Bringing Autonomic Services To Life

LUCIANO BARESI
Politecnico di Milano

ROBERTO CASCELLA
Università Degli Studi di Trento

PETER DEUSSEN
Fraunhofer Institute for Open Communication Systems

ANTONIO DI FERDINANDO AND RICARDO LENT
Imperial College London

SANDRA HASELOFF AND NERMIN BRGULJA
Universität Kassel

ANTONIO MANZALINI AND CORRADO MOISO
Telecom Italia

FABRICE SAFFRE
British Telecommunications
and
FRANCO ZAMBONELLI
Università di Modena e Reggio Emilia

The design and development of the future Internet require identifying the main architectural principles in order to exploit the real innovation necessary to meet the next-generation trends arising from the evolution of the Internet as we know it today. An interesting approach sees tomorrow's Internet becoming a networking service eco-system capable of seamlessly integrating heterogeneous devices and offering services for human-to-human, human-to-machine and machine-to-machine interactions.

This document builds in this direction by describing a distributed component-ware framework for autonomic and situation-aware communication capable of providing dynamically adaptable services. The core of this framework is the *Autonomic Communication Element* (ACE): an innovative agent-based software abstraction capable of enhancing *self-** properties (i.e. self-configuration, self-healing, self-organization and self-protection) to the extent of providing autonomic and optimized services.

An example application is also considered, whereby a future *pervasive behavioural advertisement* scenario is presented along with the architecture of a prototype ACE-based platform that suits the requirements of such scenario.

1. INTRODUCTION

Today's Internet is rapidly evolving towards a collection of highly distributed, pervasive, communication-intensive services. In the next future, such services will be expected to (i) autonomously detect and organize the knowledge necessary to understand the general context in which they operate, and (ii) self-adapt and self-configure, to get the best from any situation, to the extent of meeting the needs of diverse users, in diverse situations, without explicit human intervention.

These features will enable a wide range of new activities that are simply not possible or impractical now. However, the achievement of such capabilities requires a deep re-thinking of the current way of developing and deploying distributed systems and applications. In this direction, a promising approach consists in conceiving services as part of an "ecology" through which they can prosper and thrive at the service of users. This vision is attractive because not only provides better services to end-users, but also allows meeting the emerging economic urge for service provision and system management deriving by the higher level of dynamism and variability of communication systems.

This paper builds in the direction of such approach by describing a framework to support the development of a new generation of services governed by *autonomic applications*, i.e., applications capable of coping well with uncertain environments by dynamically adapting their plans as the environment changes in uncertain ways. The framework represents the major outcome of the CASCADAS EU-IST project[1], and is conceived around a vision based on a set of complementary founding features considered general key enablers around whom any future communication services infrastructure should be conceived. The identification of these features starts from key state-of-the-art concepts in the area of modern distributed computing and communication systems, and tries to advance and generalize them to properly account the specific characteristics of the vision of autonomic and situation-aware communication services. Thus, context-awareness becomes *situation-awareness*; self-organization and self-adaptation converge into a concept of *semantic self-organization*; scalability assumes the form of *self-similarity*; modularity takes the form of a new *autonomic component-ware paradigm* that intrinsically features *self-CHOP*[1] capabilities.

[1] Self-CHOP stands for:

These features are federated in a sound component model, which provides a robust and dynamic modular conceptual framework for building autonomic, self-organizing, semantic services. The component model acts as high-level reference model for the production of a new generation of programmable communication elements that can be reused at different stack levels. These distributed self-similar components are developed around the fundamental software engineering abstraction of *Autonomic Communication Elements* (ACEs).

ACEs are characterised by autonomic features such as self-awareness, semantic self-organisation, self-healing, self-preservation and allow creating and executing dynamically adaptable situation-aware services. The ACE component model is conceived around the notion of *organ*, capable of self-adapting to the conditions of the context the ACE operates in, in the same way as organs in the human body, from which they borrow the name, can do. ACEs' behaviour is specified in one or more *plans* contained in the ACE's *self-model*, initially created by the developer but capable of being modified autonomously by the ACE itself based on self-awareness information.

The CASCADAS Autonomic Service Framework, or the *Framework* for short, is contained in an open source toolkit for situated autonomic communications, also briefly called the *CASCADAS Toolkit* or the *Toolkit*. Through the Java programming language used for development, the Toolkit provides a run-time environment capable of supporting ACEs in all their features, i.e. life-cycle management, interaction with other ACEs and integration of legacy code. Besides, the Toolkit also incorporates libraries for advanced pervasive supervision, ACE aggregation, management of social knowledge, and security mechanisms.

The key idea of the CASCADAS Autonomic Service Framework is that advanced autonomic services can be developed and deployed in the form of ACEs and networks of them. ACEs are capable of autonomously aggregating with each other to provide composite services; they exhibit self-similarity in composition through a set of appropriate interfaces, and they own self-organization capabilities, possibly of a "semantic" (i.e. meaningful) type. The abstraction of ACE is thus the basis for the flexible component-based design and development of any complex service, for the

Self-Configuration (ability to install, to configure and integrate with other complex systems);
Self-Optimization (capability to look for ways to improve its functions, performances continuously.);
Self-Healing (property that allows detecting, diagnosing and repairing problems);
Self-Protection (feature for defending the overall system of problems produced from attacks or failures).

development of the associated supporting tools and, in the end, for delivering a well-architected framework for autonomic and situation-aware communication services.

