Context-Aware Emergent Behaviour in a MAS for Information Exchange

Andrei Olaru, Cristian Gratie, Adina Magda Florea

CONTEXT-AWARE EMERGENT BEHAVIOUR IN A MAS FOR INFORMATION EXCHANGE*

ANDREI OLARU , CRISTIAN GRATIE , AND ADINA MAGDA FLOREA †

Abstract. The plethora of interconnected devices that surrounds modern people has yet to work together as a whole. An intelligent environment must sense and react to the actions of people, but to that end a large quantity of information must be exchanged throughout the system. Under realistic conditions, it is impossible to control and coordinate the exchange of information in a centralized way. Solving this problem involves key concepts like self-organization, emergent behaviour and context-awareness. Continuing previous work on self-organizing cognitive multi-agent systems for the exchange and management of information, this paper introduces two aspects of context-awareness – pressure and interest – that make the system's emergent behaviour more context-sensitive and, therefore, more adaptive to a changing environment.

Key words. multi-agent system, self-organization, cognitive agent, context-awareness

1. Introduction. People in the modern world are surrounded by a huge number of electronic devices that have different capabilities, different sizes and different performance, all of them interconnected by wireless or wired networks, but not quite working together and cooperating towards the resolution of tasks.

Ambient Intelligence – or AmI – is the field that deals with electronic environments that are sensitive and responsive to the presence and actions of people [1]. It refers to a ubiquitous electronic environment that supports people in their daily tasks, in a proactive, but "invisible" and non-intrusive manner [14, 20]. It implies, on the one hand, embedded, natural and personalized interfaces and, on the other hand, an underlying ubiquitous network that links the devices together into one single system that acts as a whole.

An Ambient Intelligence environment may be regarded as comprising several layers [17]: the hardware – comprised of sensors, actuators, mobile devices, computers, intelligent appliances and material; the network – an ubiquitous infrastructure that provides connectivity by means of various wired and wireless, possibly ad-hoc, media and protocols; the interoperability layer – that will allow uniform communication among heterogeneous devices using different hardware and different types of connections to the network and will enable easy information transfer and code migration between devices; the application layer – that will be more oriented towards knowledge and semantic information and will offer context-aware services to the layer above; and the intelligent user interface – that will offer to the user a natural, intuitive, non-intrusive and multi-modal means of interaction with the system.

From our point of view, one essential issue is what happens *between* the upper layer of the human-machine interface and the layers of interoperability, network, and hardware – i.e. at the application layer. The exact manner in which information is transferred between devices is far from having a trivial solution. First, all aspects of one's life taken into account, the quantity of information that transits the system is very large. Second, most of the devices that use the information (mobile phones, mp3 players and even simple sensors and actuators) have reduced storage and processing capacity. Considering the required flexibility of the system and the number of devices involved, it is obvious that centralized control is not viable, therefore the system

*This research was supported by Grant CNCSIS ID_1315, 2009-2011, and Grant POSDRU 5159.

†Department of Computer Science, University Politehnica of Bucharest, 313 Splaiul Independentei, Bucharest, 060042 Romania (cs@andreiolaru.ro, cgratie@yahoo.com, adina@cs.pub.ro)

must self-organize. Moreover, considering the very context-aware nature of Ambient Intelligence, the vast majority of the information that exists in the system is bound to a certain type of context: temporal, social, but most times spatial.

An appropriate paradigm for the implementation of AmI, addressed especially to the application layer, is the agent-oriented approach. Agents have features like reactivity, proactivity, autonomy and reasoning. Cognitive agents are by definition fit to working with information in a more semantic manner. Adding self-organization [18] to a multi-agent system allows for decentralization and resiliency, as well as for a greater degree of flexibility and adaptivity.

We have addressed the issue of a self-organizing system for the management of information in our previous work [11, 12]. However, the notion of context representation and awareness had not yet been integrated.

This paper takes a step forward towards the implementation of context-awareness in a system for the distributed management of information, using agents that are cognitive but have limited storage capacity. In this case, a simple and generic enough representation of context is necessary, one that is easy to process but also useful enough for the functioning of the system. The paper proposes two elements of context representation, namely *pressure* and *interest*. The two elements relate to two aspects of context awareness: the relevance that the source associates to a piece of information (this is the pressure) and the relevance that the receiver associates with the piece of information (this is the interest).

A system for the emergent management of information, using the two aspects of context awareness, has been designed and implemented. The system has a generic design, but mainly there is an agent in the system that is assigned to each user of the system, and that manages the information that is available to, or made available by, the assigned user. The system has been successfully tested on several scenarios, that, though simple, involve a great number of agents and multiple pieces of information.

