Agents and Security in a cultural assets transport scenario

Stefania Costantini, Arianna Tocchio, Panagiota Tsintza
Dipartimento di Informatica
Università degli Studi di L'Aquila
Via Vetoio, Loc. Coppito,L'Aquila - Italy
Email: {stefcost,tocchio,panagiota.tsintza}@di.univaq.it

Leonardo Mostarda
Department of Computing
Imperial College London,
London, UK SW7 2AZ
Email: lmostard@doc.ic.ac.uk

***Abstract*— Museums and exhibitions represent a relevant contribution to the economy all over the world. In Italy, in the year 2006 the 400 national museums, monuments and archaeological sites have been visited by 34.492.875 people with an average entrance fee of 6,64 euro for person while in France 18.367.000 people decided to dedicate some time for visiting the national museums. Considered the increasing relevance of the cultural and economical level of museums, several works in Artificial Intelligence (AI) have proposed new methodologies for supporting the users during their visits. However, few research groups have faced the problem of the cultural assets transport. This paper pays attention to a particular aspect of the museum activities: how to identify and to transport in a secure way the cultural assets. In fact, a higher security in transport among museums may increase the exchanges and, consequently, the cultural offer. For reaching this goal we exploited the Galileo Satellite services and the Intelligent Agents technology and we experimented the system in a real scenario.**

I. Introduction

People often believes that transport of a cultural asset is a simple process. Instead, the act of moving a work of art from a place to another one requires attention, experience and competence. In fact, the transport phase hides risks, delays, anxieties and difficulties due to the unpredictability of the events. The quest for perfection in this field is unavoidable, as slight differences may determine a success or a failure. Every day, several companies in the world try to reach the goal of moving cultural assets while reducing as much as possible the risks. This process often involves an accurate packaging phase, an escort service from the origin to the destination and an insurance policy, as criminal actions are always possible.

A relevant role in the prevention of the criminal activities could be assigned to the Galileo satellite, a "big brother" capable not only of precisely identifying the position of a cultural asset but also of following it during the journey. The context in which technology can increase security involves both the packaging and the transport monitoring phase. In the packaging phase, particular devices and sensors are put in the cultural assets packs. In the transport phase, these devices receive the Galileo satellite signal and allow one to track the cultural assets movements.

In previous work, we have combined the Galileo infrastructure with the common security mechanisms to build the Geo Time Authentication system (GTA) [9]. The GTA provides the following services: (i) cultural asset identification and authentication; (ii) integrity of cultural assets information; (iii) secure transport of cultural assets. Secure transport is achieved by means of the GTA monitoring component. This component is wrapped in each cultural asset package and is connected to several sensors (i.e., temperature, humidity and light sensors). At run time, the GTA monitoring component controls the sensor data variation to detect package opening. It also checks the mutual position among packages to detect possible thefts and uses the Galileo signal to check the correct transport routing.

However, in real experiments [10] that we have performed we have noticed that the GTA monitoring component can rise false alarms. These are consequence of unexpected environmental conditions (e.g., quickly weather breaks, sudden track breaking) that require some intelligent deductions missing in the GTA implementation. In this paper, we present recent developments aimed at enhancing the GTA system by means of the deduction capabilities of intelligent agents. There are good reasons for adopting agents in order to improve GTA capabilities. Agents offer autonomy, reactivity, pro-activity, social ability, very useful for all applications where some degree of autonomy is needed. There are application contexts that actually offer no alternative to autonomous software. Agents provide a tool for structuring an application so as to support the design metaphor in a direct way. In this sense, they offer an appropriate support to the development of complex systems.

Agents and multi-agent systems (MAS) have emerged as a powerful technology for facing the complexity of a variety of ICT scenarios. There are now several industrial applications that demonstrate the advantage of using agents. However, agent systems have yet to achieve widespread deployment in operating environments, as technology has to move from pure research to development. In our context, intelligent agents have been used to discriminate between motivated and unmotivated warnings signaled by the GTA in the transport scenario. The improvement due to introducing agents is based on the intelligent and cooperative analysis of particular events like, for example, the change of the external temperature or a grinding halt.

Consider a possible scenario for the first case. The truck is

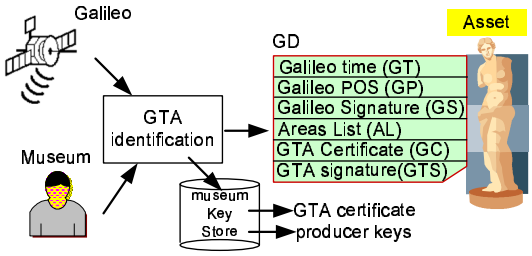

Fig. 1. The identification of a cultural asset

going from Rome to Naples. It contains several packs, each one controlled by the GTA, an Intelligent Agent and some sensors. At a certain moment, a pile-up blocks the traffic along the highway. We are in July, the external temperature is high and the sensor in a pack measures a temperature degree beyond the prefixed threshold. The GTA, after giving the alert, waits for the MAS evaluation. The Agents in the MAS compare their information about the temperature and, through a reasoning process (by evaluating the information about the season, the position of the truck, the kind of freezing plant and so on) try to deduce if the alert is motivated. In fact, if the temperature is high in all packs as well as in an external device, there are good motivations for supposing that no person opened one or more pack but, rather, that either the external temperature influenced the internal one of the packs or the freezing plant in the truck is malfunctioning. Consequently, a message is sent to the route supervisor about the false alarm and the temperature threshold is incremented. An unmotivated alert has been avoided. We are aware that in real applications in a first stage the agents behavior must be monitored for avoiding that actual alerts could be underrated: however, after an initial verification phase, the agent can be trained to efficiently support traditional techniques in security scenarios.