The rest of this paper is structured as follows. Section 2 provides a description of the CASCADAS Framework and the ACE component model. The key features of the framework are there introduced, and detailed description for each of them is the subject of the subsequent sections. In particular, Section 3 describes support for ACE's semantic self-organization through autonomous organized ACE aggregation; Section 4 studies the mechanisms ACEs employ to become situation-aware, through the use of so-called *Knowledge Networks*; Section 5 studies ACEs' self-healing properties through *Pervasive Supervision*, while Section 6 describes ACEs' self-preservation features and the approach to security aspects. Finally, Section 7 presents and describes a potential industrial future application scenario named *Behavioural Pervasive Advertisement.* Section 8 concludes the paper.

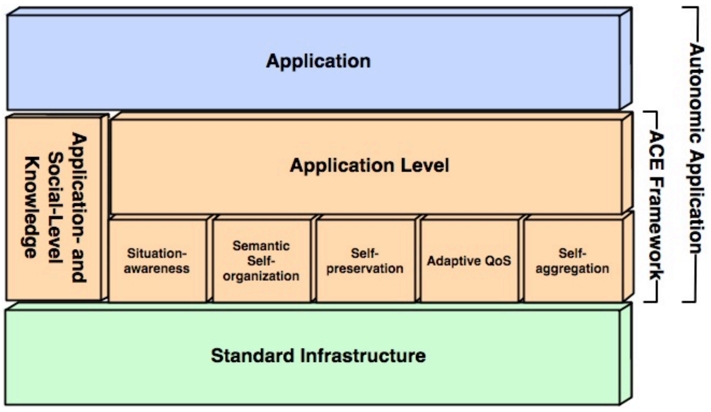

Figure 1: Architecture of the CASCADAS Framework.

2. THE CASCADAS FRAMEWORK

The autonomic service framework foresees applications and services to interact with users according to their social situation, while also evaluating the current networking, physical and contextual conditions. To this end, applications and services are developed and deployed as individual or aggregated ACEs that dynamically self-organize, and interact, to the extent of providing the desired functionality in a situation-aware way with no, or very limited, configuration efforts. Figure 1 depicts the architecture of the Framework. Applications are supported, underneath the topmost application layer, by a set of tools and services to enhance specific properties, such as situation-awareness, semantic self-organization, adaptive QoS, and security. These are contained in an intermediate layer that, in turn, relies on standard lower level infrastructures. The intermediate layer is fed with application-level and social-level knowledge, coming from continuous interactions with the upper level, to ensure adaptability and cross-layer tuning. The capability of dynamically influencing and controlling the behaviour of an application is guaranteed by the possibility of dynamically injecting in the middle level proper ACE components to exert such influence.

The framework is developed in a way to enhance a set of features that form the building blocks for provision of autonomic properties. In more detail:

- **Semantic Self-organization**. Self-organisation is one of the key design principles underpinning the autonomic management of large populations. This aspect is enhanced, in the Framework, by identifying and documenting local rule-sets capable of promoting the desired collective behaviour in a population of ACEs. This notion, generically called *aggregation*, includes algorithms for *clustering*, *differentiation*, and *synchronization* in groups of ACEs.

- **Situation-awareness**. System-wide autonomic behaviour requires the capability of detecting and interpreting changes in the execution context over time. This, in turn, requires leveraging assessed models of simple context-awareness, in which services access isolated pieces of contextual data, to a model of real *situation-awareness*, in which services access highly expressive structured knowledge related to a "situation". The Framework addresses this issue with the concept of *Knowledge Network* (KN), through which small bits of local information are correlated and organized into global knowledge through which the evolution of the system can be observed.

- **Self-healing**. Self-healing is intended as the enactment of means for management concerning detection of ACEs entering faulty states and the issuing of corresponding measures of reaction aimed at restoring an operative state. This principle takes advantage of self-organization mechanisms to configure and drive a control infrastructure offering self-management functions for a self-organized ensemble of interoperating components. In the Framework, the *Pervasive Supervision System* holds the role of such structure. Its main task is to monitor one or more ACEs to the extent of detecting faulty states and, upon detection, eventually restore it to its operative states.

- **Self-Preservation and Security.** As highly dynamic, an autonomic system poses a wide range of security issues. The approach towards these issues focuses on the provision of a framework for basic common security mechanisms, which can be aggregated in order to provide more complex mechanisms. In doing so, the integrated security framework effectively works in synergy with the supervision structure above, which provides an ideal ground for enhancing *confidentiality*, *integrity*, *authentication*, and *non-repudiation* features in a distributed fashion. Then, more sophisticated services can be provided through structured aggregation, in a way similar to self-organization principles, of these simple components.

2.1 ACE COMPONENT MODEL

The ACE forms the core of the CASCADAS Framework, and its component model enables the design of applications in a self-similar manner. Central to this, is the concept of *organ*. An organ is an ACE internal operative component and, as the name suggests, behaves as an organ in the human body. As such, it is capable of adapting its own execution to unforeseen events and, more in general, to the context in which the ACE is operating. Each organ is responsible for a specific type of tasks, and the interaction among them allows the constitution of an ACE as a standalone component. Assembling an ACE from a set of organs, each of which with clearly defined tasks, leads to a well-structured modularized component model depicted in Figure 2.

