An Energy Efficient Multiagent Middleware for Physical Complex System

Jean-Paul Jamont[1], Michel Occello[2] and André Lagrèze[2]

[1] Institut National Polytechnique de Grenoble, LCIS, 26000 Valence, France

[2] Université Pierre Mendès, LCIS/INPG, 26000 Valence, France

Abstract. Open physical complex artificial systems involve wireless autonomous entities submitted to strength constrainted energetic policies. Their features naturally lead to apply multiagent techniques to ensure both the autonomy of entities and the best whole system organization. We propose a multiagent approach for wireless communication robust management for such physical system using self-organization mechanisms. We show the performances of this approach by comparison to usual wireless protocols. The genericity of our contribution is highlighted by the proposition of a middleware layer integrated in agent. We give an insight to a real world instrumentation application using the middleware.

1 Introduction

A complex system can be defined as composed of many elements which interact each other and with their environment. These interactions are often non-linear and generally contain feedback loops. At the global level, these systems are characterized by the emergence of not observable phenomena at a local level: an external observer will understand differently the system than an internal observer. Complex systems are thus characterized by the emergence at a global level of new properties and of a new dynamic which is not easily predictible from the observation and the analysis of the elementary interactions.

Working with physical systems like collective robotics or massive instrumentation imposes the use of wireless technology. The features of such complex cognitive physical system leads to naturally apply multiagent techniques to ensure both the autonomy of entities and the best whole system organization [3].

This paper presents the different steps of the design of an energy efficient middleware. In a first section we present the necessity to adopt a message oriented middleware (MOM) for our applications. We then propose our multiagent approach based on self-organization to manage the communication of these decentralized embedded nodes networks. We give finally an insight to some quantitative results showing the benefit of a multiagent approach compared to traditional protocols in a real world application of an underground river system intrumentation.

Jamont J., Occello M. and Lagrèze A. (2005).
An Energy Efficient Multiagent Middleware for Physical Complex System.
In *Proceedings of the 1st International Workshop on Artificial Neural Networks and Intelligent Information Processing*, pages 72-81
DOI: 10.5220/0001196200720081

2 Necessity of an energy efficient MOM

Our research works deal with embedded multiagent systems (MAS) like collective robotics or physical instrumentation. Considering complex embedded control systems as networks of decentralized cooperative nodes is an attractive way to design physical intelligent applications [6, 3].

Multihop communication. In wireless networked system, communication between two hosts is generally not direct. To communicate, entities require help from other hosts (multihop communication). Such a requirement creates an important routing problem because the location updating of neighbors is difficult. All adapted wireless routing protocols use flooding technics. In a flooding technique, a host gives the message to all its neighboors which do the same. These wireless protocols can be classified by their families : The *reactive* protocols using no routing table, the *proactive* protocols using routing tables, periodically updated and *hybrid* protocols adopting a reactive protocol behavior and, if necessary, using routing tables for increasing efficiency.

Limited power ressources. Agents have limited power ressources. One of the whole system aim is so to reduce as much as possible the energy expense. When they have nothing to do generally for sparing energy they enter in a sleep mode. When they communicate they must use good routing protocols and optimal ways (generally the criteria will be the number of hops). But they must decrease as much as possible the flooding scheme because the associated power cost is very high. An agressive environment like underground river system (as for one of our applications) can cause some internal faults for agent. The communication infrastructure must be very adaptive, fault tolerant and self-stabilized : an agent failure must not have an important impact on the system. This system must provide reliable communications and, sometimes, must adapt to ”real-time” constraints. Furthermore, in the case of mobile devices the infrastructure of sytems are not persistant.

Message oriented middleware. We need to design a mobile communication management layer to manage the wireless communications between the different agents in the system. This layer must increase interoperability, portability and flexibility of an application by allowing the application to be distributed over multiple heterogenous agents. It must reduce the complexity of agent development. This layer will be a Message Oriented Middleware (figure 1).

