Shout and Act: an Algorithm for Digital Objects Preservation Inspired from Rescue Robots

Josep Lluis de la Rosa
EASY Innovation Center
University of Girona
Campus de Montilivi, E17071
Girona, Catalonia (EU)
+34 972 418854
peplluis@eia.udg.edu

Albert Trias
EASY Innovation Center
University of Girona
Campus de Montilivi, E17071
Girona, Catalonia (EU)
+34 972 418478
albert.trias@gmail.com

Silvana Aciar
EASY Innovation Center
University of Girona
Campus de Montilivi, E17071
Girona, Catalonia (EU)
+34 972 418478
saciar@eia.udg.edu

Esteve del Acebo
EASY Innovation Center
University of Girona
Campus de Montilivi, E17071
Girona, Catalonia (EU)
+34 972 418833
acebo@ima.udg.edu

Hugo Quisbert
Luleå University of Technology
University Area, 97187, Luleå
(Sweden)
+46 920 491810
Hugo.Quisbert@ltu.se

ABSTRACT

We adapt the Shout and Act algorithm to Digital Objects Preservation where agents explore file systems looking for digital objects to be preserved (victims). When they find something they "shout" so that agent mates can hear it. The louder the shout, the urgent or most important the finding is. Louder shouts can also refer to closeness. We perform several experiments to show that this system works very scalably, showing that heterogeneous teams of agents outperform homogeneous ones over a wide range of tasks complexity. The target at-risk documents are MS Office documents (including an RTF file) with Excel content or in Excel format. Thus, an interesting conclusion from the experiments is that fewer heterogeneous (varying skills) agents can equal the performance of many homogeneous (combined super-skilled) agents, implying significant performance increases with lower overall cost growth. Our results impact the design of Digital Objects Preservation teams: a properly designed combination of heterogeneous teams is cheaper and more scalable when confronted with uncertain maps of digital objects that need to be preserved. A cost pyramid is proposed for engineers to use for modeling the most effective agent combinations.

Keywords

Digital Preservation, Multi-Agent System, Rescue

1. INTRODUCTION

Shout and Act (S&A) is an evolution of Bar Systems, a family of very simple algorithms for different classes of complex optimization problems in static and dynamic environments by means of reactive multi agent systems. S&A systems are loosely inspired from the behavior shown by a staff of bartenders when serving drinks to customers in a bar or pub. The difference is that not only customers but also the bartenders call (shout) for help. Agents explore a plot of unknown land; when they find a potential victim by means of a sensor, they "shout", emitting a radio or light signal so that possible partners can hear or see it. The louder the shout, the more important the finding; loudness could also refer to proximity. This is a simple rule that works if teams of agents are prepared to operate in such a collective way. In this paper we are going to see how it can be applied to digital preservation.

The term Swarm Intelligence, which has garnered much attention, arose in the late 1990s in the Artificial Intelligence, Robotics, Artificial Life, and Distributed Problem Solving communities, inspired by the observation of social insect colonies. A commonly accepted definition of it is the property of a system whereby the collective behaviors of (unsophisticated) agents interacting locally with their environment cause coherent functional global patterns to emerge [25][26]. The chief paradigm of Swarm Intelligence is an ant colony. In it, individual ant behavior is controlled by a small set of very simple rules, but their interactions (also very simple) with the environment allow them to solve complex problems such as finding the shortest path from one point to another. Ant colonies (and we could say the same about human beings) are intelligent systems with great problem-solving capabilities, formed from a quantity of relatively independent and very simple subsystems that do not show individual intelligence. This is the "many dummies make a genius" phenomenon of emergent intelligence.

Swarm Intelligence problem-solving techniques present several advantages over more traditional ones. On one hand, they are cheap, simple and robust; on the other hand, they provide a basis for exploring collective (or distributed) problem-solving without centralized control or the provision of a global model. Over the last few years, they have been used in the resolution of a very heterogeneous class of problems. Two of the most successful Swarm Intelligence techniques currently in use are Ant Colony Optimization [1] and Particle Swarm Optimization [2]. Ant

---

InDP'09, June 19, 2009, Austin, TX, USA. Copyright 2009.

Colony Optimization techniques, also known as Ant Systems, are based on ants' foraging behavior, and have been applied to problems ranging from the determination of minimal paths in TSP-like problems to network traffic rerouting in busy telecommunications systems. Particle Swarm Optimization techniques, inspired from the way a flock of birds or a school of fish moves, are general global minimization techniques that deal with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Other Swarm Intelligence applications include collective robotics, agent navigation, planetary mapping, streamlining of assembly lines in factories, coordinated agentic transport, and banking data analysis. For more details on self-organization and Swarm Intelligence theory and applications, please refer to [3] [4] [5] [6] [7] and [8].

S&A is a type of Bar System, reactive multi-agent systems whose behavior is loosely inspired by that of a staff of bartenders, and can be enclosed in the broader class of Swarm Intelligence systems. We will explore its application to Digital Objects Preservation. The preservation of digital objects is a new issue that comes up with the continuous growth and advance of information, that is created and stored digitally. Few solutions have been given to solve how this information will be accessed in the future, even within the next decade or so [10]. Even if the information itself survives over time, the hardware and the software to access it may not. As a result "rescue information" is required to ensure ongoing access to digital preservation for as long as it is requires and for whatever. In this paper is presented how digital objects (text files, photography, audio, etc) can be "rescued" for Long-Term Digital Preservation using solutions inspired from rescue robots and swarm intelligence like algorithms.

This paper is organized as follows: section 2 will describe the Digital Objects Preservation problem and the type of algorithms it needs; the concept of Shout and Act as a type of Bar System will be presented and formalized in section 3; in sections 4 and 5 we will present the results of preservation as rescue experiments, with several configurations of agents and varying complexity of preservation grids; in section 6, we will derive some principles for the design of Digital Preservation teams; and in section 7 we present our conclusions.

