# Deliverable 8.2

# CASCADAS White Paper: Bringing Autonomic and Situation-Aware Communication Services to Life

| | | |
|---|---|---|
| Status and Version: | Final | |
| Date of issue: | 10.04.2007 | |
| Distribution: | Project Internal | |
| Author(s): | Name | Partner |
| | Antonio Manzalini (Editor), Rosario Alfano, Antonietta Mannella | TI |
| | Franco Zambonelli | UNIMORE |
| | Peter H. Deussen | FOKUS |
| | Fabrice Saffre | BT |
| | Ricardo Lent | ICL |
| | Roberto Cascella, Bruno Crispo, Roberto Battiti | UNITN |
| Checked by: | PM and Ricardo Lent | TI, ICL |

## Abstract

The complexity of modern networks raises several challenges in the design and development of communication services. The unbearable costs in configuration and management call for autonomic approaches, in which services are able to self-configure and self-adapt their activities without human intervention. The need for ubiquity of service provisioning calls for the capability of services of adapting their behavior depending on the current situation in which they are used. In this paper, we discuss the need for innovative approaches facilitating the development and execution of autonomic and situation-aware services, and analyze the key features that should underlie such a general approach. Following, we overview how, within the CASCADAS project, we are trying to identify and develop such an innovative approach, by proposing an architecture centered around the novel abstraction of “autonomic communication elements” and by integrating in it proper tools for the autonomic, safe, and reliable management, of innovative situation aware services. Also, a short roadmap for future CASCADAS activities is sketched.

This document constitutes Deliverable 8.2 “CASCADAS White Paper”.

## Table of Contents

# 1 The CASCADAS Vision

The Internet as we know it today will have to become like an immense ecology of composite, highly distributed, pervasive, communication-intensive services [KepC03, Zam05]. Such services should be able to: (i) autonomously detect and organize the knowledge necessary to understand the general context – physical, technological, social, user-specific and request-specific – in which they operate; (ii) self-adapt and self-configure their functioning to get the best from any situation, so as to meet the needs of diverse users in diverse situation without explicit human intervention. These features will enable a wide range of new activities that are simply not possible or impractical now. For instance, we expect future generation of communication services to be able to:

- improve our interactions with the physical world by, e.g., providing us with any needed information about our surrounding physical environment and exploiting such information to adapt/enrich their behavior on the basis of the actual environmental characteristics (e.g., consider adapting the behavior of a tourist information service network on the basis of the location from which the service is invoked and of the current weather and traffic conditions) [Est02, Rav05];
- get the best of the network infrastructure and resources upon which they operate, being able to adaptively ensure sufficient quality of service, guarantee their security, and tune to user needs and preferences, independently of the actual network characteristics (e.g., independently of the fact that we require them from a Wi-Fi PDA in a MANET context, from a GPRS phone, from a Bluetooth eye-glass monitor, or from whatever connectivity and connected devices will be available at that time) [CapEM03, MikM04];
- facilitate our social interactions, by properly reflecting and exploiting the social context in which we are currently employing a service, e.g. for mere entertainment, or socialization, or in the context of business activities. Today, many opportunities for social communication and interaction are simply not realized due to a lack of information. Although acquiring and using that information raises security and privacy issues, their careful exploitation will open up a wide range of valuable possibilities for communication services (e.g. simply imagine a number of individual tourists that can be supported in forming a group to obtain discounts or other benefits) [ChoP03].

Turning the above vision into reality is very challenging. It requires a deep re-thinking of our current way of developing and deploying distributed systems and applications, i.e., by conceiving them as to be parts of a sort of ecology and by enabling them to prosper and thrive in it at the service of users. However, it is worth outlining that striving for the vision is not only a necessity for giving better services to end-users, but it is also becoming a compulsory economic urge for service providers and system managers. In fact, the increasing dynamism and variability of communication systems, due to the increasingly unreliable nature of communication links, network nodes, and service nodes (as induced by increasing decentralization and mobility) and to the increase in the number of means via which services can be accessed, calls for considering that applications offering services will likely need also to exploit knowledge and the lower levels, and that will be possibly supported by dynamically reconfigurable network components that can – at their turn – "understand" the implications of dynamic system changes on applications, and adapt themselves (and/or the overall network structure and policies) accordingly.