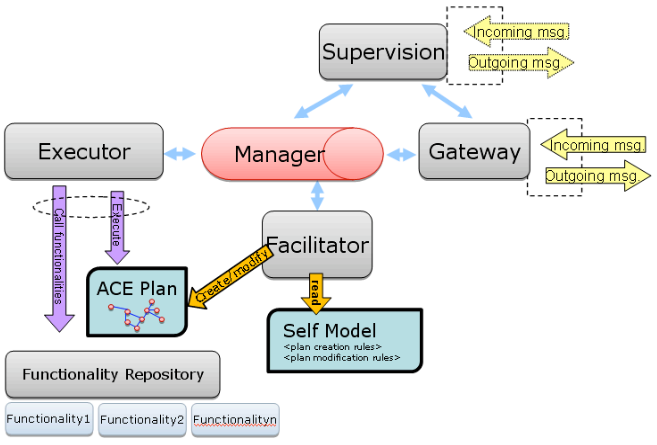

Figure 2: ACE Component Model

The *Gateway* organ is in charge of handling interactions with the external world. To this end, two different communication protocols are used: a connectionless protocol is used for initial service discovery through a publish-subscribe paradigm[13], supported by the *REconfigurable Dispatching System*[2] (REDS)**,** under the name of *GN-GA* protocol. On the other hand, a connection-oriented communication protocol is used for all other communications, where (one-to-one or one-to-many) communication channels are established through a contracting technique. The DIET [16] agent framework supports this communication type.

The *Manager* organ is in charge of handling internal communication among the organs and taking responsibilities for the ACE's lifecycle management. With respect to this activity, at any time an ACE can be in one of the four following life cycle states: *inactive*, *running*, *prepared to move*, and *destroyed*. As shown in **Figure 3**, different lifecycle actions are possible from these individual states.

Supervision features are brought, in the ACE component model, by a dedicated organ called *Supervision*. This organ is originally disabled, and its activation determines the nature of the ACE as supervisable (thus effectively enabling supervision). The organ is based on *Checker Objects*, which are delegated monitoring activities of specific points in the ACE and are consulted every time a monitoring event occurs. The event can be either accepted or denied, and this latter case may lead to an exception in the supervised organ.

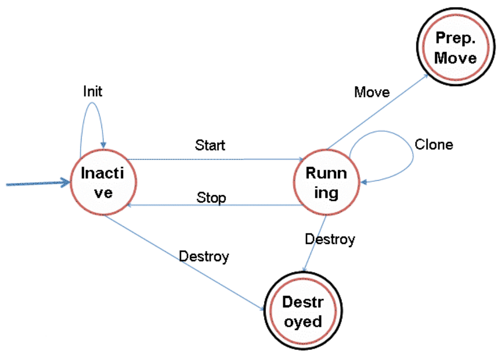

Figure 3: State diagram of all ACE organs for life cycle actions.

The *Facilitator* is the organ that provides an ACE with capabilities for autonomously adapting its behaviour to changes in the context in which it is executed. According to these changes, the Facilitator modifies the behaviour of the ACE by adapting the existing capabilities or adding new ones. The ACE's self-model provides the ground for achieving such self-awareness by formalizing the original behaviour and services, and providing the actions to be taken in order to modify the original behaviour upon occurrence of specific events. Internally, the self-model contains one or ore plans. Their semantic interpretation, performed at initialization phase, leads to creation of the original *ACE plan*, detailing which actions an ACE performs to reach its goals. Then, situation-awareness is enhanced by equipping each original plan with a set of *modification rules*, detailing the way the original plan needs to be modified when an ACE faces specific environmental situations.

The *Executor* organ governs the evolution of the ACE according to the actions taken as per self-model. Its main role is to ensure that any reasoning and decision taken by the Facilitator is put in place in an effective and efficient way by ensuring that conditions are verified, actions are executed and appropriate messages are exchanged.

Execution of plans may involve the use of specific functionalities contained in the ACE. To this extent, the Executor may query the *Functionality Repository* organ to obtain the needed ones. The purpose of this repository is store the functionalities deployed while also translating Executor requests into calls to functionalities through an invocation interface exported.

ACEs provide services in form of *functionalities*. These are stored in the repository, which is thus split into *Common* (i.e. guaranteed to be available in each ACE) and *Specific* (i.e. peculiar to a specific ACE) functionalities. Common functionalities are a major feature in terms of self-similarity. In fact, each and every ACE, regardless of its simple or aggregated nature, is capable of offering this set of capabilities albeit the set is strictly limited to functionalities covering basic operations. On the other hand, a service developer may create any arbitrary service and store it into the Specific Functionalities Repository to equip the ACE with more sophisticated and distinctive capabilities.

3. SEMANTIC SELF-ORGANIZATION

The Framework supports semantic self-organization through provision of algorithms for organized self-aggregation of ACEs. In particular, three (combinable) techniques are provided: *clustering*, *differentiation* and *self-synchronization*.

Clustering: These algorithms aim at enhancing ACEs with self-aggregation functionalities that allow the servicing of requests with the appropriate quality. The approach foresees an ACE initiating a “rewiring” procedure upon detection of a discrepancy in its list of required/available functionalities (this latter resulting from many different events such as the breaking of an existing collaborative link or a change in the local load due to a surge in demand). Depending on the circumstances, the initiator of the algorithm can choose one or more of its (contracted) first ACE neighbours as match-makers, and the constraint on the conservation of the total number of links can be relaxed or not.