This paper is organized as follows. Section II shortly describes the main characteristics of the GTA component. Section III presents the MAS structure, paying a particular attention to the environment role, while Section IV explores the Agents activities in the monitoring scenario. Section V relates about some aspects of the MAS implementation while Section VI introduces the main motivations for this choice. Finally, Sections VII and VIII conclude the paper.

II. The Geo Time Authentication system at a glance

The Geo Time Authentication (GTA) [9] is a prototype system that provides authenticity and integrity of cultural assets information. It has been conceived in the context of the CUSPIS project [10] and, afterwards, it has been generalized to the context of assets and goods where relevant problems of counterfeiting and thefts exist. To prevent these crimes, the GTA system provides the following services: (i) identification; (ii) authentication; (iii) access control; (iv) integrity; (v) privacy and confidentiality; (vi) non repudiation; (vii) secure transport of assets. As we are going to see in the rest of this section, each service is based on traditional cryptography mechanisms enhanced with the satellite information (i.e., latitude, longitude and time).

The GTA identification service (see Figure 1) permits to uniquely identify a cultural asset. To this aim, each museum is equipped with a *GTA identification component*. This component takes as input the Galileo signal and a key store that contains the museum private key. The component output is a so-called "GD" (GTA Data) for each asset.

A GD contains the following information: (i) the Galileo time (GT); (ii) the Galileo geographical coordinates (GP); (iii) the Galileo Signature (GS); (iv) the Areas List (AL); (v) the GTA certificate (GC); (vi) the GTA signature (GTS). The Galileo geographical coordinates (GP) are the longitude and the latitude of the geographical area where the cultural asset identification takes place.

The Galileo time (GT) refers to the time when the identification is performed. The concatenation of both GP and GT is referred to as Galileo Identifier (GAID) and is used to uniquely identify the cultural assets. In fact, each museum is assigned its own identification area and thus, at any time, exactly one GD can be produced. The Areas List (AL) refers to the list of geographical areas where the cultural asset will be exposed. The GTA certificate (GC) contains the identity of the museum that has generated the GD. Finally, the GTA signature (GTS) is the signature of all fields for ensuring GD integrity.

The GTA authentication service guarantees the authenticity of a GD. This ensures that the cultural assets authenticated in the museum are the ones indicated on the GD and that the GD was indeed generated in the origin indicated on it. In particular, GD authentication prevents an attacker from masquerading as a legitimate museum. In this way, introducing counterfeit objects in the market becomes more difficult. For instance, in the CUSPIS case study that we have worked out, the GTA has been used to guarantee that an ancient sculpture was indeed catalogued by the Greek museum in Athens.

The GTA access control service is the GTA ability to limit and control the access to the GTA Data (GD) information. To this aim, each entity must be authenticated, so that access rights can be tailored to the individual case. The GTA integrity service ensures that a GD is received as sent, i.e, no duplication, reuse, destruction can be performed. The GTA privacy and confidentially services guarantee that GD information is provided only to authorized people. The GTA non repudiation service prevents a producer to deny a generated GD. The GTA secure transport of assets ensures that assets are not stolen or substituted during the transport phase. In particular, this service is performed by monitoring that (i) the transport is routed along the correct path; (ii) the variation of temperature, humidity and light inside the cultural assets package does not exceed a threshold (i.e., packages are not opened); (iii) mutual position among packages do not change over the time (i.e., packages are not stolen).

However, the GTA transport security measures can be too strict and sometimes can rise false alarms that need to be handled by the security system manager. For instance, if the

environmental conditions quickly change (e.g., the weather quickly breaks, the track is inside a tunnel) then the thresholds can be subject to a considerable variation. A track braking can cause all packages to change their mutual positions and an alarm is raised. To address such unexpected situations, in this paper we enhance the GTA monitoring system through the deduction capabilities of intelligent agents. We have experimented the cooperation between the GTA and the intelligent agents in order to build a more flexible transport monitoring system that is able to recognize false alarms in the transport of cultural assets.

III. Environment and MAS structure

After explaining the main characteristics and the role of the GTA, in this section we formalize the infrastructure of the MAS environment by means of the approach of Viroli et al. ([25]). We examine the role of agents and environment in the monitoring scenario. The above-mentioned paper proposes to analyze the roles and the features of a MAS environment by decomposing it into basic bricks called *environment abstractions*. Each agent perceives the existence of such abstractions and interacts with them in order to achieve individual or social goals. Figure 2 provides an overview of the infrastructure of the MAS application.

At the bottom level, the physical support specifies the hardware components of the system whose data the MAS is interested in. In particular, we mention the ABUs (Asset Board Unit) which are satellite signal receivers contained in the packs, the Galileo infrastructure, the sensors used to capture the changes in the environmental transport conditions (variation of the temperature or of the light intensity, ...) and the generic communication infrastructure. Each ABU receives the Galileo signal and transmits the pack's position to the MAS.