As challenging as this can be, proving that the above vision can be effectively realized is the key goal of the CASCADAS project (www.cascadas-project-org). *C*omponentware for *A*utonomic *S*ituation-Aware *C*ommunications *A*nd *D*ynamically *A*daptable *S*ervices), started January 1st 2006, and funded by the European Commission. Indeed, CASCADAS has to ambitious goal of defining a general-purpose paradigm for the development of autonomic and situation-aware communication services, and at showing its feasibility via the development of a general-purpose framework for supporting the process life cycle of autonomic services, and of associated application demonstrators.

In this white paper, we firstly intend to share the results of the thorough analysis work that, during the preparation and kick-off phases of the project, we have undertaken to reach an assessed and rationale understanding of several aspects related to the above vision. This include: *(i)* identifying a few guiding features – acting as drivers towards advancing the state of the art – that any new general-purpose proposal in the area of autonomic communication and self-adaptive services should properly provide (Section 2); *(ii)* identifying a unifying abstraction on which to base a new paradigm, the necessary software tools revolving around this abstraction, all of which framed into a practical reference architecture for the design, development, and execution of situation-aware and adaptive communication services (Section 3). Following, we intend to report on how the diverse yet integrated research thrusts carried on in the first year of the CASCADAS project are already contributing to bring autonomic services to like, that is, to turn the CASCADAS vision into a practical reality (Section 4), also with the help of specifically selected application scenarios (Section 5). Finally, future planned activities and the CASCADAS research roadmap are sketched (Section 6).

## 2 Founding Features

We have identified a few complementary founding features that we consider as general key enablers for the above vision, and around which any communication services infrastructures of the future should be conceived. The identification of these features starts from the 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 autonomic and situation-aware communication services vision. Thus: context-awareness must become situation-awareness; self-organization and self-adaptation must converge into a concept of semantic self-organization; scalability must assume the form of self-similarity; modularity must take the form of a new autonomic component-ware paradigm.

### 2.1 Situation Awareness

The capability of services to autonomously adapt to the context from which they are requested and in which they execute demands the technologies to capture contextual data and at the same time the ability of the system and of applications to effectively exploit this data at the best.

Much of the technology to acquire contextual information is already becoming available, and it will soon become pervasive with the increasingly frequent deployment of sensors, location systems, users and organization profiles, and run-time systems for the monitoring of computational and network resources [Est02, Phi04]. What is still in its infancy and still needs to be properly resolved, however, is the investigation of the principles and the algorithms with which this growing amount of distributed information can be properly organized, aggregated, and made more meaningful, so as to facilitate their exploitation by services.

In other words, we think there must be an evolution from a model of simple context-awareness, in which services are given access to isolated pieces of contextual data (as in the vast majority of current proposals), to a model of "situation-awareness", in which services are given access to properly elaborated and organized information representing, in much more expressive yet still simple to be exploited ways, comprehensive knowledge related to a "situation" [BouSZ05, Tum05].

## 2.2 Semantic Self-organization

There exist basically two complimentary approaches to enforce autonomic behaviors. On the one hand, self-adaptive systems work in a top-down manner. They have a sort of semantic representation of their state, and can evaluate their own behavior and change it when the evaluation indicates that they are not accomplishing what they were intended to do, or when better functionality or performance is possible. On the other hand, self-organizing systems work bottom-up without any high-level representation, based on a large number of components that interact according to simple and local rules and in which a global adaptive behavior of the system emerges from these local interactions.

Both self-adaptive and self-organizing approaches are being extensively studied [KepC03, BonDT99]. In our opinion, self-organization is preferable in highly distributed and decentralized scenarios. Self-organization and the algorithms underlying the emergence of adaptive patterns in complex systems have been extensively studied in communications, e.g., in P2P computing [BabMM02, Rat01], ant-based optimization [BonDT99], social networks [AlbB02]. Self-organization algorithms has the potential to act as enablers for service composition and aggregation, employing proven techniques to abstract from their "organic" implementation and derive design principles adapted to the requirements of artificial systems. At the same time, the presence of self-adaptive systems capable of understanding what's happening and proper reacting accordingly (as in the canonical "autonomic computing perspective [KepC03] can hardly be disregarded to ensure proper reactions and adaptations to various situations.