Results from simulated scenarios (Figure 4) show that successful self-organisation can take place, at a predictable rate, provided that well-identified conditions are met. The resulting aggregate ACEs will reflect the presence of durable complementariness between functions provided/encapsulated by individual ACEs (i.e., long-lasting GN/GA matches) and their ability to collectively identify and realise these functional clusters through local interactions. This can be generalised to any complex web of interdependencies, with individual ACEs potentially belonging to more than one functional cluster, and including "single type" aggregates designed for load-balancing rather than complementariness.

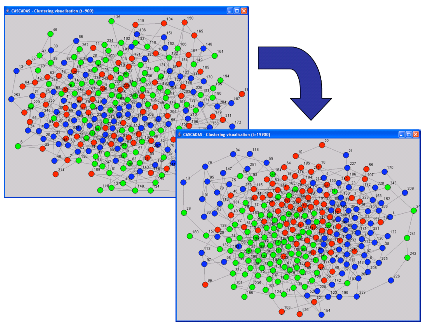

Figure 4: Self-organized aggregation through local rewiring with three ACEs.

Differentiation: Differentiation algorithms allow ACEs to decide to “self-terminate” locally when facing an inappropriate workload. This automates the transfer of resources, between applications and according to fluctuations in the demand for a variety of co-hosted services, in a way that resources released can be re-assigned to other applications. The name *differentiation* derives from biological morphogenesis with which it shares some characteristics. Algorithms integrated in the Framework enhance ACEs with capabilities to self-assess their own type according to the local observable workload, along with means for locally deciding when to change type.

Simulation results show that differentiation can help maximising the throughput of a distributed processing infrastructure while also making the system more responsive to heterogeneities in the workload when combined with the above “on demand” clustering. At the same time, results also emphasize that even the simplest set of rules tends to combine a large number of parameters to make interpretation difficult, therefore making selection of appropriate values a non-trivial problem.

Self-synchronization: Self-synchronization aims at finding or creating partnerships that adequately take into account the time activity pattern of individual constituting ACEs. This, in turn, involves (i) establishing a collaborative overlay that aggregates components featuring activity patterns that are a priori compatible starting from a random bootstrap configuration, and (ii) seeking ways of adjusting individual time-cycles so as to create opportunities for collaborations that would not exist if every individual activity pattern was "frozen" from the onset.

When the "rewiring" algorithm is employed, simulations show that the use random time signatures results in the formation of a scale-free overlay when pruning from a complete graph and trying to secure a target number of active neighbours at any one time. In addition, they also show that distributed heuristics based on the "on-demand" clustering algorithm[6] can be found so that near-optimal configurations obtained by pruning can be approximated.

4. SITUATION-AWARENESS

Situation-awareness refers to the ability of refining decisions according to the specific situation of the context the decision is influenced from, and will influence. In order for this capability to be enabled, vital appears to be the necessity of identifying models and tools for analyzing and organizing pieces of contextual information into structured collections, a *network of knowledge*. To this end, the Framework is equipped with a *Knowledge Network*[14] (KN) in charge of gathering and structuring information to form a collection of *Knowledge Atoms* (KAs), accessible by application-specific ACEs as a system-wide service.

The KN encapsulated in the Framework has been built as a collection of ACEs itself, and its architecture is depicted in Figure 5.

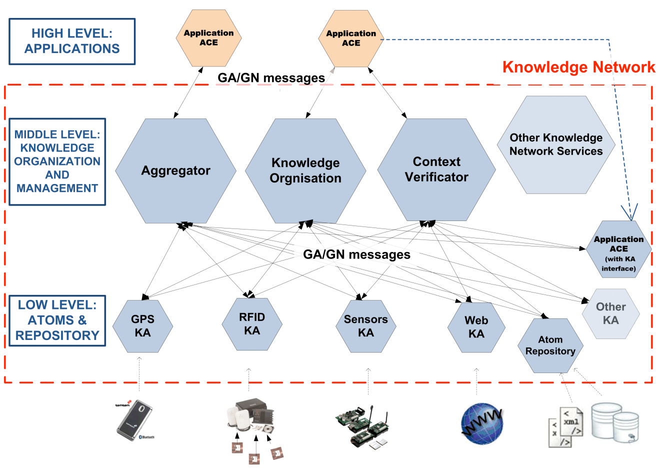

Figure 5: Architecture of the KN.

The bottom-most layer contains a number of ACEs realizing the concept of KA from heterogeneous data sources from GPS devices to the Web and system properties, with data sources having very limited power or being too dynamic and ephemeral represented in so-called *Knowledge Repositories*. This might be the case, for instance, of data from RFID readers and sensor networks, which can be accessed via a Repository rather than via individual KAs. It is worth noting that non-ACE applications may still be part of the Atom Repository through a simple interface.

On top of this layer, a number of components organize the KA so as to reify the concept of knowledge networks. The *Knowledge Organization* is in charge of organizing data in a network of knowledge containers and exporting an interface for concept-based querying (i.e., by keyword) enabling application-level ACEs, as well as any other interested ACE, to access concept-based information.