2. THE PROBLEM OF PRESERVING DIGITAL OBJECTS

Hardware and software obsolescence is jeopardizing the availability and accessibility to digitally recorded information. Moreover, large amounts of digital information have been created and this growth continues exponentially making digital preservation complex. Digital Preservation aims to alleviate threats caused by inherited digital obsolescence and the rapid growth of information. [10] However, many of preservation activities are still made by hand and labor intensive [11] implying that the exponential growth of digital content has made preservation a critical and troublesome problem [12]. Preservation activities thus need to be automated as much as possible.

The elements in a preservation realm are: a) the objects to be preserved, b) the tools for preservation (e.g. software, infrastructure or platform) and c) curators or the preservation organization. [13] The technical side of a preservation realm, the preservation environment exhibits characteristics of being uncertain and complex [14]. Furthermore, digital preservation is concentrated to preservation organizations (such as archives and libraries) and consequently the control of the material to be preserved is made by the people within the preservation organisation. However there is also a need of collaborative infrastructures in digital preservation [15].

2.1 Motivation of Using Agents in the Digital Preservation Realm

A high degree of automation is an issue in Artificial Intelligence (AI). A modern approach to AI has crystallised the Intelligent Agent design metaphor for Information Systems design. According to Wooldridge [17], such an approach based on this metaphor is appropriate when:

1. The environment is open, or at least highly dynamic, uncertain or complex.

2. Data, control or expertise is distributed. In some environments, the distribution of data, control or expertise means that a centralized solution is at best difficult and at worst impossible.

3. Legacy systems are inherited. Inheriting software that is technologically obsolete but functionally essential for an organization leads to difficulties accessing the information created with that software.

Agent technology thus has potential in approaching sustainable solutions for the problems outlined in the above section. Digital Preservation can be approached as a technical issue using Agents. Thanks to current technological advances, the field of Search and Rescue is one of the most demanding applications challenging artificial intelligence in the 21st century, as shown in table I. But one of them is the most important [23]: the number of digital objects to be preserved in growing exponentially, and we need algorithms and solutions that could offer linear complexity, better if they offer 0 complexity. Agents automate routine tasks in digital preservation as appraisal, pre-ingest, ingest, etc, as an alternative of much less complexity than current centralized approaches, based on OAIS approach. Furthermore, routine tasks are not only tedious but time and resource consuming. It is quite easy to convert 100 MS word documents to pdf-format. Ocular control over the quality of conversion, access and understandability can easily be made on a handful of samples. Although it might take 10-15 minutes it is quite different that converting and controlling 100,000 or one million documents.

The OAIS model has had a defining influence on the development of digital preservation methods. The model was proposed by the Consultative Committee for Space Data System (CCSDS), and connected to the National Aeronautics and Space Administration (NASA) in the USA [24]. The OAIS model is a conceptual framework for an archival system dedicated to preserving and maintaining access to digital information. The model is compounded by four major components:

- A *producer* (person or machine) which produces the information to be preserved;
- A *manager* who sets up a policy or framework for preservation;
- An *archive* (an implementation of the OAIS model); and
- A *consumer* (person or machine) who interacts with the archive in order to retrieve information.

The figure below (Figure 1) shows the conceptual model of the OAIS Reference model. The model demonstrates a linear flow from the producer to the consumer and delimited “stations”, the high level functions of the model, (ingest, storage, data management, access, administration and preservation planning). How the material to be preserved is handled in these “stations” is crucial issue that has to be preceded by careful planning. There is no single exhaustive solution. The OAIS model resembles the paper-based archival approach in its linear flow; the digital real offers an opportunity of a non-linear approach to preservation activities. A successful preservation of the Digital Objects to preserve, demands flexible planning.

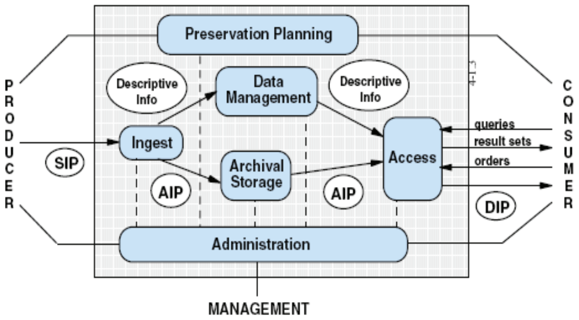


Figure 1 - The OAIS Reference Model

As shown in table 1, there are many aspects that determine the difficulty of Digital Objects preservations: planning is done in real time, and heterogeneous multiagent systems work in dynamically changing and even hostile environments in which new and complex scenarios constantly appear.

Table 1. Comparison of AI Applications (adapted from [18])

Characteristic	Digital Objects Preservation	Robo-Rescue	Robo-Soccer	Chess	Deep Space Probe
Number of agents	**> 100,000**	> 1,000	11+	1	<10
Homogeneity of agents	**Hetero-geneous**	Hetero-geneous	Homo-geneous	Homo-geneous	Hetero-geneous
Control	**Hierarchical /Distributed**	Hierarchical /Distributed	Distributed	Central	Central
Similarity to reality	**High**	High	High	No problem	Total
Situation	**Diverse**	Diverse	Simple	Simple	Diverse
Actions	**Varied**	Varied	Simple	Simple	Varied
Information gathered	**Diverse**	Diverse	Simple	Simple	Simple
Representation	**Hybrid**	Hybrid	Non-symbolic	Symbolic	Hybrid
Emerging collaboration	**Important**	Important	No	No	Average
Real-time	**days-years**	s-min	Ms	m-h	ms-s
Access to information (inter-agent comm.)	**Not perfect**	Very bad	Quite good	Total	Very good
Resources	**Highly heterogeneous**	Highly heterogeneous	Hetero-geneous	Homo-geneous	Sub-heterogeneous
Logistics	**Important**	Important	Irrelevant	Irrelevant	Impossible
Short-term planning	**Highly Important**	Important	Important	+ important	+ important
Long-term planning	**Highly Important**	Important	Not important	+ important	Important
Scenarios	**Complex**	Complex	Formations	Openings	Modes
Hostility	**Environment**	Environment	Opponent	Opponent	Faults

Softbots interact with totally alien cognitive agents, for example, human beings, with the need for monitoring and constant optimization of scarce resources.