Section 2 is dedicated to related work in the fields of context-awareness, emergence and self-organization. Section 3 presents the requirements and the general description of the implemented system. Section 4 introduces some aspects of context awareness in the described system and two elements for the representation of context are proposed: pressure and interest. Section 5 describes the design of the system and the results of the experiments are discussed in section 6. The last section draws some conclusions.

2. Related Work. In the domain of Ambient Intelligence there are many directions of research, ranging from intelligent user interfaces to sensor networks to mobile agents offering services. Architectures for the management of the layer offering context-aware services generally address particular scenarios: conference rooms [8], intelligent buildings [15] or other limited smart spaces [9]. Even generical architectures are oriented towards scenarios implying a small number of individuals, which allows the use of man centralized components [16, 3, 17]. The architecture presented in this paper aims to provide a platform that is scalable and that relies on no centralized components.

Self-organization and emergent behaviour in multi-agent systems are usually inspired from biological systems and have been studied mostly by using reactive agents [18, 10]. Most of the obtained emergents consist of the organization of agents in a spatial or space-related structure that groups agents with certain states: placement of agents in circular shapes or shapes of more complex form [21], coverage of certain areas [2], space related dynamical behaviour [13]. The limitation of the capabilities of these systems comes from the use of reactive agents, that, as opposed to cogni-

tive agents, are very simple and therefore cannot lead to very complex functions at the level of the system. Self-organization has also been studied for systems based on agents having a more elaborate model, but such approaches are less frequent [6, 7] and less adaptive. Moreover, they have not so far been used for the management of information or for the exchange of agent's beliefs.

Research in context-awareness offers complex solutions even in the field of mobile devices [4], but the representation of context lacks generality and is difficult to implement using really small amounts of memory. There are many manners in which complex information may be tackled, for example by using complex ontologies [19]. However, that means that either the device must be able to work with ontologies, or that there has to be a centralized service that offers the possibility of responding to context queries [9], which may not be scalable. This paper aims to offer a solution that requires very low storage and processing capacity. Moreover, the agent's behaviour is tuned in accordance to the context, enabling it to act with increased promptness whenever the situation requires it.

Our previous work [12] described a system for the exchange and management of information. Unlike most studies of self-organizing systems, our approach used cognitive agents instead of reactive ones, because of their increased flexibility and possibilities. The interaction between agents lead to emergent properties related to the uniform distribution of data. However, the context was not taken into account. In an AmI environment, context is essential as most information is related to, and may only have sense in, a certain context, be it spatial, temporal, social or computational [4]. Most times the spatial aspect is considered [9, 8], as it is directly related to directly available resources and services.

3. System Requirements. The purpose of the system we designed is to manage information in such a way that its users will have interesting or needed information available without the need to know where this information comes from or how it was made available to them. From the user's perspective, two operations are possible: either insert information into the system or request certain information he/she might be interested in. Also, the system might provide the user with information that is potentially interesting to him/her. We will discuss each of these key concepts in the following paragraphs.

At this point, there is no specific, real-life application of the system, as we have tried to keep it as generic as possible. However, many applications may be imagined, as most Ambient Intelligence scenarios imply heavy exchange of information between users or between users and other entities (e.g. smart spaces, AmI services), trying to deliver useful information to the users that need it [5].

The system is implemented as a cognitive multi-agent system. Agents hold information in the form of beliefs, and have goals and plans on what to do with that information. The system has been designed as to manifest self-organization, therefore the agents have been given a behaviour that is local, but that reflects the global, desired, objectives. To that end, agents have been given some human-inspired features, like curiosity, states of being under pressure, and the ability to forget. What each agent should do is to (1) realise what information may be of interest to the associated user and get it and (2) provide other agents with information that it believes that may be interesting to them.

The different pieces of information that exist in the system will be referred to generically as *data*. Data is characterized by its content, by the size of its content and by context-specific information. The actual content of data should not be mistaken

for beliefs about that piece of data. For example, a piece of data may be the recipe of how to prepare a certain kind of food. The fact that an agent A knows that B has the recipe (and knows how to prepare the food) does not mean that A can prepare the food, but this knowledge can help A get hold of the useful information.

4. Aspects of Context Awareness. In previous experiments with the information management system that we have developed [12], two issues were encountered. First, the behaviour of the agents was only regulated by the quantity of plans and messages that an agent had and was not related to the informational content of the agent's knowledge or to the received data. The agent acted the same way in the case of receiving data that needed a quick spreading and in the case of data for which there is no such requirement. Second, the data was distributed by the same rules, independent of its content or relatedness with any domain, and independent of the interest of the agents. The two issues are both related to the lack of context awareness.

