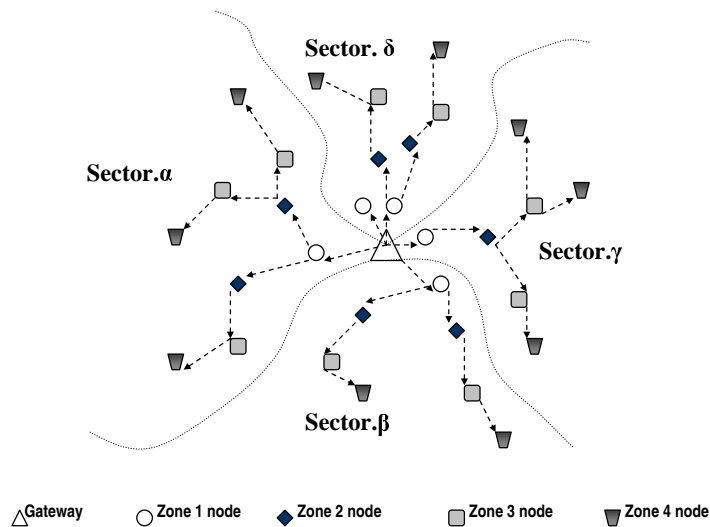


Fig. 1. The division of the network into sectors generated by zone 2 nodes by using the parent node ID

The table of neighbours The protocol MERLIN support a relative low mobility of nodes. As a result, every node must maintains a table of neighbours. The table is updated any time a node receive a packet from a new neighbouring node. Moreover the protocol includes periodical broadcast for the table of neighbours to be refreshed. For each neighbour, the table maintains the following information:

- Nodes ID;
- Time-zone;
- SectorID;
- Energy level.

As described in section 4.1, the table of neighbours is an important resource which facilitates network adaptation.

3.5 X and V Scheduling

MERLIN has been designed to support several scheduling tables that can be opportunistically switched when network conditions change. As described in [14], the X and V schedule tables in figure 2 have been studied and tested. Results derived from the implementation of the scheduling showed that the performance of the X-scheduling is more than twice that of the V-scheduling in terms of latency of messages and throughput. On the contrary, the V scheduling performs better in terms of network lifetime. Simulations reported a maximum setup time of 5 and 10 seconds for the X and V scheduling respectively. In case of a user request or a sudden application alteration, it is obvious that a facility for the autonomous and dynamic switching of scheduling is desirable. How this can be achieved is described in the next section.

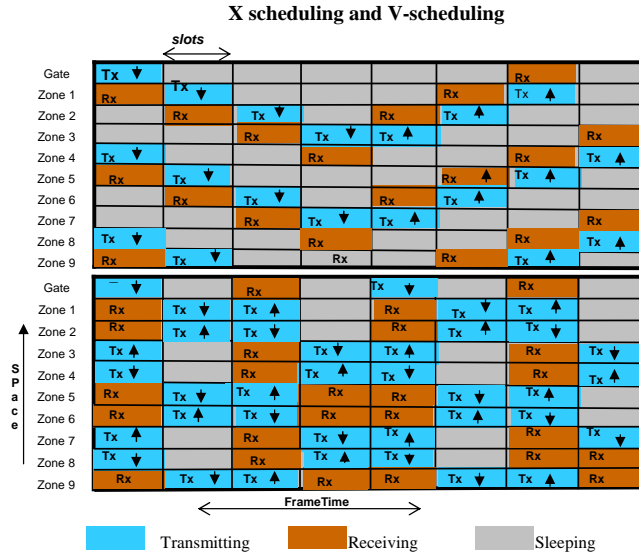


Fig. 2. Scheduling tables of MERLIN called X and V scheduling implemented and tested on the OmNet++ simulator.

4 Enabling Adaptive Scheduling within MERLIN

Realizing adaptive scheduling calls for a solution that is flexible, efficient and responsive. Potentially, a number of approaches exist that exhibit these characteristics. However, the intelligent agent paradigm was identified as one that encompassed the necessary criteria. Intelligent agents by their very nature imply a number of inherent characteristics. These include amongst others:

- Autonomy:** Agents act autonomously without a need for explicit interaction;
Reactivity: Agents can listen for and react to external events and changes in operating conditions;
Proactivity: Agents seek to be proactive in fulfilling their tasks;
Mobility: Agents have a capability to migrate to different nodes in the network while fulfilling their objectives.