3 A multiagent approach to design energy efficient MOM

3.1 Multiagent systems

Multiagent method. Multiagent methods aim at decreasing the complexity of system design by a decentralized analysis. We will follow the method of multiagent design discussed in [7] based on the AEIO according to four axes collectively accepted today: The *agent aspect* gathers all elements together for defining and constructing these entities. The *environment aspect* for dealing with the analysis of environment elements and with capability such as the perception of this environment and the actions one can do on it. The *interaction aspect* includes all elements which are in use for structuring

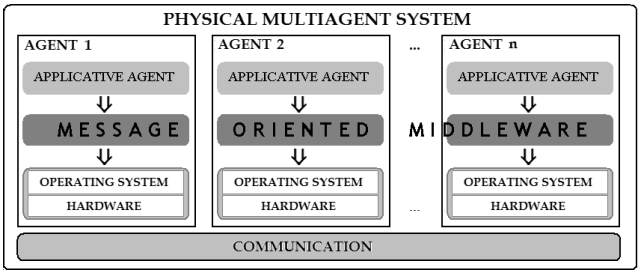

Fig. 1. Our embedded multiagent system architecture

the external interactions among the agents (communication language, interaction protocols). The *organization aspect* allows to order agent groups in organization determined according to their roles.

Multiagent systems and wireless networks. The distributed and open nature of sensor networks means that the multiagent approach is an adapted answer. Another advantage of this approach is the external representation of the interactions and of the organization. External representations offer multiple possibilities such as the monitoring by an external observer.

A few works reaching the same objectives show that the approach is interesting. We can quote the ActComm [2] project which is a military project for which the routing of information is essential: it aims at studying the communication management between a soldier team and a military camp via a satellite. We can also mention the work on wireless networks of mobile autonomous intelligent sensors [9] where agents are used to achieve open flexible cell assembly.

Our Message Oriented Middleware (MOM) must be economic in an energy point of view : it is one of the main differences with the other works on multiagent based middleware [1, 5].

3.2 The agent aspect

An agent is a software entity situated in an environment which it can perceive and in which it acts. It is endowed with autonomous behaviors and has objectives. Autonomy is the main concept in agent issue: it is the ability of agents to control their actions and their internal states. The autonomy of agents implies no centralized control. The power of an agent decomposition is the decentralization of the intelligence, i.e. the decision capabilities, and of entities' knowledge. A MAS is a set of agents situated in a common environment, which interact and attempt to reach a set of goals. Through these interactions a global behavior can emerge. The emergence process is a way to obtain, from cooperation, dynamic results that cannot be predicted in a deterministic way.

These agents have hybrid architectures, i.e. a composition of some pure types of architectures. Indeed, the agents will be of a cognitive type in case of a configuration alteration, it will be necessary for them to communicate and to manipulate their knowledge in order to have an efficient collaboration. On the other hand, in normal use it will

be necessary for them to be reactive (stimuli/response paradigm) to be most efficient. All the agents have the same communication capabilities but the communicated data depend of their roles.

3.3 The environment aspect

This part of the analysis deals with elements necessary for the MAS realization such as the perception of this environment and the actions one can do on it. The environnement can be [10]: *Accessible*, if an agent using the primitives of perception can determine the state of the environment. When an environment is *inaccessible* it is then necessary for the agent to memorize features in order to record the modifications which occur. *Determinist*, or not, according to whether the future state of the environment is, or not, fixed by its current state and the actions of the agent. *Episodic* if the next state of the environment does not depend on the actions carried out by the agents. *Static* if the state of the environment is stable while the agent reasons. *Discret* if the set of the feasible actions and states of the environment is finished.

The environment will be made of measurable information. It is deterministic, non episodic, dynamic and continuous. Agents can move in this physical environment but don't know their position.

3.4 The organization aspect

In this type of application no one can control the organization a priori. Relations between agents are going to emerge from the evolution of the agents'states and from their interactions. We are going to be content with fixing the organization parameters, i.e. agents'tasks, agents'roles.