The solution for the first issue is the integration of a measure of urgency associated with the data, representing how quickly the data should be handled and how important it is for it to be passed on. This measure was called *pressure*. Higher pressure means more urgency and the data should spread more and faster throughout the system. In our example, high pressure may be associated with very important results, or with announcements of great urgency. Lower pressure is associated with information that is only meant to spread slower, in a more limited area.

Pressure is also associated with requests for data. The higher the pressure, the more urgent the request for data is and the system should react more quickly and provide the necessary data to the requesting user. Lower pressure means that the user does not expect the data to be available immediately and the system is allowed to react less promptly.

The global pressure of all facts in the knowledge base represents the pressure on the agent. As discussed later on, the agent will act differently when it is under pressure and when it is relaxed.

The second issue relates to the definition of *interest*, measured according to some *domains* of interest. In this paper, three generic domains of interest are considered, named A, B and C.

Each piece of data is associated with a measure of interest, comprising the amount of relatedness between the data and each domain of interest. For example, data that is in domain A but also has a degree of relationship of 0.2 with domain C will have a measure of interest represented as $\langle A : 1.0,\ C : 0.2\rangle$. We will call this *data-interest*.

Each agent has a measure of its interest towards the three domains, calculated and adjusted according to the data that the agent has and the data that was produced and/or requested by its associated user. We will call this *agent-interest*. Also, each belief that an agent has about some data or some other agent is associated with an unidimensional measure of interest that represents how interesting that fact is to the agent. We will call this *fact-interest*. The representation of facts is discussed later on.

Considering the multi-dimensional (in our generic case, three-dimensional) measures of interest as vectors, the interestingness of a particular belief is calculated as a function of the norm of the difference between the vectors of interest associated with the data and with the agent. The fact-interest is normalized to be in the interval [0, 1].

In more detail, the fact-interest of a belief F about some data D is calculated in function of the data-interest of data D, that is, say, $\langle di_1 \dots di_n\rangle$ and the agent-interest of agent A having that fact in its knowledge base, say $\langle ai_1 \dots ai_n\rangle$. The fact-interest

– let it be noted fi – represents the similarity between the two, normalized to be between 0 and 1. We desired that the similarity be greater when the two vectors are closer, but also we wanted it to not evolve linear with the difference between the vectors, but make large differences between components count more, i.e. reduce the similarity. Therefore, the root mean square of the individual component differences was used, i.e. $sqrt(\sum(ai_j - di_j)^2/n)$, where n is the number of domains of interest. This value was 0 for full similarity and 1 for no similarity, therefore the similarity is calculated as the value above subtracted from 1.

For example, if an agent interested in domains A (more) and C (less) – agent-interest is $\langle A : 0.9,\ C : 0.3\rangle$ – learns (from another agent) a fact about data D – data-interest $\langle A : 0.1,\ B : 0.2,\ C : 0.9\rangle$ – the resulting fact-interest associated to the fact in the agent's knowledge base will be the equal to $1 - ||\langle 0.9, 0.0, 0.3\rangle - \langle 0.1, 0.2, 0.9\rangle||/\sqrt{3} = 0.41$ (the calculus, although not using the exact formula above, leads to the same result). The resulting interest will be moderate, because the agent is only moderately interested in domain C, that the data is mostly related to.

Context is most times related to space. In most cases, the farther the user from the space providing an information, the less the information is relevant to the user. Therefore in the model presented in this paper, pressure and interest are related to distance. Pressure decreases as information is shared farther, and interest will be lower for facts that refer to agents at a longer distance.

5. System Design. The system is conceived as a two-dimensional space in which cognitive agents are placed and each agent has a number of acquaintances with which it can communicate directly. Experiments were carried out using agents placed in a rectangular grid, each agent communicating directly with its 8 neighbours. In the future, the system will be improved to support more flexible and dynamical topologies. Right now, we have focused on how to design agents' behaviour so that the desired global result was obtained.

The multi-agent system has been implemented as a Java application, in which agents run in a pseudo-parallel manner: at each global time step, agent perform one internal step, each at a time, in order.