2.2 Approaches of Digital Preservation using Agents

We conceive three approaches of digital preservation using agents:

- First, agents work like recommenders. For example agents can work together for more accurate appraisal of the digital objects to be saved. This approach will help users share their points of view and solutions in a Web 3.0 way (3.0 as the automation of web 2.0 approaches to digital preservation) providing with easier tools for preservation. An example of this type of agents is Giulia, shown in [23].
- Second, agents work like rescue robots. Agents look for digital objects, the victims, with expected short life (for example, format problems) and try to save them. This is expected to contribute with scalability to the exponentially growing complexity of digital preservation.
- Third, agents are the digital objects to be preserved. Agents look for surviving and compete for being preserved. The log of formats is kept inside the agent, who negotiates with web services or other agents for being itself preserved, and earn credits. This is a native application of agents.

In this paper we will develop the second approach, bearing in mind three perspectives a) that this is an isolated attempt in which we use specific files types to carry out experimentation, b) the experimentation also covers format related problems and c) this paper concentrates in showing algorithmic approaches dealing with file complexity and system scalability.

2.3 Desirable Properties of Digital Preservation Agents

S&A systems are ideal to deal with Digital Preservation as well as RoboRescue requirements, as long as agents are specifically designed to support the communication skills that these systems require. Thus, we must design agents with the following properties from soft-agency:

a) *Agents must be autonomous and proactive*, able to perform desired tasks in unstructured environments without continuous human guidance. A high degree of autonomy is particularly desirable because communication delays and interruptions are unavoidable. They must be able to:
    - Gather information about the environment.
    - Move either all or part of themselves throughout its operating environment without human assistance.
    - Avoid situations harmful to digital objects, property, or themselves.
    - Learn or gain new capabilities, including adjusting strategies for accomplishing tasks and adapting to changing environments.

b) *Agents must be self aware or introspective;* this is a soft-agency property that requires an agent to observe itself, optimize its behavior to meet its goals, and communicate its current limitations to others. Some examples of self-awareness in a system are:

- State diagnosis: credit for the application of web services, available memory, available disk space, and so on.
- Ongoing activities: serving users, appraising or curating objects.
- Choosing actions under external and internal constraints.
- Knowledge and lack of knowledge.
- Purpose, intentions, hopes, fears, likes and dislikes.
- Mental state, e.g. long term goals, and beliefs (hard agency).

c) *Agents must be social, able to communicate directly and indirectly.* They must be able to negotiate among themselves to surmount unforeseen properties of a preservation master plan.

d) *Agents must be heterogeneous*, preferably with complementary skills, expected to work better together than separately, though working as a team sometimes will not be possible.

e) *Agents must know where they are situated, not in an absolute but rather in a relative, social, way*, such as a grid of neighbors in a file system.

In contrast, S&A divides the work of exploring and rescuing digital objects among several types of agents with different sets of capabilities, perception, and communications skills, and performs social mapping while acting. When a team of agents, each very light and with limited perception capabilities, is tasked to explore for digital objects and believe they have detected someone, they start shouting for help. This is very helpful, indeed. They do not necessarily know where they are and they are not necessarily able to return, so they just shout around, hoping that other agents will arrive to help. This avoids the strict requirements of making plans, and is the origin of the shout and act principle: agents put emphasis on action, with simple reactive behavior.

Thus S&A will be used by agents to move around, without a map, in an unknown environment of unknown complexity, while at the same time performing rescue tasks. In fact, the map is the grid of agents placed throughout the environment. Therefore, agents must be designed to work together and to be social, autonomous, heterogeneous, self-aware, and numerous.

3. DESCRIPTION OF THE SHOUT & ACT ALGORITHM

Agents are generally exploring the file system, and will shout when they find an item of interest. Should the shouting agent modulate its volume to indicate distance, or do you simply mean that closer agents are "heard" more loudly? Both two cases apply. The volume of the transmitted shout corresponds to the importance of the finding; the volume received also depends on the distance. The more agents in a position, the more likely it will be for a victim to be at that position. Agents will be of several types, each with heterogeneous skills and capabilities, shouting to (calling) other agents when they detect a potential victim. Those agents with more sensing capabilities tend to be more costly and therefore less numerous. They might follow the hints given by the shouts of inferior types of agents, and shout to summon even more superior types of agents when they themselves detect something.

Let us now describe S&A from the Bar Systems point of view [9]: Anyone who has tried to get served a pint in a bar crowded with customers will have had more than enough time to wonder with boredom about the method used by waiters, if there is one, to decide which customer to pay attention to at each time. Sometimes, one may have to wait a long time before being served, even if shouting for the waiter. Details like the area where the customer is located, his/her sex, whether the waiter knows him/her, and whether the waiter likes the customer's appearance determines to a large extent the way in which orders are served.

Let us examine the situation from the bartenders' point of view: many customers are ordering drinks at once, new ones arrive constantly, and the bartenders have to do all they can to serve them. Of course, they cannot do this in a random way; they have to try to maximize some kind of utility function, which will typically take into account aspects such as average service time (as the time elapsed since the arrival of the customers), average service cost, and average customer/boss satisfaction. They will have to pay attention, then, to facts such as that some of them can prepare certain drinks more quickly or better than others, that the order in which the drinks are served influences the time or the total cost of serving them, and also that moving from one place to another takes time. All of this must be done without forgetting, on one hand, that the order in which orders take place has to be respected as much as possible and, on the other hand, that they have to try to favor the best customers by giving them preferential attention and keeping them waiting for a shorter time.

The problem is not at all trivial, and we proved it to be NP-hard in [9]. Bartenders have to act in a highly dynamic, asynchronous and time-critical environment, and no obvious greedy strategy (such as serving the best customer first, serving the nearest customer first, or first serving the customer who has arrived first) gives good results. Nevertheless, a staff of good bartenders usually can manage to serve customers in such a way that the vast majority of them are, more or less, satisfied. The way they accomplish the task seems to have little to do with any global planning or explicit coordination mechanisms but, arguably, with trying to maximize, every time they choose a customer to serve, some local utility function that takes into account aspects like the importance of the customer, the cost for the waiter of serving him/her, and the time that he/she has been waiting for service.