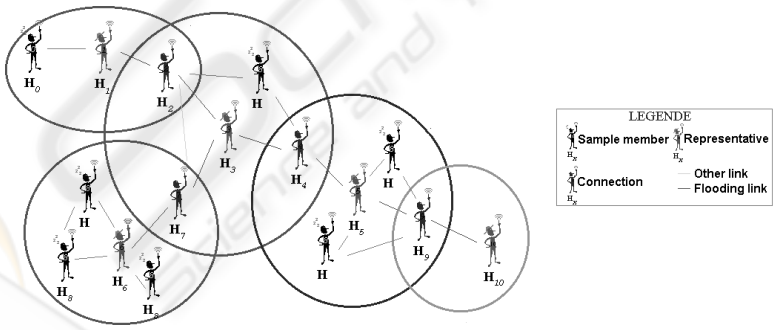

Fig. 2. Group organization

Our organizational basic structures are constituted by (see fig 2) : one and only one *group representative agent* (r) managing the communication in its group, some *connection agents* (c) which know the different representative agents and can belong

to several groups, some *simple members* (s) which are active in the communication process only for their own tasks.

With this type of organizational structure, the message path between the source (a) and the receiver (b) is $((a, r),^{*} [(r, c), (c, r)], (r, b))$. If the source is a representative agent the first term doesn't exist. If the receiver is a representative agent the last term doesn't exist.

Because a representative agent is the most sollicited agent in a group, the best one is the one having the most important level of energy and the most important number of neighbors. We use a role allocation based self-organization mechanism involving the election of a representative agent based on a function which estimates the adequation between its desire to be the boss and its capacity to be: so the organization is modified only when a problem occurs. We don't try to maintain it if we have no communication.

The energy saving is obtained owing to the fact that the flooding is only directed to the representative agents of the groups and to some connection agents. To give an order of idea, a receiver path research with flooding techniques will cost, in the case of a traditional wireless network, a number of emissions equal to the number of stations. In the case of a clustered wireless network, the number of transmitted messages are about twice the numbers of representative agents (all the representative agents are contacted via one connection agent).

However, the networks with an organizational structure must take care of the maintenance of their routing table. Generally, the adaptive features of these tables come from periodical exchanges between the different nodes. In our approach we do not wish to use this technique to ensure the maintenance of coherence. Indeed, our principle will be "if we do not need to communicate, it is useless to spend energy to ensure the coherence maintenance". However, we will thus use eavesdropping of surrounding agent communications. We extract knowledge from these messages exchange to update our beliefs about our neighboors. Moreover, our self-organization mechanism will integrate an energy management policy. These structures will thus emerge.

Our algorithm (figure 3) can be adjusted by other agents' suggestions such as an organization inconsistency. Moreover, an agent can give up its role because its power level quickly fall or fall under a limit that the agent thinks dangerous for its integrity. So it can become a sample member.

3.5 The interaction aspect

The agents will interact only with the agents in acquaintance. Agents interact by asynchronous exchange of messages. Among the different protocols that we use, the choice of an introduction protocol is essential. Indeed, this protocol allows to the agents to be known, i.e. to bring their knowledge and their know-how to the agents' society. An other important protocol is the "best representant election" protocol seen previously. These protocols are an arrangement of some of the thirteen different types of small messages defined in the following: *WhoAreMyNeighbors* is used by an agent to know who its neighbors are. This message is transmitted when an agent is created (the first goal of a new agent is to know its neighbors) or when an agent feels that its neighbor table is not coherent with reality. *IAmOneOfYourNeighbors* makes it possible for an agent to answer the preceding request. With this message, it thus provides its identifier,

```
IF neighnorNumber ≠ 0 THEN
    * One has neighbors
    IF neighnorRepresentativeNumber=0 THEN
        * None of our neighbors is representative: one decides to become to it. This case intervenes when one
        * has just created the agent or when he is unstable (the agent goes surely to carry on its path)
        myRole = REPRESENTATIVE;
    ELSE IF myRole = REPRESENTATIVE THEN
        * I am a representative agent too: I enter in conflict with the other applicants to this role an election
        * will take place and the agent with the best score will remain in place.
        RepresentativeElectionProcedure()
    ELSE IF neighnorRepresentativeNumber=1 THEN
        * One of our neighbors is representative: one subjects oneself to its authority and his even if the
        * organization is less effective than otherwise.One privileges, for the moment, stability to
        * performance in the organization. One will await a failure or its wish to leave its mandate.
        myRole = SIMPLEMEMBER
    ELSE
        * There are, in our vicinity, several representatives: one becomes connection agent for these representatives
        myRole = CONNECTION
    ENDIF
ELSE
        * One does not have a neighbor: one has any more no role
        myRole = NOTHING
ENDIF
```