5.1. Agent Beliefs. Agents are cognitive and implement the Belief - Desire - Intention model. The beliefs of an agent are held in its knowledge base and are represented using three structures: *Data*, *Goal* and *Fact*. A *Data* structure represents information about one piece of data: the content, the size and the interest relative to the domains. The content of data must not be mistaken for the beliefs about the data, as we have said earlier. A *Goal* structure contains information about an objective, or goal, of an agent. One example of an agent's goal (and the most used goal in the system) is "need to get data D", represented as $\langle Get, D\rangle$, where D is a *Data* structure. However, what agents hold in their knowledge bases and what they send to each other are *Fact* structures. A *Fact* is a tuple that can have one of the following forms:

$$\langle Agent, Data, pressure, interest\rangle$$
$$\langle Agent, Goal, pressure, interest\rangle$$
$$\langle Agent, Fact, pressure, interest\rangle$$

These three types of facts represent associations between the specified elements. The first type of fact means that "*Agent* has *Data*"; the second – "*Agent* has *Goal* objective"; the third type means that "*Agent* knows *Fact*". All three types contain an indication of *pressure* on the fact. The pressure shows how important this fact is for the *Agent*. In the first and third cases, it shows how quickly the fact should be

processed and / or sent to the neighbours. In the second case, it shows how important it is for the agent to fulfill the goal. The interest is the *fact-interest* associated with the fact, as specified in section 4.

It is very important to note the use of the third type of fact, i.e. agents may know facts about what other agents know. This kind of knowledge may also be passed to other agents, if it is of interest. In order for the agent not to be overwhelmed by irrelevant facts about distant agents, the knowledge base is updated by removing the facts that the agent is least interested in and that have lower pressure. This is also useful if agents reside on devices with lower capabilities.

Another remark about the third type of fact must be made. As this type is recursive, it is obvious that the last nested fact must be of one of the other two types. The knowledge base of an agent must carefully check the contained facts so that no circularity appears.

5.2. Agent Goals. The choice of individual agent goals is particularly important for our approach, as the system is meant to exhibit global, emergent properties as a result of local goals and local interaction between agents. As stated before, the fact that the agents only have local goals and a local image of their environment means that they need less computing power and may be deployed on smaller devices.

The global goal of the system is to integrate new information coming from the users and to make it available to other users that need it or might find it interesting. Therefore, at the local level, agents should interact with their neighbours in order to exchange data and collaborate. With this goal in mind, the agents were designed with human-inspired features that would help them share relevant information as well as prevent them from being overwhelmed by facts or by the data itself.

The agents have been given the following goals (no particular order was used; in parenthesis – the name of the type in the internal representation):

- share data (according to context) (*INFORM*);
- keep some storage available in case data is injected from the exterior (by the human user) (*FREE*);
- get interesting data from other agents (*GET*);
- fulfill external (human user) requests (*EXT*);
- help other agents fulfill their goals (*SHARED*).

Goals are represented as pairs of the form $\langle Goal\_type, Object \rangle$, where $Object$ may be the description of a piece of data (its identifier), in the case of *GET*, or a fact about a piece of data, in the case of *INFORM*.

Goals are chosen according to their importance, which is represented by means of *pressure*. In most cases, the pressure is taken from the fact associated with the goal: for external requests and for helping other agents, it is a fact of the form $\langle Agent, Goal \rangle$ that is received from a neighbour or from the exterior; for getting interesting data, it is (possibly nested into other facts) a fact of the form $\langle Agent, Data \rangle$. The goal of keeping storage available has a pressure which varies exponentially with the amount of used capacity over 75%, as it is essential to have some free storage available at any time so that the user can insert new data.

5.3. Agent Plans. The plans contain the actions that an agent must perform in order to fulfill its goals. In the system that we have designed, an agent may have several ongoing plans at the same time, and may also have a set of *waiting* plans, i.e. their completion depends on an external event, like knowledge or data expected to come from another agent, as a result of a request.

The plans that an agent may build are formed of several basic actions, like:

- send data to a neighbour, following a request;
- request data from a neighbour, if the agent knows that the neighbour has that data;
- inform a neighbour of a fact that the agent believes relevant to the neighbour. This will be done only if the agent believes that the neighbour does not already know that fact. The fact may express that an agent has certain data, or that an agent has a certain goal;
- discard some data, according to its relevance to the agent's domain of interest and according to the recent frequency of requests (external or not) that have been made for that data.

5.4. Agent Behaviour. The behaviour of the agents is fairly normal for a BDI architecture, but it has some particularities that make them suitable for the required application.

The stages that an agent goes through during a step of the system's evolution are:

- **Receive data from and send data to** neighbour agents. Data is transmitted only as a response to previous requests. These operations are simple and need no reasoning, and that is why they are handled separately and before any other decision is made.
- **Revise beliefs**. This is done based on information received from the other agents. Duplicate or circular facts are found and removed. The interest and pressure of new facts are computed. An important thing to note here is that the pressure of received facts is decreased, so that the pressure of a fact diminishes with each step it takes farther from its source.
- **Check ongoing or waiting plans** for completion (was the goal achieved?) and test whether they have become impossible. In the case of completion the plan is discarded and the pressure of the corresponding facts is cancelled. The interest towards the facts, on the other hand, remains constant.
- **Make plans**. Take the goal with the highest current importance (pressure). If there is no plan for it, make a plan composed of the actions needed to be taken and put it in the list of ongoing plans.
- **Execute plans**. Take the plan associated with the most important goal and execute its next action. Each executed action reduces the pressure of the associated goal with some amount. If there are no more actions to be performed, move the plan to the list of waiting plans. It will be checked for completion in the next cycle.
- **Fade memory** of all facts in the knowledge base. Pressure of all facts is faded. Facts in which the agent has no interest or that have low pressure may be discarded ("forgotten"). This step is necessary in order to avoid overwhelming the agent with useless facts and too old concerns.
- **Revise pressure and interest**. The pressure on the agent is recalculated according to the facts in its knowledge base, as a weighted mean of the pressures of the individual facts, giving more importance to high-pressure facts. The interest is updated as well, according to the pieces of data that the agent holds and according to its recent activity (the resulting interest is also calculated as a mean).

As it is also the case for natural systems, the behaviour of the agent is greatly influenced by the pressure that presses upon it. Besides the instantaneous pressure, the agent's state is also described by a lower and a higher limit for the pressure. If

the pressure is between the two limits, activity is considered normal. If pressure is lower than the lower limit, the agent is considered "relaxed". If the pressure is above the higher limit, the agent is considered "stressed".

The more "stressed" the agent is, the more it will focus on completing its most pressing plans. New knowledge and data acquisition will be reduced to a minimum.

The lower and higher pressure limits are not fixed, and they are adjusted in time. If the pressure on the agent continues to be high for a long time, the two limits will rise and the agent will start to consider its condition as a "normal" one.

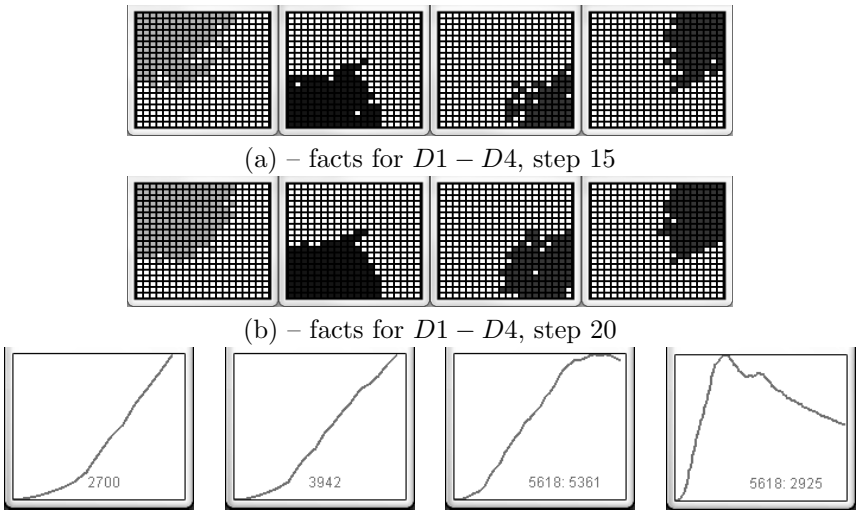
6. Experiment. There were many experiments carried out in the study of context-aware emergent behaviour. Experiments were performed using a system having 400 agents. The scenario we will present is generic, but fits a great number of real-life situations. It is focused on observing how the facts about data (inserted in single instances, into individual agents, therefore not in multiple copies) spread through the multi-agent system. In this spreading, two things were observed: the speed in which facts spread, the coverage they reach and the particular area (or areas) into which they spread.

The system was implemented as a Java application. The execution of the agents is synchronous – at any moment of time all agents are executing the n^{th} step of their evolution. Besides the objectives stated above, one other objective was to make the system run as fast as possible, making the behaviour of individual agents more efficient. This also lead to lower simulation times and the possibility to perform more experiments. A typical experiment lasted 200-250 steps in the multi-agent system's time and about 15 seconds in real time on a machine with an Intel Core 2 Duo 3.33GHz CPU, with 2GB of RAM memory.

6.1. Monitoring Tools. The behaviour of complex systems is extremely difficult to observe in simple graphical representations and the study of the system's behaviour means, most times, the minute observation of the activity log of each agent. This is why a simple scenario was used, one that makes the evolution of agent states clearer.