*Knowledge aggregator*s allow establishing meaningful relationships among KAs, also storing data in a very expressive format and processing upon request. Results can thus be made available via a pattern-matching interface *à la* Linda[1]. The idea behind the aggregator component is to provide specific information via pattern matching to form knowledge *views*.

The *Context Verificator* verifies consistency of information in the KN according to (application- or user-) configurable parameters. This is done by accessing the KA(s) under verification and/or querying knowledge organization components to verify consistency in the knowledge. According to the outcome of such verification, then, it can notify the application about problems.

4.2 THE W4 DATA MODEL

The KN data model has been engineered on the basis of the consideration that any bit of contextual knowledge is produced as a consequence of an event occurring in the context. Accordingly, a data model, named *W4 data model*, was created to represent any such fact, in a simple and expressive way, by means of a 4-tuple of the form (*Who*, *What*, *Where*, *When*): this 4-tuple represents the basic unit of information in KAs and allows to account an entity (*Who*) involved in some activity (*What*) at a certain location (*Where*) at a certain moment (*When*).

The use of the W4 data model enhances knowledge networking in a way to make it possible to identify relations between atoms on the basis of content similarity (e.g. two atoms having the "who" field set to the same value corresponds to facts related to the same entity). Once relationships between atoms are identified and organized, it is possible to process them, to produce views, on the basis of equality of values in the same fields for different 4-tuples. For instance, data from a sensor network in an environment can be "clustered" according to the geographic closeness of sensors so that a concise perspective on an activity occurring in region larger than the one sensed by a single sensor can be generated. For a wider range of examples and experimental results on these algorithms, we redirect the reader to [1], [14] and [15].

5. PERVASIVE SUPERVISION

Supervision refers to the ongoing observation of a configuration of ACEs and the issuing of corrective measures upon detection of the supervised configuration entering a state jeopardizing its effectiveness.

Automation of supervision functions (including capabilities aimed at limiting human intervention also under unpredictable conditions) addresses management capabilities (i.e., FCAPS - Fault, Configuration, Accounting, Performance, and Security), at different levels, up to a framework of ACEs. These, collectively referred to as *Self-management*, can benefit of self-organisation techniques, already employed in the framework, to enable highly distributed supervision of large collection of ACEs.

To implement this vision, the Toolkit enables a supervision approach based on two mechanisms:

- **ACE based supervision**: aims at a "service-oriented" handling of large collection of ACEs of different granularity (e.g., from “service capabilities” to “services” to “aggregate of services”); supervisors, i.e., aggregates of ACEs providing basic supervision functions, implement the so-called *pervasive supervision* service; they perform the continuous monitoring of ACEs grouped into meaningful clusters, and the enforcement of corrective measures, when required; supervisors are able to collaborate, so to achieve a *decentralized supervision*;
- **ACE tool-kit embedded supervision**: it accounts for the self-adaptive nature of ACEs behavior, all of which implement the same supervision logic. Their cooperation enables for highly distributed supervision logic. Clustering and differentiation techniques are considered to foster information exchange and support of application-specific logical neighborhood to the extent of detecting and reacting to events related to ACE’s behavior and their interactions through contracts; this cooperative approach enables for highly distributed supervision logic.

The two mechanisms should co-exist and co-operate.

Figure 6 shows the architecture of the pervasive supervision service, organized as an aggregate of ACEs, each of which implements a basic supervision function.

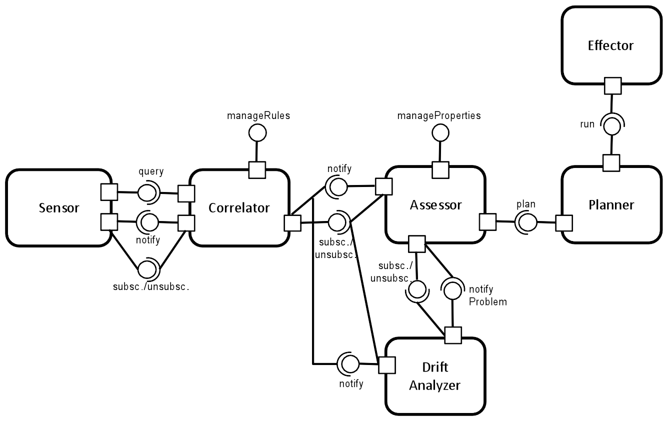

Figure 6: The pervasive supervision

The *Sensor* links the supervision system with the ACE configuration under supervision in order for this pattern to be monitored internally. Each ACE exports a management interface through which internal state and session objects can be accessed and verified.

Monitoring data gathered in this way is aggregated by the *Correlator,* to extract meaningful indicators of current health conditions of the system under supervision, and, in turn, analyzed by *Drift Analysers* that try to anticipate future problematic situations in the system under supervision. Additionally, information from the environment may be used to supplement the analytical process. The outcome of this analysis constitutes the input for *Assessors*, which make predictions on the current (or future) system health, on the basis of raw data or output of correlators and drift analysers.

The above components are concerned with detection of potential health threats for the configuration supervised. According to the outcome of this activity, a reaction is eventually decided and put in place as follows. The *Planner* tries to compute a solution plan for a detected problem, on the basis of the assessments generated by the Assessor and the actions described in the self-model of the ACE (or ACEs) under supervision. These are finally executed, through interception and modification of internal processes of the supervised ACE(s), by *Effectors*. The actual state of the system under supervision, its potential future behaviours and countermeasures on problem situations is derived from an analysis of the composition of the self-models of the ACEs that constitute the supervised system itself through the use of a mathematical framework for model composition, abstraction, and local refinement[9][10].