Fig. 3. Self-organization algorithm

its role and its membership group. *IChangeMyRole* is used by any agent to inform its neighbors that it decides to change its role. *AskConnectionAgentGroup* is used by representative agents which want to update their knowledge on the close groups. It obliges the connection agent to answer. *AnswerConnectionAgentGroup* is used by a connection agent to announce to a representative agent the other representative agent that it can contact. *VerifyNeighborGroupConsistency* is sent by an agent, to its representative, which believes to have detected an inconsistency with a close group. There is an inconsistency between two groups when two agents of different groups see themselves and their representative cannot communicate with a short path (fixed by a time-to-live). *ConflictRepresentativeResolution* is used by a representative agent, in conflict with one or more other representative agents, to communicate its score. *ISuggestYouToBeRepresentative* is a suggestion given by a representative to one of the agents of its group. *FindPacketPath* is used by a representative agent which wants to know the path (list of representative agents) to join another agent. *PacketPathResult* is the answer of the representative of the recipient of the $FindPacketPath$ message. *ACKMessage* is a configuration message used to confirm to the transmitter that its message arrived to its destination. *BadWay* is a message sent by a representative who noticed a problem. This message takes the erroneous road and the organization verifies its consistency. *EncapsuledData* is a message which encapsulates data.

For example, the introduction protocol consists in transmiting *WhoAreMyNeighbors*, receiving the answer of neighbor agents (*IAmOneOfYourNeighbors*), choosing a role according to the algorithm (figure 3) and transmiting a *IAmOneOfYourNeighbors* message.

4 Implementation and evaluation

4.1 Implementation

Therefore, we will demonstrate the feasibility of our approach in the case of the instrumentation of an underground hydrographic system [3]. In a subterranean river system,

the interesting parameters to measure are numerous: temperature of the air and the water, air pressure, pollution rate by classical pollutants, water flow, draft speed etc. All these information will be collected at the immediate hydrographic network exit by a workstation. These data will be processed to activate alarms, to study the progress of a certain pollution according to miscellaneous measuring parameters, to determine a predictive model of the whole network by relating the subterranean parameters measures of our system with the overground parameter measures more classically on the catchment basin.

We have chosen for sensors a classical three-layers embedded architecture. We use **the physical layer** which is employed by NICOLA system, a voice transmission system used by the French speleological rescue teams. This layer is implemented in a digital signal processor rather than a full analogic system. Thereby we can keep a good flexibility and further we will be able to apply a signal processing algorithm to improve the data transmission. **The link layer** used is a CAN (Controller Area Network) protocol stemming from the motorcar industry and chosen for its good reliability. **The applicative layer** is constituted by the agents' system. A hybrid architecture enables to combine the strong features of each of reactive (to the message) and cognitive capabilities (to detect inconsistency and re-organisation). The ASTRO hybrid architecture [6] is especially adapted to a real time context. The integration of deliberative and reactive capabilities is possible through the use of parallelism in the structure of the agent. Separating Reasoning/Adaptation and Perception/Communication tasks allows a continuous supervision of the evolution of the environment. The reasoning model of this agent is based on the Perception/Decision/Reasoning/Action paradigm. The cognitive reasoning is thus preserved, and predicted events contribute to the normal progress of the reasoning process.