This is also the reason why some visualization tools have been developed. The graphs that were used are of three types:

- data-fact distribution – associated with a certain piece of data; the graphic is a two-dimensional grid representation of the system where the color of a cell shows the existence of facts regarding the data in the knowledge base of the agent in that cell (e.g. any graphic from Figure 6.1 (a) and (b)).
- pressure distribution 3D – a three-dimensional surface showing the amount of pressure (z-axis) on each agent in the system (e.g. Figure 6.5 (a) - (e)).
- pressure distribution 2D – a two-dimensional grid representation showing the amount of pressure, as intensity of colour, on each agent in the system (e.g. Figure 6.4 (a) - (e)).
- interest distribution – a two-dimensional grid representation of the system where the colour of a cell shows the amount of interest that an agent has for a certain domain (e.g. Figure 6.2 (a)). The colour of each cell is generated by direct conversion of the agent-interest (which has three components in the interval [0,1]) into an RGB code.
- fact number graph – associated with a certain piece of data; a graph that shows the evolution of the total number of facts regarding the data (calculated as a sum over all the agents) in function of the time step (e.g. Figure 6.1 (c) and Figure 6.3 (b)). The evolution is shown for the whole interval between



(a) – facts for $D1 - D4$, step 15

(b) – facts for $D1 - D4$, step 20

step 15 step 20 step 35 step 100

(c) – fact number for $D1$

FIG. 6.1. *Distributions for facts regarding $D1$ to $D4$, at steps 15 (a) and 20 (b); number of facts regarding $D1$ at several moments of time. Graph scaling is explained in Section 6.1.*

the start of the system and the current time. The graphs are always scaled to fit the window, but the first number that accompanies the graph show the maximum value of the graph (the minimum is always 0). If there is also a second number, that shows the current value, in case it is lower than the maximum.

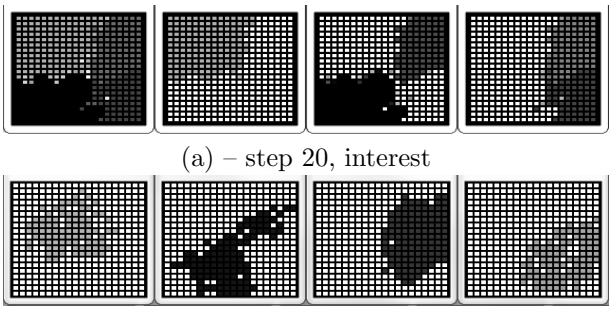
6.2. Scenario. As said, the scenario was designed so that it would emphasize the context-ware behaviour, i.e. the influence of pressure and interest on the spreading of facts through the system.

At system start (time 0), the agents in the system have no defined interest and are under no pressure. In the first phase, 4 pieces of data are inserted into the agents in each corner of the grid. They are named $D1 - D4$. The first three relate each to 1 domain of interest ($A - C$) and $D4$ is strongly and equally related to both domains B and C. They all have moderate pressure, except for $D1$ which has higher pressure than the others. The system is given time to stabilize (20 steps) and then 4 more pieces of data are inserted into the system, into agents in the center of the grid. These are named $D5 - D8$ and are related to domains A, B, C and both A and C, respectively. After some time (after 20 more steps, at time step 40), a new piece of data – $D9$ – that is not related to any domain in particular, but has a very high pressure, is inserted into one side of the grid.

The expected results are the following:

- facts with higher pressure spread more, and faster
- facts spread more in areas where agents are interested in them

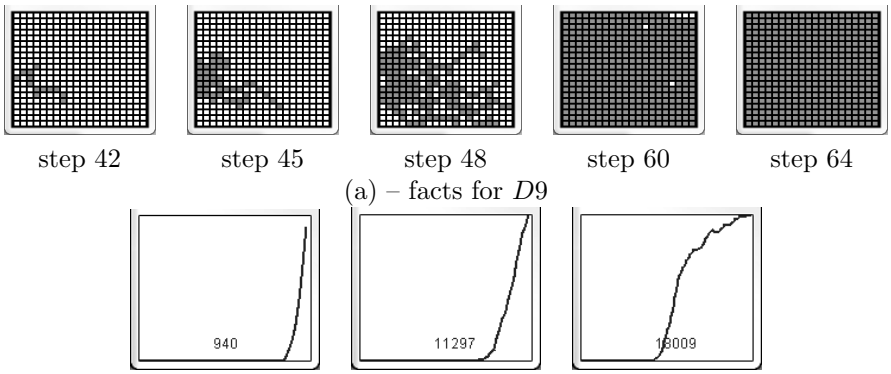
6.3. Results and Discussion. Relevant results that have been obtained after experimentation are displayed graphically in the Figure of this section. We will comment on some interesting phenomena that can be observed, with regard to the two measures of context-awareness that were introduced.