The execution platform includes the operating systems, the virtual machine and other middlewares. The interpreter of the DALICA MAS agents has been written in Sicstus Prolog [22] while other functionalities have been implemented in Java. Jasper [22] allowed us to interface the prolog language with the Java one. The execution platform also includes the Agent/Environment communication middlewares: the Linda Tuple Space [22] and the Event-based communication support. Each information coming from external sources (ABUs, sensors,...) is transformed into an event and put in the Tuple Space to be received by the agents. When the agents needs to communicate with the external world, its messages are transformed into events through the Jasper interface and delivered to the corresponding devices.

Components external to the MAS communicate by means of an event-based mechanism. Figure 3 synthesizes the communication infrastructure. Messages coming from the MAS and addressed to the external components are received by the Output Agent and, via the Jasper Interface and the Server RMI, are sent to the Messaging Component. The Messaging Component implements a plug-in architecture for supporting run-time registration of both server and client components. After the registration process, each client interacts with the messaging component and receives an ID that will be used to send and receive events. In other words, after the registration process, each client can build and send to the messaging component an event containing its ID and the server component ID. It is worth noticing that a basic event does not provide meaningful information since it only contains routing information (i.e., the receiver and the sender IDs). Therefore, a basic event can be (if needed) specialized into a new one that contains an additional field. This field can be used for instance by a hardware component in order to send detected data to a server component. The MAS component uses it for receiving environmental data and for replaying to external components such as as VCC, GTA and so on.

At the top layer, we find the MAS application. The Application Agents area contains the three kinds of agents composing the MAS:

Control Device Agent: The role of this agent is to monitor the condition of the assigned pack, by checking both the position through the Galileo satellite signal and the sensors conditions.

Transport Device Agent: This agent has the role of coordinating several Control Device agents. It is capable of integrating the warnings coming form the Control Device Agents. It attempts to deduce what is happening and to evaluate the alert degree. The role of this agent will be explained at more length in the next sections.

Output Agent: Manages communications between the MAS and the external infrastructures such as VCC (Virtual Control Center), Sensor interface, GTA and so on.

Finally, the Application Environment contains the Sensors Interface that allows Control Device agents to get information about the temperature, humidity or light in the packs; the GTA component and the VCC components. The former one receives the results of the MAS deduction process while the latter one manages the communication between the MAS and the security control center where human operators monitor the transport conditions.

IV. Monitoring the Cultural Assets transport

In the transport scenario, Control Device Agents are used for checking several transport parameters and have the responsibility of informing the authorities in case of tampering or theft. In the following, we describe all phases involved in the transport of cultural assets (see Figure 4) by emphasizing the agents roles.

In the transport planning phase, the owner of the cultural assets, the renter (i.e., the entity who wishes to take the cultural assets) and third-part entities (i.e., those who vouches for the content and the routing of transport) cooperate to produce different certificates. In this paper, we focus on the authorization and the transport certificates since they are used by the MAS.

Each package of the transport is equipped with an authorization certificate that contains the list of all cultural assets inserted into the package. This certificate is used to check the presence of the cultural assets inside the package.