The *communication module* calls the MAS middleware services supplied through a component. The agent must use a `WCommunication` package, write in Java langage and translated into C++ langage because a lot of physical plateforms use this langage. This package contents two abstracts classes (`Identifier` and `Message`) and two main classes called `Communication` and `BitField`. In the `Message` abstract class the designer must implement the primitives to convert the message in a bit field (`BitField MessageToBitField(Message m)` and the recipocal primitive `Message BitFieldToMessage(BitField b)`). In the Identifier abstract class the designer must implement the type of identifier and two primitives `BitField IdentifierToBitField()` and `Message BitFieldToMessage(BitField b)`. The primitive to convert the identifier in a bit field must be implemented by the designer. The Communication class contains a list of couples `(Identifier,Message)` for the emission and the reception. This list is private and must be accessed via `Bool SendMessage(Identifiant, Message)` and `CoupleIdentifierMessage ReceiveMessage()`.

The package must be connected to the operating system. The operating system must give the battery energy level (primitive `SetBatteryLevel(Float l)` to the `Communication` class and must give the bit field which arrives. In an other hand, the middleware gives to the operating system the bit field to send by calling `BitField GetBitFieldToSend()`.

These agents are embedded on autonomous processor cards equipped with communication modules and with measuring modules. The KR-51 (the real time kernel's name) allows multi-task software engineering for C515C microcontroller. We can produce one task for one capability. We can then quite easily implement the parallelism inherent to agents and satisfy the real-time constraints.

4.2 Evaluation

In order to evaluate and improve such agents' software architectures and the cooperation techniques that they involve, we introduce a simulation stage in our development process. The simulation first allowed us to experiment our approach and the software solutions that we provide for the various problems.

The simulation step. The simulation software structure is very basic. We have two types of components: SimSensor and SimNetwork. A SimSensor component simulates the sensor behavior. It possesses its own model and architecture. All the sensors have the same communication capabilities. They transmit their requests to the SimNetwork component which sends this information to all sensors which can receive them, in the environment. SimNetwork can appear as the inference mechanism for the simulation.

We have compared our MAS to three traditionnal solution based on ad-hoc protocols. The DSDV protocol (Destination-Sequenced Distance-Vector protocol [8]) and the natural DSR protocol (Dynamic Source Routing protocol [4]) do not appear in this comparaison because its efficiency were lower than the ehanced version of DSR which uses a route maintenance (memorization of the main route). We thereafter call efficiency the ratio between the theoretical useful volume of the optimal way divided by the volume of each transmitted communication.

Case of an application with unidirectional communication. All agents communicate only with the workstation situated at the end of the undergound river system : it is a unidirectionnal protocol. In this case, messages are small. For this example, three messages are send by five seconds. The same scenario is applied for the different protocols.

We can see that the benefit (fig 4) of our approach is important. Our routing method can deliver quickly all messages with a good efficiency. Higher is the number of sensors better is the reactivity of our approach. We must note that if the system knows no pertubation or mobility variation of DSR will be better from an efficiency point of view is normal because in this case DSR learns all the routes (succession of sensors) allowing to communicate with the workstation. It is not really the case of our approach which reasons about the group and not from the sensors. One consequence is that the routes used by the messages with our approach are not optimal. We can see that our approach support the addition of a lot of sensors. The number of groups don't explose with the number of sensors but their density increases.

Case of an application with multidirectionnal communication. In this case, we envisage communication between collaborative sensors. We choose to give to the message a size of thirty bytes (elaborated measure). In this case the behavior of our approach is