3.1 Definition

The *Shout and Act* system is a quadruple (E, T, A, F) where:

1. E is a (physical) environment. The state of the environment at each moment is determined by a set of state variables VE. One of those variables is the time. S is the set of all possible states of the environment E, that is, the set of all the possible simultaneous instantiations of the set of state variables VE.

2. $T = \{t_1, t_2, ..., t_M\}$ is a set of tasks to be accomplished by the agents within the environment E. Each task t_i has associated:
    - $pre(t_i)$. A set of preconditions over VE that determine whether task t_i can be done (*introspection* and *heterogeneity*)
    - $imp(t_i)$. A nonnegative real value that reflects the importance of task t_i. (*pro-activity* and *autonomy*)
    - $urg(t_i)$. A function of VE that represents the urgency of task t_i in the current state of the environment E. It will

usually be a no decreasing function of time (*pro-activity* and *autonomy*)

3. $A = \{a_1, a_2, \ldots, a_N\}$ is a set of agents situated in the environment E. Each agent a_i may have different problem-dependent properties (e.g., weight, speed, location, response time, maximum load, perception, and communication skills, all part of the *heterogeneity*). For each agent a_i and each task t_j, $cost(a_i,t_j)$ reflects the cost for agent a_i to execute task t_j in the current state of the environment E. This cost can be divided into two parts: first, the cost for a_i to make the environment fulfill the preconditions of task t_j (this can include the cost of stopping his current task), and then, the cost for a_i to actually execute t_j. If an agent a_i is unable to adapt the environment to the preconditions of task t_j or if it is unable to carry out the task by itself, then $cost(a_i,t_j)$ is defined ∞.
4. $F : S \times A \times T \rightarrow \Re$ is the function reflecting the degree to which agents are "attracted" by tasks. Given a state s of the environment, an agent a_i, and a task t_j, $F(s, a_i, t_j)$ must be defined so that it increases with $imp(t_j)$ and $urg(t_j)$ and decreases with $cost(a_i,t_j)$.

In Shout and Act, agents operate concurrently in the environment in an asynchronous manner, eliminating, thus, the typical operation cycles of other Swarm Intelligence systems (Ant Systems, Particle Swarm Optimization Systems, Cellular Automata, etc.). The individual behavior of agents is given by the following algorithm:

Algorithm: Individual agent's behavior

```
procedure ShoutAndActAgent
  repeat
      Find the most attractive free task M
      if the agent is doing M OR trying to fulfill $pre(M)$ then
         Continue doing it
      else
         Stop doing the current task, if any
         if $pre(M)$ holds then
            Start doing M
         else
            Do some action in order to fulfill $pre(M)$
end if
      end if
  until no tasks left
end procedure
```

The crucial step in the algorithm above is the choice of the task that the agent will execute for the next time step. In its simplest form, this can consist of choosing the task that maximizes the attraction function *F*. It can also involve some kind of negotiation between agents and even some kind of local planning. One of the tasks will be to *shout* for help.

It is worth stressing that the algorithm allows the agents to respond in real time to changes in the environment like the appearance of new urgent tasks or the temporal impossibility of fulfilling the set of preconditions of a given task.

3.2 Inter-agent Communication

S&A requires simple communicative skills in the agents to attain the coordinated and self-organized behavior typical of Swarm Intelligence Systems. We can identify three main purposes that communication can serve in order to increase S&A problem solving capabilities:

- *Conflict resolution and negotiation*. The way we defined S&A makes unavoidable the occurrence of conflicting situations in which two or more agents choose the same task to carry out. Lack of communication will lead to a waste of resources because of several agents trying to fulfill the preconditions of the same task, even to the extent that only one of them may carry it out. In such situations, it would be convenient to have some kind of negotiation method, which can be as simple as "the first one to see it goes for it". In the case study in section 4, we will discuss more elaborate negotiation strategies.
- *Perception augmentation*. For those agents with limited perception capabilities (we refer to *capability to perceive the tasks* as, for example, whether they have little skill for appraisal of the need of an object to be preserved or not), communication can allow a agent to transmit information to the others about pending tasks they are not aware of, by means of shouts and pheromones by semantic tags. Let us suppose we want to do exploratory tasks in a vast file system, which, following with the analogy, is a terrain where points of interest must attract and be explored by agents. It would be useful if agents had the ability to share information about the points of interest that they have located during their exploratory activity. In this way, agents would have access to information about the location of points of interest that lie beyond their perceptual capabilities. If agents have the capability to sense where the shout comes from, the rule is simple: go towards the *shout*.
- *Learning*. The attraction function *F* does not need to be fixed in advance. Agents can learn it through their own activity and their communicative interactions with other agents. For example, an agent can discover that a certain kind of task has a high cost and communicate this fact to other agents. Furthermore, agents can even learn from other agent's ways of carrying out new tasks.

On the other hand, it is worth differentiating between two main classes of inter-agent communicative processes:

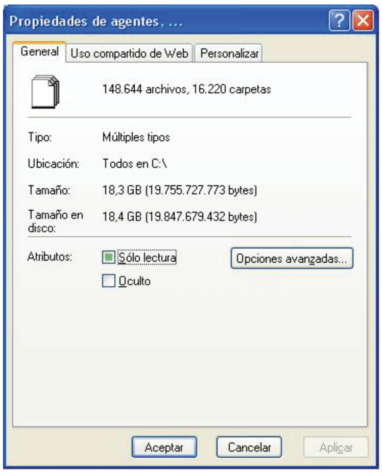
- *Direct*. Agents establish direct communication with each other via some channel and following some kind of protocol.
- *Indirect*. Agents communicate with each other through their actions and shouts, causing changes in the environment. In the S&A framework, this can be seen as Agents generating "communicative tasks" that, when carried out by other agents, increase the information they possess (about the environment and the task set). This is the case for Ant Systems, and this is the case for Bar Systems, and for S&A.