Accordingly, we think that a major advance with respect to most of the prior art is to provide a way to exploit self-organization approaches and enrich self-organizing components with more "semantic" and/or "cognitive" abilities, in the direction of self-adaptation. This raises the important question of evaluating the amount of information that has to be processed individually by system components, versus collectively by the self-organizing group. Our key goal is to preserve the simplicity and robustness of self-organization phenomena while simultaneously bringing the benefits of semantics self-adaptation and situation-awareness, to achieve what can be defined as "semantic self-organization".

## 2.3 Self-similarity

To realize the vision and make its embodiment manageable, any proposed approach must be fully scalable, i.e., its chosen design principles should be practically applicable to small systems (e.g., a few number of homogeneous nodes), as well as to very large systems (i.e., systems possibly made by thousands of heterogeneous nodes and service components).

While traditional approaches to distributed systems mostly focus on performance scalability, when the focus is on the development of autonomic services, one should also consider architectural scalability, i.e., the possibility for the adopted approach to scale in the without any increase in conceptual (and consequently in design and development) complexity.

In this direction, one promising option is to explore the potential of self-similarity, where any complex service can be realized by individual atomic components that self-organize and self-aggregate so as to reproduce nearly identical structures over multiple scales [AlbB02], and eventually to make an aggregated service appear again as if it were atomic. Self-similarity, which have been so far investigated only with regard to the structure and properties of complex social and technological networks [Dil03], may indeed represent be a key enabler also for the composition of complex communication services, as well as for the structuring of complex situational knowledge.

A successful use of self-similarity would carry on two closely related and complementary advantages: *(i)* it would facilitate understanding, description and management of services (due to the same structural and organizational principles being in force at different scales); *(ii)* it would allow "diving" into specific sub-systems whenever necessary, without having to modify abstractions and tools to work at finer levels of granularity.

## 2.4 Autonomic Component-ware

All the above features should federated by a sound "autonomic component" model [KepC03, LiuP04], which should provide both a robust and dynamic modular conceptual framework for building autonomic, self-organizing, semantic services, and to act as abstract and generic reference model for the production of a new generation of programmable communication elements that can be reused at different stack layers (i.e., for the implementation of communication services at both the network layer and at the application layers) .

This component model has to supply proper abstractions and tools to support self-similarity, self-organization and situation awareness. Therefore, autonomic service components will have to be explicitly conceived as situated in a world of situational knowledge, fitted with mechanisms for semantic self-aggregation and composition, and designed so as to promote the emergence of high-level ensembles that exhibit self-similarity independently of scale.

Identifying the specific nature and structure of such a dynamic autonomic component model is not an easy task. A number of and well-established research areas, such as multi-agent systems [ZamJW03], programmable networks, "traditional" component-oriented engineering, as well as more novel service-oriented architectures can provide useful insights and sources of inspiration, but requires leveraging the level of abstraction and the intrinsic support for dynamisms.

## 3 The CASCADAS Autonomic Service Framework

Identifying the basic along which to proceed and to advance the state of the art does not solve the issue of building a conceptual and practical framework supporting the design, development, and execution of communication services in line with such features. In the CASCADAS project, we propose to face this via the introduction of a specific software engineering abstraction, i.e., that of "autonomic communication element" (ACE), on which to rely for the flexible component-based design and development of any complex service, for the development of the associated supporting tools and, in the end, for delivering a well-architected framework for autonomic and situation-aware communication services.

## 3.1 The ACE Abstraction

The ACE abstraction represents the cornerstone of our component model, in which the four driving founding features will properly converge and around which proper tools can be developed (Figure 1). ACEs represent the basic unifying component abstraction on which to rely for the development of communication services. ACEs acts as entities that can implement (typically in a distributed way) communication services, and also act and are perceived as service access points.

While we expect service-specific behavior to be integrated in specific ACE classes, the basic ACE model has to integrate in all ACEs the capability to autonomously aggregate with each other to provide composite services at their best; it has to promote self-similarity in composition through a set of appropriate interfaces; it has exhibit self-organization capabilities, possibly of a "semantic" (i.e., meaningful) type; to this end, ACEs have to be both loci and consumers properly organized multi-faceted knowledge, coming from a variety of sources and sensors, overall leading to situation-awareness.