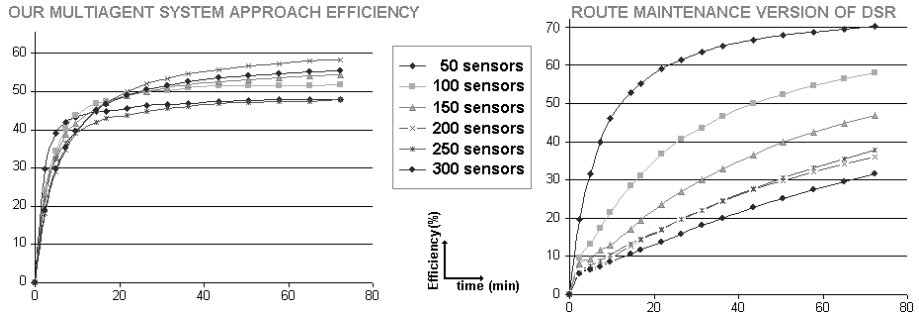

Fig. 4. Approach comparison for unidirectionnal use case

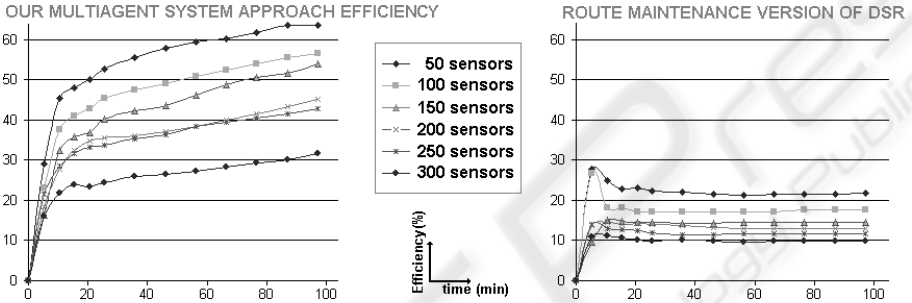

Fig. 5. Approach comparison for the multidirectionnal use case

much better than DSR because its route management is more complicated. If we add some perturbations on these scenarios (one perturbation by three minutes) the efficiency is nearly the same (it is not the case for the DSDV protocol).

5 Conclusion

We have presented in this paper a MAS to manage wireless communication between agents in respect to energy constraints. From our different works on physical complex systems, we concluded that more genericity can be introduced in communication management. We have built a multiagent middleware based on a multiagent approach which allows to make abstraction of this energy efficient communication management at the application level. We use this middleware in the case of a wireless sensor network. In this application, all the agents have a hybrid decisionnal architecture based on the ASTRO model. The middleware is included in the ASTRO communication module.

Agents present interesting features of software engineering such as genericity allowing an easy evolution of the applications. Generic aspects of agents allow us to envisage differents applications for this network type such as diagnosis, risk management, data fusion...

References

1. M. Calisti, T. Lozza, and D. Greenwood. An agent-based middleware for adaptative roaming in wireless networks. In *AAMAS 2004 Workshop on Agents for Ubiquitous Computing*, 2004.
2. R.S. Gray. Soldiers, agents and wireless networks: A report on a military application. In *Proceedings of the Fifth International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, 2000.
3. J.-P. Jamont and M. Occello. Using self-organization for functionnal integrity maintenance of wireless sensor networks. In *Proceedings of IEEE International Conference on Intelligent Agent Technology*, 2003.
4. D.-B. Johnson and D.-A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.
5. M. Mamei and F. Zambonelli. Self-organization in multi-agent systems : a middleware approach. In *AAMAS 2003 Workshop on Engineering Self-Organising Systems*, pages 233–248, 2003.
6. M. Occello, Y. Demazeau, and C. Baeijs. Designing organized agents for cooperation in a real time context. In *Collective Robotics*, volume LNCS/LNAI 1456, pages 25–73. Springer-Verlag, March 1998.
7. M. Occello and J.L. Koning. Multi-agent based software engineering: an approach based on model and software reuse. In *From Agent Theory to Agent Implementation II - EMCSR 2000 Symposium*, pages 645–657, Vienna, April 2000.
8. C.E. Perkins, E.M. Royer, and S. Das. Highly dynamic destination-sequenced distance-vector (dsdv) routing for mobile computers. In *ACM SIGCOMM'94*, 1994.
9. Petriu, E.M. et al. Intelligent robotic sensor agents for enviroment monitoring. In *Proceedings of IEEE International Symposium on Virtual and Intelligent Measurement Systems*, pages 19–20, 2002.
10. M.J. Wooldridge, N.R. Jennings, and D. Kinny. The gaia methodology for agent-oriented analysis and design. In *AAMAS*, volume 3, pages 285–312. KAP, 2000.