3.3 Coalitions of Agents

Given N agents, the number of coalitions that can be generated is 2^N. Each coalition may have a value that represents how efficiently it can perform tasks [22] . This may be due to differences in capabilities or constraints. As a matter of fact, S&A will apply to a coalition as well as a single agent. The important thing for S&A is that there are diverse behaviors and capabilities that emerge for best adapting to *E*.

4. EXPERIMENTS

Fig 3, shows a rescue scenario with a number of victims v_i at unknown positions in a 2D map representation of a file system. We can see a hierarchical file system like a three-dimensional normal distribution, where the root of the file system is in the centre of the three-dimensional normal. There are more than 100,000 files in more than 15,000 directories. This is an imperfect 2D representation of a 3D information infrastructure that is useful for demonstration of the basic properties of the S&A algorithm.



148.644 archivos, 16.220 carpetas

Tipo: Múltiples tipos

Ubicación: Todos en C:\

Tamaño: 18,3 GB (19.755.727.773 bytes)

Tamaño en disco: 18,4 GB (19.847.679.432 bytes)

Atributos: ☑ Sólo lectura ☐ Oculto

Opciones avanzadas...

Figure 2. A file system to be made of 2DMap of Victims and agents (in Spanish Windows XP)

Also we can superimpose a grid, with the centre square in the normal distribution centre. The value of the normal shows us a heuristic of the connections under this square. It also helps to see a matrix like a file system with links the folders to their brothers, having a similar connectivity.

The structure of the file system is projected onto a grid to give intuitive idea of how agents move and locate the victims, the digital objects to preserve, as shown in Fig. 3.

There are several agents of type A and agents of type B. The two types of agents are designed to detect and confirm the locations of victims.

Repast (REcursive Porous Agent Simulation Toolkit http://repast.sourceforge.net/) [21] was chosen for developing the experiments, as shown in Fig 3. It is an open source freely-available agent-based simulation toolkit specifically designed for social science applications. Repast permits the systematic study of complex system behaviors through controlled and replicable computational experiments. Originally developed by David Sallach and other researchers at the University of Chicago and the Argonne National Laboratory, Repast is now managed by the non-profit volunteer organization ROAD (Repast Organization for Architecture and Development). Repast provides a core collection of classes for the building and running of agent-based simulations and for the collection and display of data through tables, charts, and graphs. A particularly attractive feature of Repast is its ability to integrate GIS (geographical information science) data directly into simulations. It has been released in four versions supporting model development in three different programming languages: (1) RepastJ (Java based); (2) RepastPy (based on the Python scripting language); (3) Repast.Net (implemented in C#, but any .NET language can be used); and (4) RepastS (Repast Simphony, Java based). We use RepastS.

The environment we designed in Repast Simphony consists of a 2D radar screen of 50 x 50 cells within an unlimited world. Agents can get lost by departing from the radar screen. In Repast, agents simulate Victims, Agents, and Shouts.

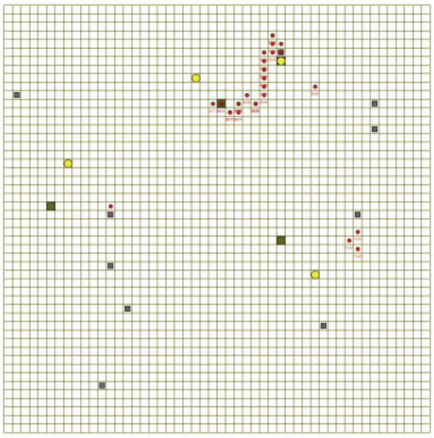


Figure 3. The Radar screen in the grid of REPAST-Simphony. Victims are yellow, agents are squares, and shouts are red dots

4.1 Victims

These represent the digital objects to be rescued. They are placed randomly in the grid (which is the representation of the file system) and are static throughout the simulation. Each Victim has a lifetime, using with a clock that decreases with every tick of the simulation until it is rescued or dies. The lifetime of a Victim, which simulates the severity of its injuries (i.e. its format obsolescence), is one of the factors that influences the magnitude of a shout emitted by an agent that detects him, if the agent is able to determine it. Victims are shown in yellow when waiting to be rescued and green after they have been rescued. If they die, they are removed from the grid, as they might become undetectable for our agents.

So the catastrophe is a change of format for MS excel that requires.xls update to .xlsx. This update is the actual preservation action that avoids the contents of the sheet to be inaccessible in the long term.

The four digital objects are:

1. A MS-**excel** of a budget. High risk of losing information in the format upgrade with wrong calculations if updated.
2. A MS-**word** document with embedded MS-Excel objects of budgets. Moderate risk of losing information, because the embedded objects can be kept visible though it could be updated.
3. A MS-P**owerPoint** document containing embedded MS-Excel objects. Low risk, because it can keep the information of the excel object visible.
4. An **rtf-formatted MS-Excel** with a budget. Very low risk, because the .rtf is designed (!) to be transportable between excel formats.

4.2 Agents

Their goal is to rescue victims or help to do so. There are 2 different types, A and B, which have different capabilities and are initially placed randomly in the grid.

- *A Type* agents are faster and more numerous than B type agents. Their main goal is to detect Victims. In this experiment, they spend most of their time moving randomly in the grid. They can perceive Victims by fast appraisal methods (very simple diagnostic capabilities), and they spawn a Shout in that case. The Shout is implemented by writing a message on a blackboard an several levels in the filesystem. If, after a certain period of time, an A Type agent has not been able to perceive any Victims, it moves to the closest or strongest Shout emitted by another agent.
- *B Type* agents are less numerous and slower than the A Type agents, because they have superior ability to detect, appraise (further complex diagnostic capabilities) and rescue (actually preserve) victims. They follow the Shouts that A Type agents emit. When a B Type agent hears one or more Shouts (by reading them in the blackboard on the root of a given level of a filesystem), it will set the position of the Shout of highest magnitude as the target endpoint, provided that it can find a trajectory. In order to find out if this condition is met, it carries out some *introspection* and then communicates (if possible) with the other agents of its type. Thus, it determines its state (free / assisting a victim / moving to the position of a shout) and distance to the shout, and asks other Type B agents how close they are from the position where the Shout has been emitted. The next Type B agent does the same reasoning as well, deciding if it should call another agent or act by itself. With this, we achieve some basic coordination between the agents, avoiding the situation where two or more agents of equal type are assisting the same victim. Once a Type B agent reaches the position of a Shout, it performs a scan to locate and confirm the expected Victim.