Some researchers in the AI community subscribe to a stronger notion of agency [1] and envisage agents as being endowed with sophisticated reasoning facilities. One popular and computationally tractable implementation of such agents is that of one which conforms to the BDI architecture [15]. Such agents maintain a mental state that may include information about themselves and their operating environment. In the BDI scheme, the mental state takes the form of beliefs. Naturally all agents have a number of objectives or tasks to undertake. In BDI parlance, these are represented as desires. In practice, an agent can only realize its desires under certain predefined circumstances. When a situation arises that an agent is in a position to fulfill one of its objectives (or desires), it proceeds to do so. In the BDI case, the objective is formulated as an intention which the BDI agent immediately proceeds to fulfill. When realizing a software solution around agents, it is the software designer's prerogative to judge as to the necessity and appropriateness of the various agent characteristics. In the case of adaptive scheduling, the autonomous, reactive and mobile characteristics are most important. The BDI model offers an intuitive mechanism for modeling the continuous monitoring of the agent's environment and specifying the various criteria (usually in the form of rules) that determines certain agent actions, for example, under what conditions should the agent migrate to a certain node and under what circumstances should it commence a procedure to switch to a different scheduling model.

How a change in scheduling is Effected The X and V scheduling tables have the same slot length, same frame time and same number of zone as depicted in figure 2. As a result, they can be interchangeable under certain conditions and timing. Such an expedient makes the agent able to order a change of scheduling after the related parameters are identified, whenever it is considered appropriate. The agent can order a change of scheduling for the entire sector or a portion of it, for example from the zone N to the zone N+M in a sector. Such a situation will imply for nodes in the border zones (N and M) an adoption of both the old and the new scheduling to keep the continuity of message flow. In order to have a simultaneous scheduling adoption for the entire group of nodes involved, the migrating agent should firstly identify the number of zones that will join the change, secondly calculate the overall time necessary so that the packet can be forwarded to all the nodes interested, through the graph provided in [14]. Finally, the agent should generate two messages for:

1. *AdoptSector(V-SCHEDULING)* or *AdoptSector(X-SCHEDULING)*, the *Time* at which the event occurs and *timeZones* involved.

2. $AdoptSector(V_SCHEDULING)$ & $AdoptSector(X_SCHEDULING)$, the *Time* at which the event occurs and *timeZones* of nodes in the border.

4.1 Agents in MERLIN

In MERLIN, agents reside at the gateway and continuously monitor the status of the network. Information monitored includes:

- Total number of messages per minute received on average: *totalReceptRate* that exceeds a certain threshold *NetThreshold*
- Number of messages per minute received from an individual sector: *sectorReceptRate*
- Number of messages per minute received from an individual node: *nodeReceptRate*
- Percentage of data with the same sensed value received from a node, a sector, or a zone;
- Percentage of message with a value over a certain threshold received from a node, a sector or a zone: *NodThreshold*
- Changes to the sampling rate used in a node, sector or zone.

Though dependent on the purpose of the WSN, usually a number of scenarios could potentially arise that either require an alteration to the scheduling currently employed by the network as a whole, or more likely, in a sector or a zone. In the latter case, an agent would migrate to a particular node in order to investigate further. As an illustration of this, consider the following scenarios (note that the use of Agent Factory notation [16, 17] is used to express the various commitment rules):

1. The total number of messages received at the gateway from all the nodes in the subnet increases significantly in a short space of time. Such a situation implies that nodes are increasing their energy consumption considerably. Thus the agent needs to review the operation of the network and, if possible, identify a strategy for optimizing the longevity of the network. As previously mentioned, V scheduling is more suitable in situations where ensuring longer network lifetime is a priority. Therefore, the agent, based on parameters received, can make a unilateral decision to flood the entire network with an order to adopt V scheduling. In such circumstances, it does not need to migrate to any nodes in the network and can take the decision while remaining at the gateway. A commitment rule that it might adopt in such circumstances could be as follows:

$$\begin{aligned}
& BELIEF(totalReceptRate(?val)) \& BELIEF(NetThreshold(?trigger)) \\
& \quad \& BELIEF(currentScheduling(X_SCHEDULING)) \\
& \quad \quad \quad \implies \\
& COMMIT(Self, Now, Belief(True), AdoptGlobal(V_SCHEDULING))
\end{aligned}$$