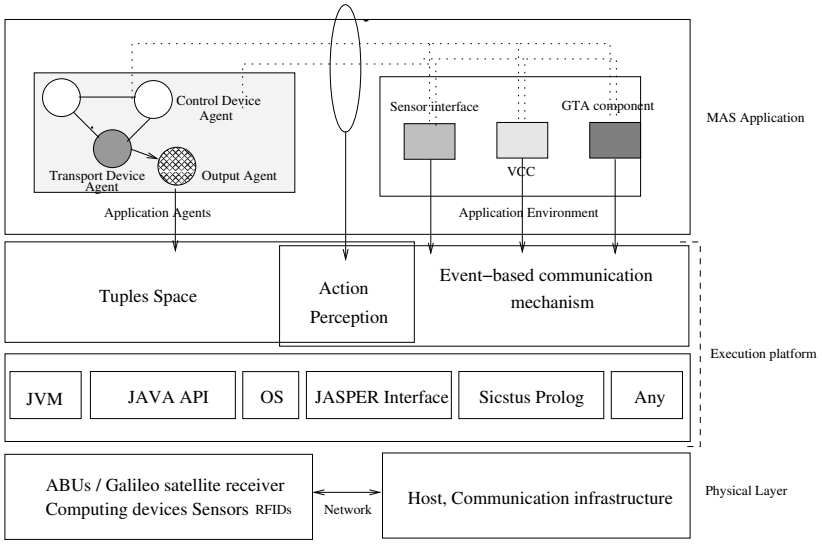

Fig. 2. MAS Environment Application Layers

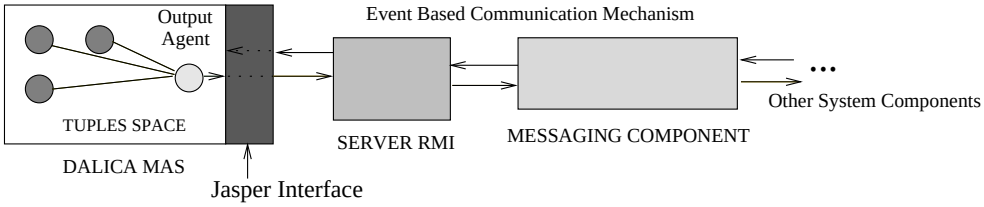

Fig. 3. Communication Infrastructure

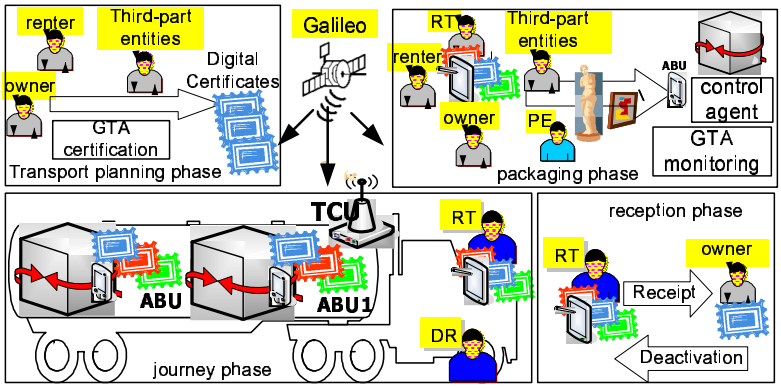

Fig. 4. The CUSPIS system

Each transport has exactly one transport certificate that contains the correct routing. The routing is defined in terms of a list of couples $\{(A_s, T_{A_s}), (A_1, T_{A_1}) \ldots (A_i, T_{A_i}) \ldots (A_n, T_{A_n}) (A_d, T_{A_d}) \}$ where A_s is the starting transport area and T_{A_s} the related date (i.e., day and hour), A_i an area that the transport has to pass and T_{A_i} the related date; (A_d, T_{A_d}) is the destination area and its date. It is worth noticing that for specific assets transport other certificates (e.g., insurance certificates) can be added and are signed by a certain number of entities.

In the packaging phase, the above entities in cooperation with the responsible of transport (RT) and the packaging expert (PE) supervise the packaging of assets. Each package is filled with: (i) a set of assets, each of them identified by an RFID tag, which is an object that can be attached to or incorporated into a product, animal, or person for the purpose of identification using radiowaves; (ii) a Control Device (ABU); (iii) a sensor of humidity; (iv) a sensor of light; (v) a sensor of temperature. The ABU unit contains the authorization and the transport certificate. Moreover, it hosts both the GTA monitoring component and the control agent. While the GTA monitoring component provides security by means

of traditional mechanisms (i.e., cryptography, certificates and detection algorithm) the control agent enhances security issues through intelligent deductions.

During the journey, each Control Device Agent, which is capable of acting in a proactive way, performs the following activities: (i) correct routing checking; (ii) cultural assets verification; (iii) sensor data checking; (iv) package position verification.

The Correct routing checking is performed by the Control Device Agent by using both the Galileo signal and the transport certificate. In particular, the agent uses the Galileo satellite to check that each area is passed at the right time. The cultural assets verification is an activity in which the control agent loads all cultural assets IDs contained in the authorization certificate and checks their presence inside the package. It is worth noticing that this activity is performed repeatedly over time. Both correct routing checking and cultural assets verification are simultaneously performed also by the GTA monitoring component. Therefore, the agents contribution is a redundant check that enhances the system fault tolerance.

The sensor data checking verifies the variation of the sensor data over time. This variation must not exceed a given threshold that, as we are going to see in the following, is dynamically adapted by means of the agents cooperation. This check ensures that a package is neither opened nor kept in a dangerous environment. In this case, the agent contribution is indeed needed since temperature checking must involve some intelligent reasoning. For instance, temperature may change for environmental reasons so that the GTA may raise a false alarm. If the temperature of the environment changes for all packs because of a natural process and overcomes the threshold, the agents in the ABUs can activate a communication process and reach the conclusion that no package has been tampered because each of them signals the same temperature.

The package position verification ensures that all packages are in the correct position. In this process the agents exploit their cooperation capabilities. In fact, from time to time each Control Device Agent sends a message to the other ones asking their position. Then, it computes the distance and verifies that the positions do not vary over time. In fact, a variation of the package mutual position can imply a package theft. In this case, the agent reasoning is indeed needed to enhance the whole system effectiveness. In fact, due e. g. to a quick break, the packages mutual position can change. In this case, the agent contribution can detect and avoid GTA false alarms.

When a Control Device Agent detects some anomaly, it sends a warning message to the Transport Device Agent whose role is to cross the information with those coming by the other entities and to verify the seriousness of the warning. Elaborated the degree alert, the Transport Device Agent sends a message to the VCC terminal. Moreover, each Control Device Agent maintains a direct communication channel with the GTA component. In fact, if the GTA identifies a suspicious situation, it contacts the agent in order to trigger a checking process of the package status.

V. The approach in more detail

Control Device, Transport Device and Output Agents have been implemented in the DALI language. DALI has a Sicstus Prolog interpreter, while the other components, like the GTA, have been implemented in Java. The interface between the two languages has been provided by the Jasper Sicstus Prolog library. In the rest of this section, we illustrate the main features of the DALI language and explain how DALI reactivity and pro-activity has been used for implementing the agents behavior.

A. The DALI language in a nutshell

DALI [5] [6] [24] [7] [8] is an Active Logic Programming language designed in the line of [16] for executable specification of logical agents. DALI is a prolog-like logic programming language with a prolog-like declarative and procedural semantics [17]. In order to introduce reactive and proactive capabilities, the basic logic language has been syntactically, semantically and procedurally enhanced by introducing several kinds of *events*, managed by suitable *reactive rules*. All the events and actions are time-stamped, so as to record when they occurred. These features are summarized very shortly below.