4.3 Shouts

Shouts are emitted by A Type Agent agents whenever they perceive a Victim agent. Its magnitude could be proportional to the severity of the injuries of the Victim agent. Shouts disappear some time after being emitted, and disperse with distance. The more A Type agents are shouting and the stronger the shout, the more likely it is that other B Type or A Type agents will arrive, consequently increasing the probability of detecting and rescuing a Victim.

Shouts are implemented by tagging the document, in a sort of pheromone, to help other agents detect and start appraising. They also can give pheromones in directories of several higher levels to help attract the mates. The instrument to write down all the shouts is a blackboard [23].

5. RESULTS

First, a series of ten runs was carried out to test the stability of the experiments. Then, a series of experiments was performed with increasing complexity, different number of victims, and different number of agents of any type. Finally, three scenarios were tried, including new agents with the highest performance.

5.1 Setup Experiments

These experiments are for estimating the amount of runs that would be necessary for the experiments. We run S&A 10 times, with random initial grid positions of victims and agents (controlled initial conditions), 4 victims, 10 A type agents, and 4 B type agents. The number of ticks (the discrete measure of time) it takes to rescue all the victims is recorded. We then run the algorithm with 10 B agents (we rename them C type to distinguish them from the first experiment) that move randomly like A Type agents in the same 10 initial conditions, and then compare the results. Fig 4 shows that the homogeneous agents performed slower in all cases; the Y axis represents the number of ticks needed to rescue the victims alive, and the X axis is the number of runs.

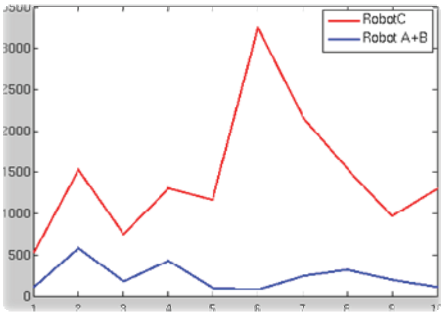


Figure 4. Setup experiments. 10 runs with random initial positions of 4 victims, 10 A agents, 4 B agents, and 14 C agents

As we can see, the performance when using combined A and B type agents is higher than using only C type agents, even though C type agents have the same performance as type B agents and there is a larger number of them (10 against 4). The fact that C agents are not assisted by the faster but unreliable and low-performance A type agents makes them less efficient. Further experiments show that to achieve similar performance, at least 18 C type agents are needed. Taking into account the fact that A type agents, due to their lower performance (in defective capacity both to detect victims and to assist them), are assumed to be of much lower cost than B/C agents, combining A and B type agents appears more suitable than increasing the number of C type agents.

With 10 experiments per each type of robots combination is sufficient for valid conclusions, though there is high variability in the homogeneous robots (C type only).

5.2 Heterogeneous Teams of A+B Agents vs. Homogeneous Teams of C Agents: Scalability Analysis

Let us see now how they behave with increasing complexity, represented as a function of the number of victims to rescue and the number of agents to coordinate. From Table 2a and Table 2b, one can tell that with low complexity (1 to 4 victims and 1 to 4 agents), the team of C agents outperforms the heterogeneous A+B agents. On the other hand, the more complex the rescue problem (larger number of victims or agents), the more time a team of homogeneous agents needs, and consequently the worse it performs.

For the sake of simplicity of analysis, we compared the cases at three levels of complexity: low, medium, and high. This shows clearly how the growing complexity (in terms of more victims) causes the solution with C type agents to also increase in time, while the A and B types remain unchanged. In another analysis, increasing the number of agents while increasing the complexity (number of victims) causes A+B agents to rescue digital objects in a much shorter time, even with a small number of A+B agents, a trend much stronger than shown by the C agents.

Table 2a. Behavior of Heterogeneous A+B type agents teams with different complexities of the Rescue problem

High	6-4	Mediun	6-4	Low	6-4
Avrg	2522	Avrg	1712	Avrg	1333
Desv.	1246	Desv.	1246	Desv.	1010
Median	1500	Median	1600	Median	900

High	10-8	Mediun	10-8	Low	10-8
Avrg	460	Avrg	340	Avrg	356,1
Desv.	382	Desv.	147	Desv.	288
Median	340	Median	310	Median	240

High	14-12	Mediun	14-12	Low	14-12
Avrg	343	Avrg	264,4	Avrg	315
Desv.	238	Desv.	157	Desv.	330
Median	220	Median	210	Median	230

Table 2b. Behavior of Homogeneous C type agents teams with different complexities of the Rescue problem

High	6-4	Mediun	6-4	Low	6-4
Avrg	1927	Avrg	2511	Avrg	4760
Desv.	1257	Desv.	1658	Desv.	2691
Median	2500	Median	1630	Median	3900

High	10-8	Mediun	10-8	Low	10-8
Avrg	469	Avrg	1380	Avrg	1802
Desv.	304	Desv.	625	Desv.	785
Median	450	Median	1200	Median	1780

High	14-12	Mediun	14-12	Low	14-12
Avrg	590	Avrg	841,1	Avrg	1322
Desv.	349	Desv.	304	Desv.	561
Median	450	Median	970	Median	1350

The main conclusion is that under *uncertain complexity,* it is advisable to have the higher as possible of numbers of simple agents (of type A) and few of superior types. This is the consequence that the heterogeneous teams normally outperform the homogeneous teams because of their better scalability. As shown in Table 2a, the heterogeneous teams complexity does not increase while increasing complexity (more digital objects to preserve), and as shown in Table 2b, the homogeneous agents complexity grow along the increasing complexity of digital preservation. By *uncertain complexity* we mean we do not know the number of digital objects that are to be preserved or are worth preserving. In this paper there is no policy influencing the selection and adoption of particular strategies for digital preservation.