6. SECURITY AND SELF-PRESERVATION

The distinctive feature of a system built as a collection of ACEs is the absence of a centralized authority. As a consequence, a-priori trust relationships between ACEs belonging to different administrative domains cannot be assumed.

ACEs can show selfish, uncooperative or, in the worst-case, malicious behaviour. Therefore, it becomes of paramount importance to address security issues in two distinct directions to cope with this wide range of attacks. Indeed, albeit it is necessary to secure the communication infrastructure by providing confidentiality, integrity, and authentication, the behaviour of ACEs needs to be monitored. The Framework approaches these through a security framework enhancing *hard*- and *soft-security* mechanisms, concerned with cryptographic mechanisms and behavioural attacks respectively. In the course of its operations, the security framework exploits the supervision infrastructure, and employs self-organization techniques already presented, to guarantee survivability of the system while ensuring self-preservation of (both simple and aggregate) ACEs and resources.

Hard-security: Hard security mechanisms aim at protecting the system from threats such as impersonation, eavesdropping, spoofing, and data modification. However, deploying cryptographic algorithms directly on ACEs would make them cumbersome, thus potentially prejudicing execution on pervasive device. For this reason, the Framework exploits aggregation functionalities and the GN/GA communication protocol to provide security in an adaptive and flexible way.

Security functionalities are provided through dedicated ACEs, which can aggregate on a request basis in order to equip composite ACEs with self-preservation features.

The Toolkit integrates standard cryptographic libraries in the specific part of the Functionality Repository, which can be aggregated on demand in order to define more sophisticated security features. As an example, consider system where an ACE needs to aggregate in order to provide a sophisticated service. If, say, constraints on the bandwidth available are detected, it might decide to use simple, e.g. DES, encryption service. Then, if at a later time the constraint on bandwidth decades, the same ACE might want to employ a more complex encryption service, say AES with 256 bits key, which can be achieved by aggregating with another ACE providing the service. This solution enables for flexible adaptation of ACEs to the security requirements of the application and type of service by implementing reusability of components for different purposes.

Sofr-security: Giving ACEs responsibilities for the survivability of the system creates new challenges, whereby they are expected to react to eventual attacks, from malicious nodes targeting disruption of the system, by autonomously reconfiguring so as to exclude malicious ACEs.

Social control, in the form of trust and reputation mechanisms, is employed in the Framework to this extent. Heuristics, or aggregation functions, can be defined so that ACEs' behaviour can be captured and exclusion of malicious (selfish, or rational) entities can be enabled.

Social control enables analysis of system evolution and interactions, and techniques borrowed from the game theory are used to minimize uncertainty in service provision and composition. This allows deriving conclusions on system performance in presence of selfish or uncooperative ACEs, and the cooperation level when a reputation management scheme drives interaction decisions[7].

System-wide self-preservation can also be enhanced, through a similar analysis, by defining policies and strategies for the selection of specific partners on the basis of an evaluation of the level of involvement of an ACE. For instance, the use of such polices might enable an ACE to maximize its outcome in participating to system. This leads to an enhancement of the system survivability when the structure so formed is used to detect, and remove, malfunctioning components in order to keep the same service level[18].
Current development targets protection against Distributed Denial-of-Service (DDoS), perhaps the most difficult type of attack to deal with. Development of feasible protection mechanisms involving both detection and reaction are under development, where two possible response mechanisms derive from generic DoS detection schemes. The first allows ACEs to self-protect by filtering detected unwanted traffic, and it is applicable to DoS attacks within the ACE communication domain. The second, exploits service migration (self-configuration) to escape malicious flows.

7. A FUTURE PERVASIVE BEHAVIOURAL ADVERTISEMENT SCENARIO

The CASCADAS Toolkit described throughout this document has been successfully used to build a fully working prototype system to suit the needs of a future use-case scenario with a potentially immediate industrial spin-off. We named such scenario *Behavioural Pervasive Advertisement*.

This scenario considers a crowded venue with a number of public screens, used to advertise the venue itself as well as commercial advertisements. Examples of such advertising screens can be seen in public spaces, such as museums, airports, metro and train stations, etc. In such venues, it is realistic to assume the presence of infrastructures (i.e. wireless networks, RFID receptors, etc) that provide pervasive services (e.g. downloadable maps or events program for the venue, web navigation, etc.). Likewise, it is also realistic to assume visitors to be equipped with personal mobile devices (smartphones, laptops, PDAs, etc.) through which the above services might be accessed subject to registration or other means for storing publicly accessible information on user's interests (e.g. fidelity cards or badges).

As of today, advertising screens display information cyclically in a way independent of the context (i.e. independent of who is actually close to that screen). A smart service might exploit availability of the pervasive infrastructures, and the presence of pervasive devices, to gather publicly accessible information on users so as to adapt the contents to be shown on the basis of the peculiar interests of people detected. This would transform the advertising service with a set of desirable features in terms of effectiveness, as the exposition impact for the advertisement would be maximized, and business investment, as the level of guarantees provided on the effectiveness of the investment would be higher.