In CASCADAS, the ACE abstraction is the basis for implementing application-level communication services, as well as the basis on which to implement network-level and middleware level services. We will make (and are indeed already making) ACEs be able to operate with only a very minimal support infrastructure. This includes: the support for the automation of the ACE service life cycle, i.e., the post-development life span of communication-intensive services, including the autonomic and situation-aware deployment, the internal configuration of ACEs; the monitoring of its internal activities and the basic mechanisms for handling internal ACEs events; the provisioning of the basic mechanism to enable inter-agent communication (while the policies and the routing strategies for inter-ACEs communication are expected to be ACE-specific).

The specific nature of the basic ACE structure and of the basis protocols we envision for enforcing dynamic inter-ACEs interactions and aggregations, as being matured within the CASCADAS project, will be analyzed in Section 4.

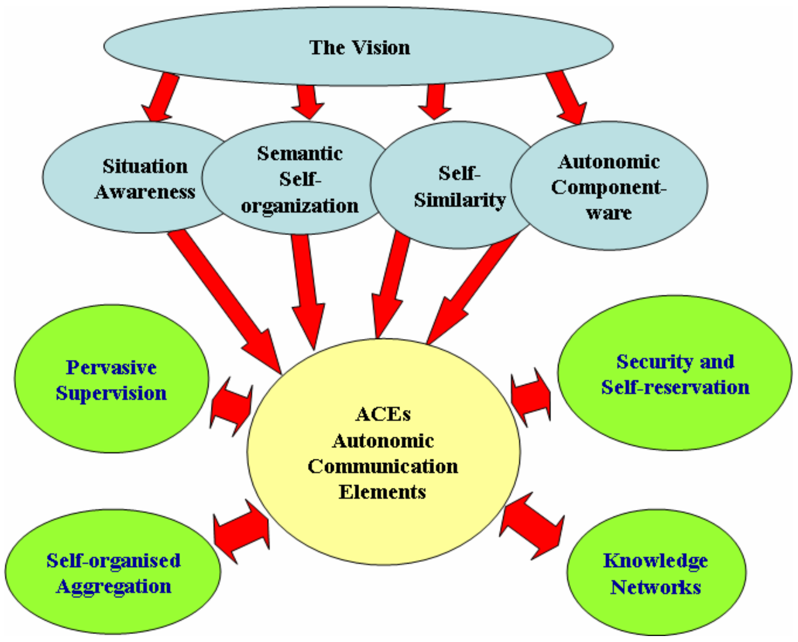
The Vision
Situation Awareness
Semantic Self-organization
Self-Similarity
Autonomic Component-ware
Pervasive Supervision
ACEs Autonomic Communication Elements
Security and Self-reservation
Self-organised Aggregation
Knowledge Networks

**Figure 1: ACEs as the Central Abstraction of a new Paradigm for Autonomic and Situation-Aware Services, Around which Proper Tools can be Organized.**

## 3.2 ACE-based Tools

The key idea of the ACE abstraction is that, beside the described minimal support, any kind of communication service can be implemented via proper dynamic composition of ACEs. In other words, we clearly expect that application-level ACEs will be provided with the necessary algorithmic tools, security tools, knowledge tools, and with any needed infrastructural services. However, we also envision that all of these tools and infrastructural services can be realized in terms of ACE-based services in their turn, and that all of them will lead to a practical and trust-worth paradigm.

While we expect to be able to implement any needed tools and services via ACEs (and the preliminary research results strongly support such expectation), CASCADAS researches is also focused on the definition and experimentation of a few specific services and tools that we consider of a basic paramount importance, and that will give, to different extents and with different flavors.

We are focusing on developing algorithms and techniques to achieve dynamic QoS adaptation and enforce given service properties through automated aggregation of ACEs. Smart aggregation will be the basis for identifying and exploring opportunities for co-operation within an ensemble of ACEs, which would allow the collective system to exhibit certain desired properties and to hit situation-dependent QoS targets. This research thrust is primarily relevant to the founding features of self-organization and situation awareness, and is expected to provide notable advances along these two directions.