Another interesting result is that 8-10 A and 4 B type agents outperform 14 B type agents, though this assertion has a flaw: the B agents really need of A agents to work, so one can claim this comparison is not fair. The following section will address this issue.

5.3 Creating Perfect Agents and the Impact on Design

There remains the question: *What if we make a team with the best properties of A and B agents, the D type super-agents?* The answer is that the mixture of A and B agents have the analogous performance as the same number of D agents. Let us examine table 3 for the features of the agents.

Table 3. Complexity of Cases 1 Heterogeneous A+B Agents and Cases 2 and 3 Homogeneous C and D Agents. N X means number of agents of type X, V X means the speed of the agents of type X, and P X means the perception (diagnostic capabilities) of the agents of type X

Team	N A	N B	N D	V A	V B	V D	P A	P B	P D	Cost (kU)
1: A+B	10	4	0	2	1	-	5	1	-	25,00
2: C	0	14		-	-	1	-	-	1	70,00
3: D	0	0	14	-	-	2	-	-	5	700,00

Type	A	B	D
Features	6	8	9
Cost (kU)	0,50	5,00	50,00

One can see the types of agents and teams have diversity of skills as well as different costs, measured in a kU, an estimation of a cost that such type of software could imply in terms of effort, knowledge, time or money. The more skills, the more costly.

Table 4. Descriptive statistics of the experiments

	Case	Mean	Std. Deviation
Time (ticks)	1: A+B	531,37	364,87
	2: C	859,27	622,54
	3: D	432,36	274,27
Rescued	1: A+B	9,79	0,37
	2: C	9,63	0,53
	3: D	9,82	0,32

The experiments are shown in Table 4, 50 runs of every case, through a complexity range of 1-9 victims, identical initial conditions in every case. The result is that the same number of A+B type heterogeneous agents perform similarly to D type homogeneous agents, even though the D type agents combine the best features of types A and B, and consequently are the most expensive. The shorter time to save digital objects implies that more digital objects are rescued.

The conclusion that the heterogeneous agent teams perform reasonably well, as good as the best team of homogeneous agents will have an important impact on the design of teams of agents. For example, taking the team of 10A+4B softbots and increasing it by 3 A agents then it will perform as well as a team of 14 D agents. This is a symptom that slight cost increments of the heterogeneous teams imply big increase in performance. This has impact in the design of digital preservation teams, in the next section.

6. DESIGN OF DIGITAL PRESERVATION TEAMS

After acknowledging that the homogeneous team composed of "perfect" agents does not perform *that* much better than the heterogeneous team of both cheap and expensive agents, some agent design issues then arise. The purpose is to design types of agents specifically designed to cooperate under S&A. Existing formulations for designing physical agents by taking into account the self-awareness and social dimensions were proposed in [19]. The novelty here is that the engineer can design a pyramid of agents, every type of agent at one level, deciding how many levels and what properties are in each level:

a) The ratio among the levels could be, for example, one order of magnitude. In our example, 4 B type agents can work with 20-40 A-type agents. Consequently, 1 Super agent (C) could work with up to 10 B type agents.

b) The cost ratio between the levels could again be one order of magnitude in computational cost.

c) Agents must have web semantic, crawling and appraisal capabilities.

d) Agents must be autonomous, proactive, and reactive, though superior types of agents could work with advanced planning and OISE preservation systems.

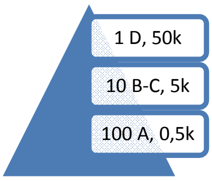


Figure 5 A Pyramid of agent teams. The numerous, simple, and lighter A agents form the base. The cost, capabilities, and sophistication increase through the higher levels, B and C.

If these principles for designing Digital Preservation teams are applied, then the costs and performance of Table 5 would be achievable. The value of each configuration is heuristically estimated as the ratio of cost over performance, so our measure of value is the cost for each precious tick (time) used in rescuing digital objects. We considered that the cost of agents increases exponentially in function of the number of on-board properties. In Fig. 5 cost is modelized with 1 order of magnitude each type. Using this measure of value, the heterogeneous A+B team configuration is the optimal one (0.085 value), and the homogeneous team of super agents D is the worst (1.619 value). However, if saving (preserving) digital objects is extremely critical and there are infinite resources available, then A+B and D configurations would have equal values. Deciding which one to apply becomes a matter of taste or art.

Table 5. Descriptive statistics of the experiments

Unit \ Cost (kU)	0.5	10	50	Cost	Performance	Value
A+B	10	4		45	531.37	0.085
C		14		140	859.27	0.163
D			14	700	432.36	1.619

7. CONCLUSIONS

Our approach exhibits the aspects of the handing over the control of monitoring digital objects to be rescued (preserved) and real time preservation planning to the rescue agents (softbots or robots). There is no centralized service required because the softbots are broadly autonomous. Moreover, a non-linear approach to preservation is also achieved due to the abilities of the softbots to allocate resources in run-time. Our approach also shows a different way of prioritizing which digital objects need to be rescued and consequently an optimisation in using preservation resources. Creating new types of agents specifically to work cooperatively will greatly improve the efficiency of preserving digital objects, by following the analogy of rescuing people in unknown environments. This approach copes with the today and future exponentially increasing complexity of the digital objects preservation. The fact that heterogeneous teams of agents outperform homogeneous ones was stated in [20] and is now shown again using algorithms like Shout and Act that takes advantage of the finding. Still, there are pending problems like what happens when complexity takes on a third dimension, i.e., multiple formats that are at risk across multiple files. Finally some engineering principles should be taken into account in designing Digital Objects Preservation teams, since lower-cost heterogeneous teams of agents can achieve the same efficiency as more costly homogeneous teams. This is an important conclusion to back this approach on the application of agents to digital preservation, since the growing complexity of digital objects is one of the most challenging issues to solve.