In the presence of a large number of screens and parties interested in buying time slots on them, solutions for allocating time slots and generating added value for interested parties must be identified. From this point of view, auctions appear an excellent solution as they prioritize allocation to advertisers who value them the most. Therefore, our platform employs an auction-based allocation paradigm whereby advertisers compete in a context-aware fashion in order to acquire the rights of advertising on a specific screen at a specific time.

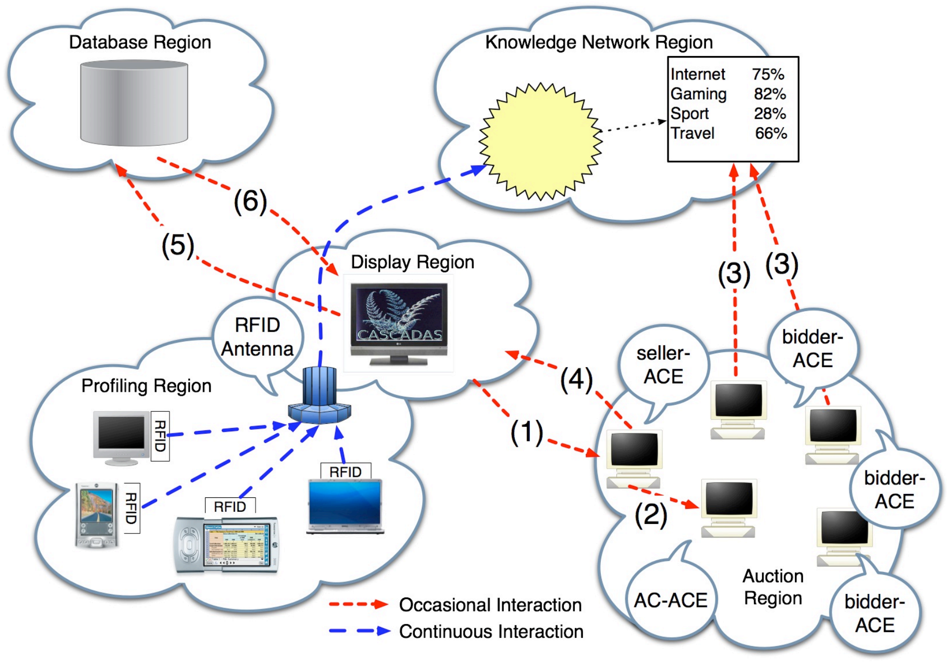

Figure 7: The Pervasive Behavioural Scenario

7.1 SYSTEM STRUCTURE

The platform we propose is a good example of ecology of services, or eco-system, where collections of ACEs are organized in *Regions* as shown in Figure 7. Each region makes available different services accessible system-wide, and cooperates to the extent of delivering the promised service in the promised terms and conditions. Referring to Figure 7, the *Profiling Region* makes available services for obtaining user information. This region is composed by a collection of ACEs truly interacting with badges equipped with RFID tags via RFID antennas positioned by the screen in such a way to detect the presence (and gather information) of people appearing and disappearing from the screen range. This activity is carried out on a continuous basis, as the shape of arrow denotes in the figure. Bits of information so gathered are exchanged with the *Knowledge Network Region*, whose ACEs give it the shape of structurally organized knowledge in the way previously described. This process outcomes availability of contextual information that other regions of the system obtain and use in order to acquire high degrees of situation-awareness.

The *Display Region* provides the system with displaying capabilities for the advertisement to be shown. Allocation of the actual content to be shown takes place by invoking, every 30 seconds and in anticipation to the start of the slot, the auction service the *Auction Environment* exports (interaction (1) in the figure). In this latter, an iterative English auction, lets ACE advertisers compete to the extent of acquiring the rights to expose own products in the slot of time under auction. Internally, the slot under auction is advertised by an *Auction Centre* (interaction (2) towards the AC-ACE in the figure) through which is made available to advertisers (i.e. bidder-ACEs in figure) whose decisions on whether to submit a bid or not make intrinsic use the querying services offered by the Knowledge Network Region (interaction (3)). Thus, a bid is submitted if and only if the trends of interests reported in the (situation-aware) result of the query show sensitive relevance with the range of products the advertiser is competing to display.

Upon auction termination, communication of the auction winner is returned back to the Display Region (interaction (4)), and is used by the ACEs there contained to select the right advertisement to display. The selection is offered as a service by the *Database Region*, which contains an advertisement database repository where advertisements are tagged based on owner and dominant relevance of interests. Thus, selection of the right advertisement is done by simply invoking the service with auction winner and the interest the winning bid was submitted for, returned by the Auction Region, as input (interaction (5) for the query, and (6) for the response).

The prototype platform has been fully developed and deployed on a number of distributed machines. Executions show that ACEs developed are stable and capable of supporting the interaction model described above in extensive long-term sessions. In addition, preliminary performance evaluation tests have been conducted to the extent of evaluating effectiveness of the platform in terms of impact of the advertisement currently shown on the current crowd as based on a matching of relevant interests. Results show that our system enables high impact on the current crowd, as compared with the classical "round-robin". In addition, analysis on such data allowed inferring that the *effective* investment cost for the advertiser, intended as the cost per-matching-person, decreases even though the price paid for advertising results many times increased (with a consequent increase in the screen owner's revenue).