An *external event* is a particular stimulus perceived by the agent from the environment. We define the set of external events perceived by the agent from time t_1 to time t_n as a set $E = \{e_1 : t_1, ..., e_n : t_n\}$ where $E \subseteq S$ and S is the set of environment states. The e_i's are atoms indicated with postfix E in order to be distinguished from both plain atoms and other DALI events. External events allow an agent to react through a particular kind of rules, reactive rules, aimed at interacting with the external environment. When an event comes into the agent from its external world, the agent can perceive it and decide to react. The reaction is defined by a reactive rule which has in its head that external event. The special token $:>$, used instead of $:-$, indicates that reactive rules performs forward reasoning.

A reactive rule has the form:
$ExtEvent_E :> Body$ or
$ExtEvent_{1E}, ..., ExtEvent_{nE} :> Body$
where $Body$ has the usual (logic programming) syntax and intended meaning except that it may contain the DALI event and action atoms.

The *internal event* concept allows a DALI agent to be proactive independently of the environment by reacting to its own conclusion (notice that this feature can be considered as a form of introspection). More precisely: An internal event is syntactically indicated by postfix I and implies the definition of two rules. The first one contains the conditions (knowledge, past events, procedures, etc.) that must be true so that the reaction (in the second rule) may happen:
$IntEvent : -Conditions$
$IntEvent_I :> Body$

Internal events are automatically attempted with a default frequency customizable by means of user directives in the initialization file that can tune also other parameters such as how many times an agent must react to the internal event

(forever, once, twice,...) and when (forever, when triggering conditions occur, ...); how long the event must be attempted (until some time, until some terminating conditions, forever).

Actions are the agent's way of affecting the environment, possibly in reaction to either an external or internal event. An action in DALI can also be a message sent by an agent to another one. An action atom is syntactically indicated by postfix A. Clearly, when an atom corresponding to an action occurs in the inference process, the action is supposed to be actually performed by suitable "actuators" that connect the agent with its environment. In DALI, actions may or may not have preconditions: in the former case, actions are defined by actions rules, in the latter case they are just action atoms. An action rule is a plain rule, but in order to emphasize that it is related to an action, we have introduced the new token $:<$. External and internal events that have happened (i.e., that have been reacted to) and actions that have been performed are recorded as past events, that represent the agent's memory, and the basis of its "experience".

B. Reactivity and proactivity in DALICA MAS

In this section, we present a snapshot of the Control Device Agent, paying a particular attention to some reactive and proactive capabilities of the agent implemented in DALI. The signal of the Galileo satellite is received by the agent by means of a DALI reactive rule:

$posE(Lat, Lng, Time, Date, Integrity, \_) :>$
$def\_position(Lat, Lng, Time, Date, Integrity).$
$def\_position(\_, \_, \_, \_, Integrity) : -$
$Integrity = 0, no\_correct\_signalA.$
$def\_position(Lat, Lng, Time, Date, Integrity) : -$
$Integrity = 1, positionA(Lat, Lng, Time, Date, 1).$
$def\_position(Lat, Lng, Time, Date, Integrity) : -$
$Integrity = 2, positionA(Lat, Lng, Time, Date, 2).$

where Lat and Lng are, respectively, the latitude and the longitude of the pack position, while Time and Date have the obvious meaning. This reactive rule "filters" the Galileo signal according to its integrity value. Only if the integrity is different from 0, the signal is accepted and, by means of the action $positionA(Lat, Lng, Time, Date, \_)$, enables related pro-active rules for the needed inferential activities. In fact, the action $positionA(Lat, Lng, Time, Date, \_)$ is transformed into the past event $positionP(Lat, Lng, Time, Date, \_)$. This transformation, managed by the DALI interpreter, records the past event into the agent's memory. Then, the past event can trigger proactive inference as a condition of an internal event.

Proactive and reactive capabilities are adopted by the Control Device Agent for monitoring the cultural assets transport. We propose a snapshot of the DALI code that controls and manages the temperature threshold.

The internal event $monitor\_temperature$ checks wether the temperature of the cultural assets in the pack is greater than the prefixed threshold $threshold\_temp(Y)$. If so, then the Control Device Agent sends a message to the other Control Devices Agents (cda's) in order to know if the increase of the temperature is general or has happened only in its pack.

$monitor\_temperature(X) : -sensor\_data(X),$
$clause(threshold\_temp(Y), \_), X > Y.$
$monitor\_temperatureI(X) :> clause(agent(A), \_),$
$messageA(cda, send\_message(give\_temperature(X, A), A)).$

The content of the message requires that only those agents that detect in their packs a temperature greater than the threshold will reply. In particular, they will send the external event $value\_temp\_highE(Ag)$ where Ag is the name of the sender agent. After receiving this external event, the agent sends a message to the Transport Device Agent (tda) in order to check its sensors values. In fact, if the increase of the temperature is justified by a natural motivation as, for example, the sun or the stop in a gallery, an alert is not in order.

$value\_temp\_highE(Ag) :> clause(agent(A), \_),$
$messageA(tda, send\_message(give\_temp(A), A)).$

If the temperature X communicated by the Transport Device Agent via the external event $value\_temp\_tda(X)$ is under the threshold but some parameters are evaluated negatively, then the Control Device Agent sends an alert message to all authorities who are in charge of the transport process. The $reasoning\_module(X, Y, R)$ is the responsible of the environmental parameters evaluation process. It takes as input the temperatures and other available parameters for evaluating if the temperature change is motivated. If the response R is negative, then the change is not motivated and an alert is sent. In this case, the increase of the temperature in the pack does not correspond to that of the external environment. If instead the external temperature overcomes the threshold (like for the packs), then there is no clear evidence of risk. However, the agent informs the responsible of the transport (tr) about the higher temperature and asks for the explicit authorization to increase the related threshold.