We are developing models, algorithms, and tools for the self-organization, correlation and self-composition of contextual knowledge, according to which ACEs can exploit all the available information about their situation, however sparse and diverse. Situation is intended here as a generalization of context, relating to both *(i)* the social-organizational context from which services are invoked (i.e., by a specific users living in a specific social context and accessing the network with specific devices and network technologies); *(ii)* the technological and physical environment in which ACEs live and execute, primarily their networked environment. This research thrust is obviously primarily relevant to the principle of situation awareness, and will contribute notable advances in the direction of context-awareness, but also represents a ideal substrate to study and experience innovative self-organization algorithms.

We are developing pervasive supervision functionalities across an ensemble of interacting ACEs. Pervasive supervision addresses the runtime construction of an ad hoc and dynamic runtime structure that encompasses a set of cooperating ACEs, and exerts a fully automated and de-centralized control of the communication-intensive service provisioned collectively by those ACEs. This research thrust is primarily relevant to the advance the state of the area in the directions of self-organization (specifically, self-management) and scalability/self-similarity, although it also strictly related relates to situation-awareness.

We are studying and developing innovative security techniques, an aspect which is of paramount importance because of the very assumptions upon which the idea of ACEs relies: the heterogeneous nature of the network, the varied capabilities of ACEs, their ability to self-organize and cooperatively supervise each other, which implies the lack of centralized administrative control. Since an ensemble of ACEs possesses those highly dynamic adaptation characteristics, we intend to exploit them to make sure that the resulting system is highly robust and secure, and trust-worth. This research thrust is primarily relevant to the founding features of self-organization and situation-awareness, and is expected to provide notable security-oriented advances in this direction.

## 3.3 The Architectural Perspective of the Framework

To acquire a more "operational" perspective of the CASCADAS approach and of its objectives, one can refer to Figure 2, sketching the overall architecture envisioned for our ACE-based framework.

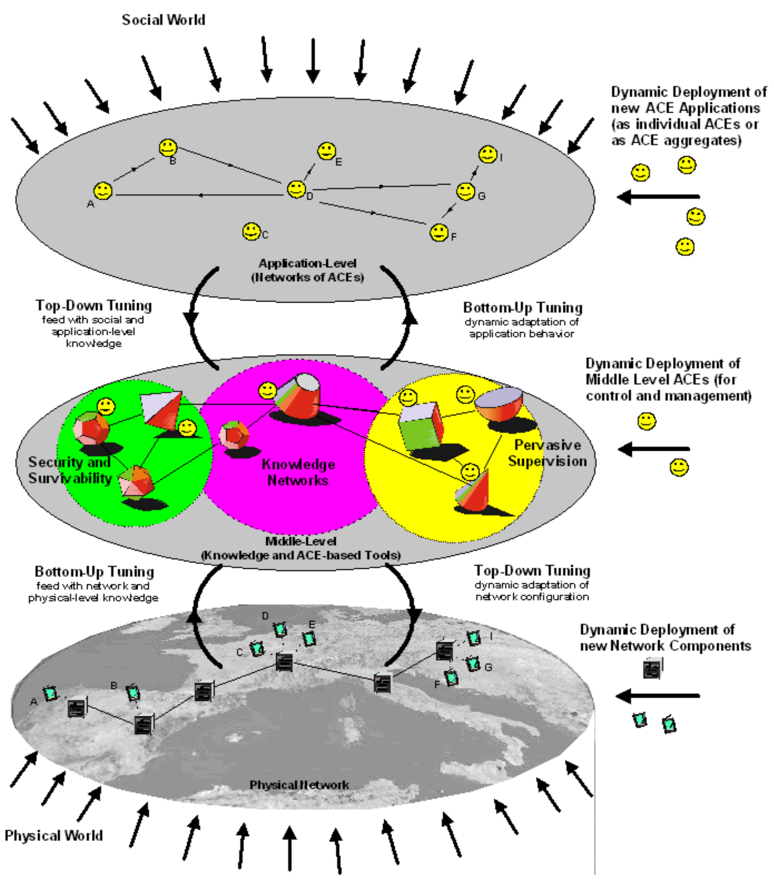
Social World
Dynamic Deployment of new ACE Applications (as individual ACEs or as ACE aggregates)
A
B
C
D
E
F
G
I
Application-Level (Networks of ACEs)
Top-Down Tuning
feed with social and application-level knowledge
Bottom-Up Tuning
dynamic adaptation of application behavior
Dynamic Deployment of Middle Level ACEs (for control and management)
Security and Survivability
Knowledge Networks
Pervasive Supervision
Middle-Level (Knowledge and ACE-based Tools)
Bottom-Up Tuning
feed with network and physical-level knowledge
Top-Down Tuning
dynamic adaptation of network configuration
Dynamic Deployment of new Network Components
Physical Network
Physical World