2. The agent can from the gateway order a change in one more sectors that present an anomaly (e.g an extraordinary increase of the sector reception rate), then the commitment rule would be:

$$\begin{aligned}
& BELIEF(sectorReceptRate(?val)) \\
& \& BELIEF(SectorThreshold(?trigger)) \\
& \& BELIEF(currentScheduling(X_SCHEDULING)) \\
& \implies \\
& COMMIT(Self, Now, Belief(True), AdoptSector(V_SCHEDULING, ?T)
\end{aligned}$$

3. The number of messages received from a particular node increases beyond a predefined threshold. Clearly, this warrants further investigation. The agent can then decide to migrate by using the *nodeSector*, *nodeTimeZone* and *nodeID* provided at the gateway. The triggered commitment role is:

$$\begin{aligned}
& BELIEF(nodeReceptRate(?val, ?nodeID)) \\
& \& BELIEF(NodeThreshold(?trigger)) \\
& \& BELIEF(NodThreshExceed(TRUE)) \\
& \implies \\
& COMMIT(Self, Now, Belief(True), \\
& SEQ(Migrate(?nodeSector), Migrate(?nodeTimeZone))
\end{aligned}$$

4. On arriving at the relevant node, the agent proceeds to examine a number of parameters including:
- The number of messages in the buffer;
 - The values of the messages encountered whether exceed a percentage of similarity *similPercent*.

By using the node's table of neighbors, an agent can determine what nodes are nearby and migrate to each of these nodes to acquire further information concerning their status. For instance, the commitment rule to migrate to the neighbouring node is:

$$\begin{aligned}
& BELIEF(nodereceptRate(?val, ?nodeID)) \\
& \& BELIEF(NodeThreshold(?trigger)) \\
& \& BELIEF(NodThreshExceed(TRUE)) \\
& \implies \\
& COMMIT(Self, Now, Belief(True), Migrate(?nodeID))
\end{aligned}$$

Should the agent require, (in case it has enough information to believe that one or more sectors present an anomaly), it can order a change of node scheduling at time *T* or an adoption of the two scheduling tables for nodes in the border zones as described in section 4. Consequently, the agent can commit to further migrating to a node in a further time zone or sector; In such a case the agent determines the *timeZone*, *sectorID* and prevailing scheduling mechanism in use before proceeding to migrate. The related sequence of commitment rules is:

$$\begin{aligned}
& BELIEF(BufferSize(?val)) \\
& \& BELIEF(similPercent(?threshold)) \\
& \implies \\
& COMMIT(Self, Now, Belief(True), \\
& SEQ(AdoptZone(?ZoneID, V_SCHEDULING, ?T), \\
& Migrate(?timeZone + 1, ?sectorID, ?nodeID)))
\end{aligned}$$

5 Conclusion

This paper has investigated the use of intelligent agents in the delivery of adaptivity at the networking layers. This is achieved by using two preexisting technology sets developed in part by the authors; these are: the energy-efficient integrated MERLIN protocol and Agent Factory a rapid prototyping environment for agent deployment. Effective and efficient use of limited energy resources is of paramount importance within WSNs. This research has described a method of optimizing energy resources in times when unexpected or heavy network activity occurs. Three instruments facilitate this: the provision of two efficient and interchangeable scheduling tables; the ability to generate virtual network sectors; the adoption of autonomous mobile agents. Such agents offer the deductive apparatus by which WSN adaptivity is delivered. Agents monitor network activity and determine which of the two scheduling regimes would be most appropriate at the network either level or alternatively through the creation of virtual network sectors at the sector level. Autonomous agents can deliberate and dynamically apply the respective schedules at either network or sector level. Determination of the network context in order to inform such decisions will often necessitate agent migration. This approach maximizes network performance while minimizing energy usage. Agent characteristics of autonomy, social ability and mobility suggest that they represent an intuitive choice for this task. Ongoing research is investigating utility functions, which underpin network adaptivity based on incomplete, localized, conflicting, or partial network information.