(a) – step 20, interest

(b) – step 35, facts for $D5 - D8$

FIG. 6.2. *(a) Agent interest for domains A, B and C. In the first image interest in all domain is represented. Next three images represent the interest for one domain of interest each, respectively.*



step 42 step 45 step 48 step 60 step 64

(a) – facts for $D9$

steps 0-48 steps 0-60 steps 0-100

(b) – fact number for $D9$

FIG. 6.3. *(a) Evolution of the distribution of facts regarding D9; (b) graphs of the number of facts regarding D9. Graph scaling is explained in Section 6.1*

Pressure indicates the importance and urgency of a piece of data or fact, and, in the case of agents, the need for quick reaction. This makes facts with higher fact-pressure spread quicker and over larger areas. This can be observed in several cases in the figures.

First, in Figure 6.1 (a) and (b) one can notice that facts for $D1$ spread more, and faster, as they have higher pressure. For comparison, at step 20 there are almost 4000 facts about $D1$ in the system, which is 36% of all the facts. As the agents have no initial interest the interest is formed after the spreading of the 4 initial facts. Since $D1$ has the highest pressure, there will be more agents with interest for domain A, as seen in Figure 6.2 (a).

Second, when $D9$ is inserted into the system, having maximum pressure, it spreads visibly fast. It takes only 14 time steps for the information to be known by every single agent in the system (see Figure 6.3 (a)). This shows that if urgent information is to be inserted, it will indeed spread very fast. This very quick reaction is also due to the agents adjusting behaviour under high pressure. The number of facts regarding $D9$ rises exponentially in the beginning (see Figure 6.3 (b)) and the facts reach a long distance from source very quickly, as compared to the evolution of other facts inserted into the system.

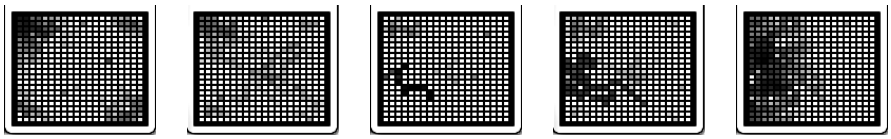


FIG. 6.4. *Pressure on agents in the system, represented as shades of gray, at different moments of time.*

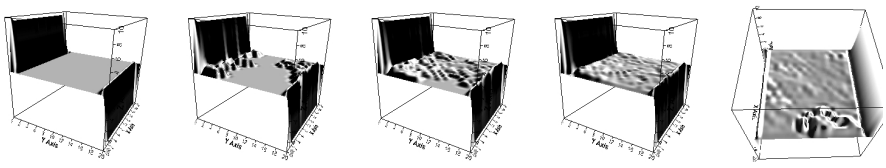


FIG. 6.5. *Pressure distributions: (a) initial state (step 0); (b) immediately after inserting new pieces of data into the system; (c), (d) stabilization of pressure across the whole system; (e) a "wave" of pressure at the insertion of a new piece of data.*

Also regarding pressure, Figure 6.4 shows the pressure on agents at different moments of time. Notice, for example, that in the beginning there is higher pressure in the corners, where new data has been inserted. At steps 42 and 45 the pressed agents are the one holding facts about $D9$. And, due to the effect of $D9$, the whole left side of the system is under high pressure, as it is closer to the origin of $D9$.

A more intuitive perspective upon pressure is presented in Figure 6.5, where one can observe how pressure rises in not necessarily contiguous spots and then becomes more uniform as facts spread in the system. When new data is inserted, pressure rises again (Figure 6.5 (e)). Observe the "wave" of pressure. Behind the "wave", pressure is high, but more uniform, and decreasing. On the edge of the "wave", pressure is non uniform, as there are still some agents that haven't received the facts yet.

As pressure of the facts fades, they may be forgotten, especially if interest in them is lower. Therefore, after the initial surge, the number of facts starts decreasing towards stabilization (see Figure 6.1 (c)). In our experiment, the number of facts for each data stabilized at a mean of 3 to 5 facts per agent.

Interest controls the direction of information exchange and the area of spread, not in terms of size, but in terms of the previous experience of agents. That is, agents that have exchanged more facts related to a certain domain of interest will become more interested in that domain. Positive feedback will lead to the formation of groups of agents interested in a certain domain.

In support of this statement, one can observe Figure 6.2. The most eloquent distribution is that of facts regarding $D6$ – which is related to domain B, that follows the interest of agents in domain B, starting from the center and moving towards the two corners of the grid. On the other hand, as $D9$ is not related to any particular domain of interest, it will spread uniformly through the system.

Regarding the number of facts, although it may seem as fairly high, one should remember that the representation of facts is very simple and individual facts take very little space. In the future, we will work further towards designing better forgetting strategies that will reduce the number of facts even further.

The experiments have shown that exchanging information based on context leads to good results, even if the representation of context is primitive.

7. Conclusion. One of the essential elements required for the implementation of ambient intelligence is a decentralized, self-organizing exchange of information between devices of reduced storage and processing capacity.