**Figure 2. The Reference Architectural Perspective of the CASCADAS Framework**

From the hardware/infrastructural viewpoint, CASCADAS considers a scenario in which dynamic and heterogeneous networks, possibly enriched with sensors and devices connecting with the physical world, have to host the dynamic deployment and execution of applications and services. Although CASCADAS researchers does not actually deal with hardware network architectures and with physical sensing and embedded systems, CASCADAS will indeed account for the network and the physical levels in terms of the information that, from such level, can reach the higher levels and can be exploited to enforce situation-awareness.

Applications and services have to serve users according to both their social situation and the current network and physical situations. To this end, at the higher-levels, CASCADAS considers developing and deploying application and services (by individual users as well as by software companies and system managers) in terms of ACE components or of ACE aggregates. These components – supported by a proper autonomic framework for ACE management – can dynamically self-organize as needed with each other and with the already deployed ones, and can start interacting so as to provide the desired functionality in a situation-aware way without (or with very limited) configuration efforts.

Below the application level, a sort of "middle-level" set of tools and services has to be provided to all applications. These include services to access contextual knowledge (properly organized and aggregated into sort of "knowledge networks"), and ACE-based tools to enforce specific properties such as situation-awareness via knowledge networks, semantic self-organization, adaptive QoS, and security. This middle level is fed both by application-level and social-level knowledge (coming from the upper levels) and by network-level and physical-level knowledge (coming from the lower levels), and continuously interact with these levels, in a sort of continuous tuning feedback that ensures adaptability and, thanks to the connection with the lower-levels, also cross-layer tuning. The power of dynamically influencing and controlling the behavior of the network and of the application is guaranteed by the possibility of dynamically injecting in the middle-level proper ACEs components to exert such influence.

## 4 Bringing Autonomic Services to Life

The CASCADAS project started in January 2007 and, as it always happens in such large coordinated projects, we had to spend some time to assess a globally shared understanding of the above described vision, and in reaching some global agreement on terminologies and research methodology. After that, actual research has proceeded very fast and, after a year, it  has produced several relevant research results, which we try to summarize in the following of this section. Of course, we are still unable to report well assessed and fully integrated research results. Yet, all the reported results, as preliminary and apparently fragmented as they can be, have been achieved with the same overall vision in mind, and each of them is already contributing its bit in bringing autonomic services to life.

### 4.1 The ACE Internal Model

As already stated, the whole of the CASCADAS project is based on the grounding idea to identify a general component model (the ACE) for the design and development of autonomic communication services. Services at both the user and the infrastructural level can then be developed by the self-organisation of ACEs performing specific functions (e.g. invoking telco/ICT enablers and using data, information, etc.), and able to dynamically interact/aggregate/compose with each other without central coordination (i.e., in a peer-to-peer way). Accordingly, one of the key activities for the first year of the project has related to identifying the general characteristics and architecture of such a general component model, thus paving the way for its future implementation.

The key contrasting goals we have tried to achieve in the identification of such general model have been generality and simplicity. On the one hand, the model should be general enough to meet the diverse needs of a variety of services. On the other hand, the model should be kept as simple as possible, also to make it possible its implementation for resource constrained devices. By recalling that (see Subsection 3.1) an ACE should include both a common part, shared by all ACEs, and a specific part, to implement within an ACE specific functionalities, facing the trade-off between generality has implied a careful analysis of what to include in the common part and what to include in the specific part.  Such an analysis has been performed, also by taking a careful look at the state in the area. As a result of this analysis, the general architecture of the ACE model has been defined (see Figure 3 and, for further details, [Hoe06]).

Basically, in the identified architecture, the common part of an ACE is identical for each type of ACE and contains a minimum set of fundamental capabilities to enforce self-management capabilities, while the specific part contains additional functionality that is required for solving specific tasks (i.e., specific operations and/or protocols) and can be different for different classes of ACEs. Let us know quickly summarize the key components of the ACE architecture.