6 Acknowledgments

Gregory O'Hare, Antonio G. Ruzzelli and Richard Tynan gratefully acknowledge the support of Science Foundation Ireland under Grant No. 03z/IN.3/1361. Michael O'Grady gratefully acknowledges the support of the Irish Research Council for Science, Engineering & Technology (IRCSET) through the Embark Initiative postdoctoral fellowship programme.

References

1. Ruzzelli, A., Evers, L., Dulman, S., Hoesel, L.V., Havinga, P.: On the design of an energy-efficient low-latency integrated protocol for distributed mobile sensor networks. IWWAN International Workshop on Wireless Ad hoc Networks (2004)

2. Ye, W., Heidemann, J., Estrin, D.: Medium access control with coordinated adaptive sleeping for wireless sensor networks. Twenty-First Annual Joint conference of the IEEE Computer and Communication Societies (INFOCOM) **3** (2002) 1567–1576
3. Rajendran, Obrazka, Garcia-Luna-Aceves: Energy-efficient, collision-free medium access control for wireless sensor networks. Conference on Embedded Networked Sensor System (2003) 181–192
4. Dam, T.V., Langendoen, K.: An adaptive energy efficient mac protocol for wireless sensor networks. ACM Sensys (2003)
5. Hoiydi, A.E., Decotignie, J.: Wisemac: An ultra low power mac protocol for multi-hop wireless sensor networks. In: Proceedings of the First International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS 2004), Lecture Notes in Computer Science, LNCS 3121. (2004) 18–31
6. Johnson, D.B., Maltz, D.A.: Dynamic source routing in ad hoc wireless networks. Mobile computing **353** (1996) Kluwer Academic publishers.
7. Akan, O.B., Akyildiz, I.F., Sankarasubramaniam, Y.: Event-to-sink reliable transport in wireless sensor networks. IEEE-ACM Transactions on Networking (2004)
8. Lu, G., Krishnamachari, B., Cauligi, Raghavendra, S.: An adaptive energy-efficient and low-latency mac for data gathering in sensor networks. International workshop on Algorithms for Wireless, Mobile, ad Hoc Sensor Networks (WMAN 04) (2004)
9. Hoesel, V., Chatterjea, Havinga: An energy efficient medium access protocol for wireless sensor networks. proRISC 2003 (2003)
10. Bieszczad, A., P.B., T., W.: Mobile agents for network management. IEEE Communications Surveys **1** (1998)
11. Haque, N., J.N.R.M.L.: Resource allocation in communication networks using market-based agents. International Journal of Knowledge Based Systems (2005)
12. Wang, Y., C.L.B.J.: Intelligent radio resource management for ieee 802.11 wlan., IEEE Wireless Communications and Networking Conference (WCNC), Atlanta, Georgia USA (2004)
13. Kevin Mayer, K.T., Ellis, K.: Cattle health monitoring using wireless sensor networks. The 2nd IASTED International Conference on Communication and Computer Networks, Cambridge Massachusetts (2004) 8–10
14. Ruzzelli, A., Tynan, R., G.M.P.O'Hare: A low-latency routing protocol for wireless sensor networks. To appear on SENET'05 Advanced Industrial Conference on Wireless Technologies. Montreal (2005)
15. Rao, A.S., G.M.: Modelling rational agents within a bdi architecture., Principles of Knowledge Representation. and Reasoning, San Mateo, CA. (1991)
16. G.M.P., O.: Agent Factory: An Environment for the Fabrication of Multi-Agent Systems, in Foundations of Distributed Artificial Intelligence. John Wiley and Sons (1996)
17. Collier, R.W., O.G.L.T.R.C.: Beyond prototyping in the valley of the agents, in multi-agent systems and applications iii. Proceedings of the 3rd Central and Eastern European Conference on Multi-Agent Systems (CEEMAS'03), Prague, Czech Republic **3** (2003) Lecture Notes in Computer Science (LNCS 2691), Springer-Verlag.