This paper describes a system for information exchange that includes the key concept of context-awareness, by using a simple but effective representation of context, comprised of two elements: pressure and interest. The system has been tested on a scenario involving a large number of agents and experiments have shown that the system, as a whole, is indeed considering context in its behaviour.

The paper does not refer to a specific application. The system and its behaviour have been kept as generic as possible, trying to deal with the general problem of context-aware information exchange.

REFERENCES

[1] J. Augusto and P. McCullagh, *Ambient intelligence: Concepts and applications*, Computer Science and Information Systems/ComSIS, 4 (2007), pp. 1–26.
[2] C. Bourjot, V. Chevrier, and V. Thomas, *A new swarm mechanism based on social spiders colonies: From web weaving to region detection*, Web Intelligence and Agent Systems, 1 (2003), pp. 47–64.
[3] G. Cabri, L. Ferrari, L. Leonardi, and F. Zambonelli, *The LAICA project: Supporting ambient intelligence via agents and ad-hoc middleware*, Proceedings of WETICE 2005, 14th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, 5 (2005), pp. 39–46.
[4] G. Chen and D. Kotz, *A survey of context-aware mobile computing research*, tech. report, Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.
[5] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J. Burgelman, *Istag scenarios for ambient intelligence in 2010*, tech. report, Office for Official Publications of the European Communities, 2001.
[6] M. Gleizes, V. Camps, and P. Glize, *A theory of emergent computation based on cooperative self-organization for adaptive artificial systems*, in Fourth European Congress of Systems Science, 1999.
[7] D. Hales and B. Edmonds, *Evolving social rationality for MAS using "tags"*, Proceedings of the second international joint conference on Autonomous agents and multiagent systems, (2003), pp. 497–503.
[8] B. Johanson, A. Fox, and T. Winograd, *The interactive workspaces project: Experiences with ubiquitous computing rooms*, IEEE pervasive computing, (2002), pp. 67–74.
[9] T. Lech and L. Wienhofen, *AmbieAgents: a scalable infrastructure for mobile and context-aware information services*, Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, (2005), p. 631.
[10] J.-P. Mano, C. Bourjot, G. Leopardo, and P. Glize, *Bio-inspired mechanisms for artificial self-organised systems*, Special Issue: Hot Topics in European Agent Research II Guest Editors: Andrea Omicini, 30 (2006), pp. 55–62.
[11] A. Olaru and A. M. Florea, *Emergence in cognitive multi-agent systems*, Proc. of CSCS17, 17th International Conference on Control Systems and Computer Science, (2009), pp. 515–522.
[12] A. Olaru, C. Gratie, and A. M. Florea, *Emergent properties for data distribution in a cognitive mas*, Proc. of IDC 2009, 3rd International Symposium on Intelligent Distributed Computing, (2009), pp. 151–159.
[13] G. Picard, *Cooperative agent model instantiation to collective robotics*, in Engineering Societies in the Agents World V: 5th International Workshop, ESAW 2004, Toulouse, France, October 20-22, 2004: Revised Selected and Invited Papers, Springer, 2005.
[14] C. Ramos, J. Augusto, and D. Shapiro, *Ambient intelligence - the next step for artificial intelligence*, IEEE Intelligent Systems, (2008), pp. 15–18.
[15] N. Sadeh, F. Gandon, and O. Kwon, *Ambient intelligence: The MyCampus experience*, 2005.
[16] I. Satoh, *Mobile agents for ambient intelligence*, Proceedings of MMAS, (2004), pp. 187–201.

[17] A. E. F. Seghrouchni, *Intelligence ambiante, les defis scientifiques.* presentation, Colloque Intelligence Ambiante, Forum Atena, december 2008.

[18] G. D. M. Serugendo, M.-P. Gleizes, and A. Karageorgos., *Self-organization and emergence in MAS: An overview*, Informatica, 30 (2006), pp. 45–54.

[19] J. Viterbo, L. Mazuel, Y. Charif, M. Endler, N. Sabouret, K. Breitman, A. El Fallah Seghrouchni, and J. Briot, *Ambient intelligence: Management of distributed and heterogeneous context knowledge*, CRC Studies in Informatics Series. Chapman & Hall, (2008), pp. 1–44.

[20] M. Weiser, *Some computer science issues in ubiquitous computing*, Communications - ACM, (1993), pp. 74–87.

[21] F. Zambonelli, M. Gleizes, M. Mamei, and R. Tolksdorf, *Spray computers: Frontiers of self-organization for pervasive computing*, Proceedings of the 13th IEEE Int'l Workshops on Enabling Technologies, WETICE, (2004), pp. 403–408.