$value\_temp\_tdaE(X) :> once(evaluate(X)).$
$evaluate(X) : -clause(threshold\_temp(Y), \_),$
$X < Y, reasoning\_module(X, Y, no),$
$clause(agent(A), \_), messageA(authorities,$
$send\_message(alert\_temperature(A), A)).$
$evaluate(X) : -clause(threshold\_temp(Y), \_),$
$X >= Y, reasoning\_module(X, Y, yes),$
$clause(agent(A), \_), messageA(tr,$
$send\_message(alert\_temperature(A), A)), messageA(tr,$
$send\_message(new\_temperature\_threshold(X, A), A)).$

The monitoring of the packs distances is based on the ABUs positions. When packs are charged into the truck, the initial distance among packs is recorded. For preventing a thief to shift a pack unobserved, agents in the ABUs start, at the beginning of the journey, a cooperative activity with the objective of checking the relative distances among their packs. If the monitoring activity of an agent detects that a distance has been modified, it starts an interaction phase involving all agents in the ABUs. The distances among the packs are verified, the difference between the right distance and the new one is computed for each couple of packs and a specific

algorithm evaluates wether the movement has been collective. If the global movement is coherent, then the Transport Device Agent sends an information message to the VCC and resets the packs distances; else, it sends an alert message. For lack of space, we just propose an example of the proactive Control Device Agent behavior. The following internal event allows the agent within a pack to check the distance among it and the other Agents.

$monitor\_dist(Lat1, Lng1, Date, Ag_1) : -messageP(Ag,$
$send\_message(my\_position(Lat1, Lng1, \_, Date, \_), Ag_1)).$
$monitor\_distI(Lat1, Lng1, Date, Ag_1) :>$
$positionP(Lat, Lng, \_, Date, \_), clause(agent(Ag), \_),$
$clause(default\_distance(Ag, Ag_1, D), \_),$
$verify\_distance(Lat, Lng, Lat1, Lng1, D).$

where $verify\_distance(Lat, Lng, Lat1, Lng1, D)$ computes the distance between the Galileo coordinates (Lat, Lng) and $(Lat1, Lng1)$ and compares it with the default distance D. Every time the agent receives a position from another one, the internal event is triggered and the check is performed.

VI. Motivation: why DALI logical agents

Nowadays, many kinds of applications need some degree of autonomy. There are application contexts that actually offer no alternative to autonomous software. Agents provide a tool for structuring an application in a way that supports its design metaphor in a direct way. In this sense, they offer an appropriate support to the development of complex systems.

Platforms for building autonomous software require dedicated basic concepts and languages. At the level of individual agents, representational elements such as observations, actions, beliefs and goals are required. Reactivity is the ability of an agent to perceive its external environment and take appropriate measures in response to perception. Proactivity is the ability of an agent to take initiatives based on its own evaluation of relevant conditions. Going further on the line of autonomous software, new applications need "intelligence", in the sense of the ability to exhibit, compose and adapt behaviors, and being able to learn the appropriate way of performing a task rather than being instructed in advance.

A multi-agent system (MAS) is a collection of software agents that work in conjunction to each other. They may cooperate or they may compete, or they may adopt a behavior that combines cooperation and competition. In order to form a MAS, agents must have communication abilities together with an internal mechanism for deciding when social interactions are appropriate, both in terms of generating requests and of judging incoming requests.

Among the potential applications, distributed monitoring/control systems (DCMS's for short) appear to be a natural application for agents, by virtue of controllers being in principle autonomous entities. This kind of application implies measuring a system, so as to verify whether specific measurable values are within a pre-defined range, and acting on the system so as to keep specific observable values within the range characterizing an acceptable behavior of the system itself. Measuring a system implies selecting the correct checks to perform at each stage. Controlling the system implies being able to either modify or restore its operational parameters as behavior requires. Agents can replace a human operator in this kind of task. If the controlled system is composed of several parts, single agents can control the various parts, and can cooperate so as to enforce the system overall behavior.

The DALICA system can be seen as a distributed monitoring system. The "objects" of the agents monitoring activity are the packs containing cultural assets. DALICA in fact has been designed as a cultural assets DCMS based on the Galileo platform. An agent-based solution has been chosen for the following reasons. Each pack has to be supervised individually in autonomous way, reacting to every stimulus but also to every relevant exogenous state change: these changes must be detected and appropriate measures must be taken.

Typical events that may happen in a DCMS such as the DALICA system are highly asynchronous. The reactive nature of agents allow asynchronous events to be coped with in a natural way. Moreover, proactivity allows system parameters to be observed and tuned whenever necessary, thus potentially preventing the occurrence of critical situations. The distributed nature of this DCMS implies the need for each component to possibly communicate the events occurred to other components. Then, agents supervising single components form a MAS with the overall objective of coordinating activities also in case of critical situations. Agents communicate to the others that some potentially unwanted change has occurred, and the MAS cooperatively establish which are the necessary actions to be undertaken. However, no agent is allowed to directly force other agents to behave in a certain way. This improves the system reliability also in presence of software malfunctioning.