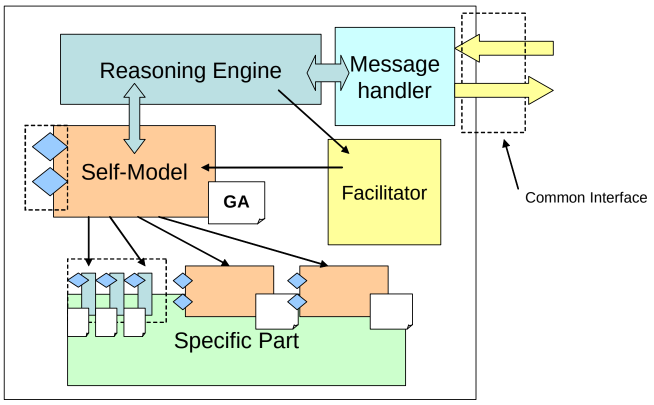

**Figure 3. The ACE architecture**

The *Common Interface* is basically the way ACEs communicate and interact with the world outside (i.e., other ACEs or the environment). The communication is message-based and therefore, the Common Interface is implemented by a Message Handler that is able to understand a fixed set of messages. The ACE collaborations and aggregations are exclusively carried out by the exchange of these messages.  The Common Interface rely on a basic set of messages that implement an innovative protocol (the GN/GA protocol) for ACEs discovery and aggregation. It defines two classes of messages: *(i) Goal needed (GN)*, a sort of request, with a semantic description attached, which specifies what kind of functionalities the ACE needs from other ACEs to achieve its goals; *(ii) Goal Achievable (GA)*, used by an ACE to state what kind of task it is able to provide. Two or more ACEs can get in touch with each other upon matches of a GN message with a a GA message. In other words, the GN-GA protocol (which is detailed later in section 4.2) is a semantic advertisement protocol by which ACEs advertise their capabilities through the GA message.

The *Self-Model* describes the possible states for the ACE and the possible transitions between pairs of states. In other terms, it could be defined as a state machine. Therefore, the Self-Model is a description of the steps the ACE will execute to achieve its goals. Any state is described by a semantic description used by the ACE to reason about its current state with the help of a reasoning element. The transition functions are the specific features, which must be invoked during a state transition. The ACE Self-Model is published outside by using the GN – GA protocol, i.e., a semantic advertisement protocol by which ACEs advertise their capabilities.

The *reasoning engine* executes (the implementation of) the self-model and its main role is to keep trace of: the state reached in the Self-Model execution, also be tracing history of previous states; the environment and the contextual information, which include any GA coming from other ACEs as well as any "Knowledge Available" (KA) coming from elements implementing the concept of knowledge networks (see also Subsection 4.4). Mainly, it has to be able to run the state machine used to describe the Self-Model. It should check if a transition may take place invoking the proper specific features if specified, and it has to proper represent the semantic description of the new state reached.

The *Facilitator* is be the core autonomic part of the ACE, adapting its behaviour to the changed conditions, situations or faults. The adaptation of the behaviour means changing the self-model state machine when a specific feature exhibits different behaviour depending on its state.

As far as the specific part is concerned, it contains the ACE specific functions. It exposes these functions through the Specific Interface. The Specific Interface contains a description of each function of the ACE Specific Part: the set of features, which characterize the ACE behaviour. For each feature a semantic description of the job the ACE is able to do (GA) and the indispensable and essential actions, conditions to accomplish it (GN) are specified. For example, given a specific function which executes a query on a DB containing personal data information, the semantic description of the GA could be something like: "get people profile" and the GN could be something like: "a connection with a database is necessary".

At the same time, we have started implementing a first prototype of a toolkit to act as a distributed framework for ACE development and execution. We expect such an implementation to release, in the next few work, a framework for the development of ACE-based autonomic services, there included services for implementing knowledge networks and supervision functionalities. In parallel with this activity, we have developed several pencil and paper application exercise to test (at least conceptually) the effectiveness of the proposed architecture, and we are following with great attention the specific algorithms for ACE aggregation and for enforcing security that will be soon an integral part of the overall ACE framework.