8. CONCLUSIONS

This document describes the distributed component-ware framework for autonomic and situation-aware communication designed in the context of the CASCADAS project. The framework is capable of providing dynamically adaptable services through the innovative abstraction of Autonomic Communication Elements (ACEs) that are made available through a development Toolkit. This latter is fully usable and capable of exploiting highly innovative networking and application SAC services through provision of a number of features. Semantic self-organization allows aggregating and organizing aces on the basis of dynamic contextual changes; self-healing provides a distributed approach towards detection of changes in the health of one or more ACEs, while also anticipating reactions based on projections of the way the state will evolve; situation-awareness enriches the system with the ability to take into consideration the context evolution in local decisions by providing system-wide knowledge in a highly expressive and flexible format. The framework also addresses security features, where confidentiality, integrity, non-repudiation and authentication mechanisms are provided in form of on-demand services.

A demonstrator use-case was selected and fully developed as a collection of ACEs: the scenario anticipates a potential future industrial scenario, and relies on the sole CASCADAS framework on order to build a robust and effective platform in a fully distributed fashion. At present, this represents the only example of fully distributed platform relying on ACEs.

REFERENCES

[1] Ahuja S., Carriero N., Gelernter D., Linda and Friends, IEEE Computer, Vol. 19, No. 8, pp. 26-34, 1986.

[2] REDS – A Reconfigurable Dispatching System. In Proceedings of the 6th International Worksho on Software Engineering and Middleware (SEM2006), Portland, Oregon, USA. Available online: http://zeus.elet.polimi.it/reds.

[3] Bicocchi N., Mamei M., Zambonelli F., Self-organizing Spatial Regions for Sensor Network Infrastructures, Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops, Niagara Falls, IEEE Computer Society Press, CA, 2007, pp. 66-71.

[4] Cascadas Project "CASCADAS White Paper" D8.2, http://www.cascadas-project.org/Deliverable.html

[5] Cascadas Deliverable D5.3 "The Open Toolkit for Knowledge Networks"

[6] Cascadas Deliverable D3.4 "Synchronization"

[7] Cascella R. G., "The "Value" of Reputation in Peer-to-Peer Networks" in Proc. of the Fifth IEEE Consumer Communications & Networking Conference (CCNC 2008), Las Vegas, NV, USA, January 10-12, 2008.

[8] Castelli G., Mamei M., Zambonelli F., "Engineering Contextual Knowledge for Pervasive Autonomic Services", To appear in *International Journal of Information and Software Technology*.

[9] Deussen P. H.. Model based reactive planning and prediction for autonomic systems. In *Proc. INSERTech (Innovative SERvice Technologies), workshop co-located with: Firs tInternational Conference on Autonomic Computing and Communication Systems (Autonomics 2007)*, Rome, Italy, October 2007. ICST. Published on CD.

[10] Deussen P. H., Valetto G., Din G., Kivimaki T., Heikkinen S., and Rocha A.. Continuous on-line validation for optimized service management. In *EURESCOM Summit*, 2002.

[11] Di Ferdinando A., Rosi A., Lent R., Zambonelli F. and Gelenbe E.. "A Platform for Pervasive Combinatorial Trading With Opportunistic Self-Aggregation". *To appear* in Proceedings of the 2nd IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications, Newport Beach, CA, USA, 2008.

[12] Di Ferdinando A., Lent R. and Gelenbe E.. "A Framework for Autonomic Networked Auctions". In Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems (AUTONOMICS07), Rome, Italy, 2007.

[13] Eugster P., Felber P., Guerraoui R. and Kermarrec A.-M.. "The Many Faces of Publish/Subscribe". ACM Computing Surveys, 35(2):114-131, 2003.

[14] Greer K., Baumgarten M., Mulvenna M., Curran K., Nugent C., Autonomic Supervision of Stigmergic Self-Organisation for Distributed Information Retrieval. In *Proceedings of the Workshop on Technologies for Situated and Autonomic Communications* (SAC), December 10-13, Budapest, Hungary, 2007.

[15] Greer K., Baumgarten M., Mulvenna M., Curran K., Nugent C., Knowledge-Based Reasoning through Stigmergic Reasoning, In *Proceedings of the International Workshop on Self-Organising Systems* IWSOS'07, In: David Hutchison and Randy H. Katz (Eds.), Lecture Notes in Computer Science, LNCS 4725, Springer-Verlag, 240 – 254, 2007.

[16] Kephart J. and Chess D.. The vision of autonomic computing. IEEE Computer, 36(1):41–52, 2003.

[17] Marrow P., Koubarakis M., Van Lengen R.H., Valverde-Albacete F., Bonsma E., Cid-Suerio J., Figueiras-Vidal A.R., Gallardo-Antolin A., Hoile C., Koutris T., Molina-Bulla H., Navia-Vasquez A., Raftopoulou P., Skarmeas N., Tryfonopoloulos C., Wang F. and Xiruhaki C.. "Agents in Decentralised Information Ecosystems: the DIET Approach". In *Proceedings of the Artificial Intelligence and Simulation Behaviour Convention* (AISB 2001), York, UK. Available online: http://diet-agents.sourceforge.net

[18] Vassilakis C. C., Laoutaris N., and Stavrakakis I., "The impact of playout policy on the performance of P2P live streaming," in Proc. of SPIE/ACM MMCN '08, San Jose, California, Jan 2008

This research was supported by the European Commission under the FP6 IST-FET Integrated Project Cascadas.