An Object-Oriented solution has not been applied as in the DALICA scenario it appeared less suitable and more difficult to implement. In fact, in the Object-Oriented paradigm message exchange means methods invocation, i.e., synchronous procedure call, which means that autonomy and privacy of components are hard to reach, especially whenever "public" methods are allowed.

In the particular DCMS that we have considered, some kind of "intelligence" is needed so as to interact with the environment in a flexible customizable evolving way, rather than through predefined rigid unalterable patterns. In our DALI implementation, computational logic has taken a relevant role in this sense, as a good tool for building intelligent agents. The DALI logic language in fact, due to the traditional "fast prototyping" character of logic languages in general, to the new efficient implementation and to the new concepts that it embodies, has proven to be suitable for implementing such an advanced application.

Moreover, in a complex distributed environment, rule-based logical languages allow behaviors to be defined by means of independent sets of rules. These behaviors will be triggered in any order by what happens in the environment, without a complex control structure that should foresee all cases or

combination of cases.

Logic languages in general are evolving from static to "active", and are being enriched with new capabilities based on the "agency" metaphor. In fact, the application presented in this paper practically demonstrates that logic agent-oriented language may provide an affordable way of introducing the engineering of intelligent behaviors into software engineering and development practice. In addition, the clear semantics of such languages allows formal properties of an implemented system to be proved, which is relevant in critical application domains.

VII. Related work

To the best of our knowledge, no other systems exist capable of monitoring the transport of cultural assets by exploiting the Galileo satellite signal and agents technology. However, agents have been adopted for monitoring activities in several contexts.

The work of Bunch et al. in [4] is centered on a particular experimentation scenario: the monitoring of chemical processes by means of software agents. In particular, the KARMEN multi-agent system monitors specific combinations of process conditions of interest to individual plan operators, supervisors and other personnel and notifies them through modes that the individuals select in accordance with corporate policy and personal preferences. The monitoring of Medical Protocols by means of agents is described in the work of Alsinet et al. in [1]. Specialized domain agents assist and supervise the execution of medical protocols in hospital environments. As in the DALICA system, in this sensible context authors have introduced privacy, integrity and authentication methods for guaranteeing a secure process of information exchange among agents.

Finally, we mention the works by Muller [18] and Ramasubramanian et al. [21]. In the former, Intelligent Agents improve network operations by identifying, for example, the overall load of network traffic, what times of the day certain applications load the network, which servers may be over-utilized and so on. Instead, the latter integrates intelligent agents in an Intrusion Detection System. The critical context where agents are put at work (a Corporate Bank, Chennai, India) proves their reliability in facing complex and critical tasks. In order to reduce the points of failures which are unavoidable in centralized security systems, the authors designed a dynamic distributed system in which the security management task is distributed across the network by using intelligent agents.

If we concentrate our attention on the available systems capable of exploiting the satellite signal for monitoring activities, an interesting tool is Track-King [12]. It supervises the temperature of the goods during the transport phase. In particular, the system provides continuous verification that the refrigerator equipment is operating at the right set point. The satellite system has a wide coverage and allows remote intervention if necessary. Track-King does not use intelligent agents for reasoning activities. Other systems exploit the GPS signal for tracking purposes, among which are those proposed by several companies ([13], [14], [15]) for protecting the cars from thefts.

In general, they advice the car owner when the vehicle is moving or when it goes out the prefixed route. The reaction often involves the car halt by means of particular mechanic devices or the sending of an alert text message to the cell phone of the owner. These systems imply reactive capabilities only and no proactivity or learning methods are applied. Moreover, the car monitoring process does not require the adoption of cooperation strategies because each monitoring system in a car works independently of others.

Finally, several works in literature have proposed systems for supporting the users during their visits to museums ([19],[20],[2],[11]). We cite in particular the KORE system [3] where agents have been adopted for reasoning about the visitors profiles. The architecture of KORE is based on a distributed system composed of some servers, installed in the various areas of museums, which host specialized agents. The KORE system practically demonstrates that intelligent agents can have a relevant role in capturing the user profile by observing the visitor behavior. They possess the capability to be autonomous and to remain active while the visitor completes her/his visit; they can percept through the sensors all choices performed by the user and, consequently, activate a reasoning process.

In summary however, to the best of uor knowledge the GTA and the DALICA system jointly applied to a cultural assets transport scenario constitute a novelty in the field of AI.

VIII. Concluding Remarks

In this paper, we have presented the DALICA MAS architecture applied in a transport monitoring scenario. Considering the risks involved in the cultural assets transport, we think that the GTA and the DALICA MAS can be profitably exploited in this context, thus making the whole process more secure. As discussed in the previous section, the role of the agents in monitoring activities is increasingly effective in time. Their reasoning capabilities are relevant in all those cases where it is necessary to discriminate the events and to correctly interpret the reality. In the future, we will enhance DALICA system by introducing new features for reasoning on a wider range of environmental factors. We also intend to improve the learning capabilities of agents both in general and in this scenario.

References

[1] Alsinet, T., Bjar, R., Fernanadez, C., and Many, F. *A Multi-agent system architecture for monitoring medical protocols*, In Proc. of the Fourth international Conference on Autonomous Agents (Barcelona, Spain, June 03 - 07, 2000). AGENTS '00. ACM Press, New York, NY, 499-505. URL=http://doi.acm.org/10.1145/336595.337580

[2] Amigoni, F., Della Torre, S., and Schiaffonati, V., *Yet Another Version of Minerva: The Isola Comacina Virtual Museum*, In Proc. of the First European Workshop on Intelligent Technologies for Cultural Heritage Exploitation, at The 17th European Conference on Artificial Intelligence, 1-5, 2006.