8. ACKNOWLEDGMENTS

This research was partially funded by the European Project Num. 216746 PReservation Organizations using Tools in AGent Environments (PROTAGE), FP7-2007- ICT Challenge 4: Digital libraries and content, the PR2007-0432 Salvador de Madariaga Grant, and the Spanish MCYT project DPI2007-66872-C02-01, Arquitecturas Multiagente de Control e Interacción de Agents Móviles en su Aplicación al Rescate de Supervivientes de Catástrofes con Agua.

9. REFERENCES

[1] Dorigo, M. & Stützle, T. 2004. Ant Colony Optimization, MIT Press.

[2] Parsopoulos, K.E. and Vrahatis, M.N., 2002. Recent approaches to global optimization problems through particle swarm optimization, Neural Computing, 1 (2-3), 235–306.

[3] Holland, E. O., Beckers, E., R. and Deneubourg J.L., 1994. From Local Actions to Global Tasks: Stigmergy in Collective Robotics, 181–189, Artificial Life IV, MIT Press, Cambridge, MA.

[4] Bonabeau, E., Dorigo, M. & Théraulaz, G. 1999. Swarm Intelligence. From Natura to Artificial Systems, Oxford University Press, 1st edn.

[5] Bonabeau, E. & Théraulaz, G. 2000. Swarm smarts, Scientific American, March 2000, 72–79.

[6] Engelbrecht P. A. 2006. Fundamentals in Computational Swarm Intelligence, John Wiley and Sons.

[7] Lewis, M. A. & Bekey, G. A. 1992. The Behavioral Self-Organization of Nanoagents Using Local Rules, Proceedings

of the 1992 IEEE/RSJ International Conference on Intelligent Agents and Systems.

[8] Resnick Turtles, M. 1997. Termites & Traffic Jams, Explorations in Massively Parallel Microworlds, MIT Press.

[9] Acebo E., de la Rosa J.L., Bar Systems. 2006. A Class of Optimization Algorithms for Reactive Multi-Agent Systems in Real Time Environments, 17th European Conf. on AI. (ECAI2006) Intl Workshop on New Trends in Real Time AI. pp: 128-133, Riva de Garda, Italy, Aug 28-29.

[10] Quisbert, Hugo. 2008. On Long-term Digital Information Preservation Systems – a Framework and Characteristics for Development. Doctoral Thesis. 2008:77. Luleå University of Technology.

[11] Ross, S & Hedstrom, M. 2005. Preservation research and sustainable digital libraries. International Journal on digital Libraries, Vol. 5, No 4, pp 317-324.

[12] Cheney, J, Lagoze, C. and Botticelli, P. 2001. Towards a theory of information preservation. In Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries.

[13] Runardotter, M., Quisbert, H., Nilsson J., Hägerfors, A., & Miriamdotter, A. 2005. The Information Life Cycle – Issues in Long-term Digital Preservation. Proceedings of IRIS 28 – Information system research seminar in Scandinavia. Kristiansand, Norway, August 6-9.

[14] Moore, R. 2008. Towards a Theory of Digital Preservation. International Journal of Digital Curation. Vol. 3, No. 1, pp 63- 75

[15] Day, M. 2008. Toward Distributed Infrastructures for Digital Preservation: the Roles of Collaboration and Trust. International Journal of Digital Curation. Vol. 3, No. 1, pp 15-28

[16] Hiroaki Kitano, Satoshi Tadokoro, Itsuki Noda et al., 1999. RoboCup Rescue: Search and Rescue in Large-Scale Disasters as a Domain for Autonomous Agents Research, Proc. 1999 IEEE Intl. Conf. on Systems, Man and Cybernetics, Vol. VI, pp.739-743, Tokyo Oct, 1999 (SMC 99).

[17] Wooldridge, M. 2002. An Introduction to Multiagent Systems. Wiley & Sons Ltd.

[18] Satoshi Tadokoro, Tomoichi Takahashi, Hironao Takahashi, et al.2000, The RoboCup-Rescue Project: A Robotic Approach to the Disaster Mitigation Problem. ICRA 2000.

[19] de la Rosa J.Ll., Figueras A., Quintero C., Ramon JA., Ibarra S. and Esteva S.2008. Outline of Modification Systems, Studies in Computational Intelligence ISSN: 1860-949X Vol: 57, pp: 231-247, Springer Verlag.

[20] de la Rosa J.Ll., Muñoz I. 2007. Learning and Adaptation in Physical Heterogeneous Teams of Agents, VIII Workshop in Physical Agents WAF 2007, pp: 9 – 18.

[21] Tesfatsion L., Iowa State University http://www.econ.iastate.edu/tesfatsi/repastsg.htm

[22] Rahwan, T., Ramchurn, S. D., Dang, V. D., Giovannucci, A. and Jennings, N. R. 2007. Anytime optimal coalition structure generation. In Proc. 22nd Conf. on Artificial Intelligence (AAAI), pages 1184–1190.

[23] de la Rosa, J. L., Bengtsson, J., Ruusalepp, R, Hägerfors, A, & Quisbert, H. 2008. Using Agents for Long-Term Digital Reservation the PROTAGE Project, Book Series Advances in Soft Computing Publisher - International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008), Vol. 50/2009 pp:118-p:122, ISSN 1615-3871(Print) 1860-0794 (Online), Springer

[24] Dong, J., Chen, S., Jeng, J.J., 2005. Event-Based Blackboard Architecture for Multi-Agent Systems, itcc, pp. 379-384, International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II.

[25] T. White. Swarm Intelligence, http://www.sce.carleton.ca/netmanage/tony/swarm.html, 2007.

[26] Vitorino Ramos, Carlos Fernandes, Agostinho, C. Rosa Social Cognitive Maps, Swarm Perception and Distributed Search on Dynamic Landscapes, Brains, Minds & Media, Journal of New Media in Neural and Cognitive Science, NRW, Germany, 2005, arXiv:nlin/0502057v1 [nlin.AO]

InDP'09, June 19, 2009, Austin, TX, USA. Copyright 2009.