[3] Bombara, M., Cal, D., and Santoro, C., *Kore: A multi-agent system to assist museum visitors*, Proc. of the Workshop on Objects and Agents (WOA2003), URL=http://citeseer.ist.psu.edu/708002.html, 2003

[4] Bunch, L., Breedy, M., Bradshaw, J. M., Carvalho, M., Suri, N., Uszok, A., Hansen, J., Pechoucek, M., and Marik, V. *Software agents for process monitoring and notification*, In Proc. of the 2004 ACM Symposium on Applied Computing (Nicosia, Cyprus, March 14 - 17, 2004). SAC '04. ACM Press, New York, NY, 94-100. URL= http://doi.acm.org/10.1145/967900.967921.

[5] Costantini, S., and Tocchio, A. *A logic programming language for multi-agent systems*, In Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf., JELIA 2002, LNAI 2424, Springer-Verlag, Berlin, 2002.

[6] Costantini, S., and Tocchio, A. *The DALI logic programming agent-oriented language*, In Logics in Artificial Intelligence, Proc. of the 9th European Conference, Jelia 2004,LNAI 3229, Springer-Verlag, Berlin, 2004.

[7] Costantini, S., Tocchio, A., and Verticchio, A., *A Game-Theoretic Operational Semantics for the DALI Communication Architecture*, In Proc. of WOA04, Turin, Italy, December 2004, ISBN 88-371-1533-4,2004.

[8] Costantini, S., and Tocchio, A. *About declarative semantics of logic-based agent languages*, In Post-Proc. of DALT 2005, LNAI 3229, Springer-Verlag, Berlin, 2006.

[9] Costantini, S., Inverardi, P., Mostarda, L., and Tocchio, A. *A Geo Time Authentication System* In IFIPTM 2007, Joint iTrust and PST Conferences on Privacy, Trust Management and Security, in print.

[10] *CUSPIS PROJECT WEB SITE* URL=http://www.cuspis-project.info/demonstrations.htm

[11] Damiano, R., Galia, C., and Lombardo, V., *Virtual tours across different media in DramaTour project*, In Proc. of the First European Workshop on Intelligent Technologies for Cultural Heritage Exploitation, at The 17th European Conference on Artificial Intelligence, 2006

[12] Fick, T. *Remote monitoring: a logistical solution for profiting in a changing world*, In Food Safety, GDS Publishing URL=http://www.gdspublishing.com/ic_pdf/usfs/thki.pdf.

[13] GPS car monitoring system. *SAT track GPS Vehicle Tracking System*, URL=http://www.pcmobile.ca/PC-CDN-Inside.pdf.

[14] GPS car monitoring system. *LoJack - GPS Vehicle Tracking*, URL=http://www.lojack.com/gps-vehicle-tracking.html.

[15] GPS car monitoring system. *Millennium - Advanced Internet Based GPS Vehicle Tracking*, URL=http://www.vehicle-tracking-gps.com.

[16] Kowalski, A. *How to be Artificially Intelligent - the Logical Way*, Draft, revised February 2004, Available on line, URL=http://www-lp.doc.ic.ac.uk/UserPages/staff/rak/rak.html, 2006.

[17] Lloyd, J. W. *Foundations of Logic Programming (Second, Extended Edition)*, Springer-Verlag, Berlin, 1987.

[18] Muller, N. J. *Improving Network Operations With Intelligent Agents*, Int. J. Netw. Manag. 7(3) (Jul.1997), 116-126. $URL = http : //dx.doi.org/10.1002/(SICI)1099 - 1190(199705/06)7 : 3 < 116 :: AID - NEM239 > 3.0.CO; 2 - Q$

[19] Park, D., Nam, T., Shi, C., Golub, G.H., and Van Loan, C.F., *Designing an immersive tour experience system for cultural tour sites*, In CHI '06 Extended Abstracts on Human Factors in Computing SystemsACM Press, 2006.

[20] Pilato, G., Augello, A., Santangelo, A., Gentile, A. and Gaglio, S., *An Intelligent Multimodal Site-guide for the Parco Archeologico della Valle dei Templi in Agrigento*, In Proc. of First European Workshop on Intelligent Technologies for Cultural Heritage Exploitation, at The 17th European Conference on Artificial Intelligence, 2006.

[21] Ramasubramanian, P., and Kannan, A. *Intelligent Multi-agent Based Database Hybrid Intrusion Prevention System*, In ADBIS, 2004, URL=http://springerlink.metapress.com/openurl.asp?genre=article&issn=0302-9743&volume=3255&spage=393, DBLP:conf/adbis/2004, DBLP, URL=http://dblp.uni-trier.de.

[22] *SICSTUS PROLOG HOME PAGE* URL=http://www.sics.se/isl/sicstuswww/site/index.html.

[23] Stallings, W. *Cryptography and network security: Principles and Practice*, Prentice Hall, 2006.

[24] Tocchio, A., *Multi-Agent sistems in computational logic*, Ph.D. Thesis, Dipartimento di Informatica, Universitá degli Studi di L'Aquila, 2005.

[25] Viroli, M., Holvoet, T., Ricci, A., and Schelfthout, K., and Zambonelli, F., *Environments in multiagent systems*, The Knowledge Engineering Review, 20(2), 2005.