## 4.2 The GN/GA Protocol

As anticipated in the previous Subsection, ACEs discover and self-aggregate with each other on the basis of a simple GN/GA protocol relying on two classes of messages "Goal Needed" (GN) and "Goal Achievable", made available by all ACEs in the Common Interface. We recall that a GN is a sort of request, i.e., a semantic description which specifies what kind of functionalities an ACE needs from other ACEs to achieve its goals, and that GA is a sort of advertisement that an ACE uses to communicate what he can do.

In general, the identified GN/GA protocols builds on the lessons of adaptive multicast protocols [EugG02] to discover communication partners and exchange messages in open and dynamic networks. The key idea behind is that, if a component knows what it need but it does not know (and does not care) how to get it, it has to dynamically search in the network until it finds something to satisfy its needs. Thus, by searching in the network, a need can be eventually satisfied despite the network is continuously changing and despite the fact that one does not know communication partners.

The specific solution adopted by CASCADAS for the GN/GA protocol, however, does not consider flooding a network with GN requests. Rather, it considers propagating GA advertisements, according to specific policies, so that a requester does not have to propagate GN request, but can look for matches locally. More in details, the interaction model works in the following way:

- If an ACE is able to do something (expressed by means of GAs) and it is ready to do it, it advertises this information to all its neighbours (with the exceptions mentioned later, the information is flooded throughout the network).
- When an ACE needs something (expressed by means of GNs), it doesn't need to start any search. The ACE only needs to check if someone has already advertised a feature (GA) which can address its need: this mechanism may be based on a blackboard metaphor.

To some extent, we can describe the key aspect of the interaction model using a simple metaphor: the ACE interaction model is fully based on a "pull" semantic, which generates a sort of "altruistic environment". Each ACE is altruistic: if it is able to do something and it is available to do it, the ACE will propose its help to other ACEs as much as possible.

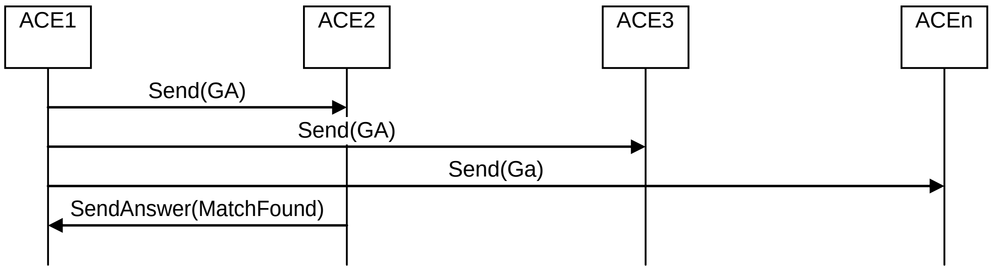

**Figure 4. Call-flow for the GN/GA Protocol.**

The key challenge in the GN/GA protocol is to ensure that the information about ACEs availability and capability reaches the proper ACEs in a proper time, and possibly without fully flooding the network with GA requests. In order to achieve this, we think it is needed to define a P2P protocol aimed at diffusing the information about ACEs capability and availability in a proper time and without unacceptable overhead. To limit the flooding overhead of GAs messages, we plan to enforce is the implementation of a sort of semantic time-to-live: if the incoming GA (i.e., the features described) belongs to the same semantic domain of the receiver ACE then the GA is propagated; otherwise, it is discarded as it is possible that ACE’s neighbours may not interested in that GA. Figure 4 is a simple example of a call-flow with a proposal sent by ACE1 (i.e., GA) to a certain number of ACEs and the possible dialog between the ACE1 and an ACE receiver (e.g., ACE2) which needs the advertised features of ACE1. The ACE2 discovers a semantic matching of a received goal-achievable with its goal-needed, it sends back a kind of acknowledgment to the ACE1. Also, the strong involvement of local aggregation algorithms should avoid overhead due to GAs flooding.

The mechanism described above is based on the following main assumption: if the GA received does not pertain to the semantic domain of the receiver ACE, it would be highly probable that none of the receiver's neighbours are interested in that GA, so in most cases it is better not to forward the message.  The semantic domain is defined by all the ACE where the GN/GA matching is satisfied. The interaction model outlined in this section is mainly based on the assumption that if two ACEs are closed (using a proper distance heuristic) we can state that the two ACEs deal with correlated topics. The assumption above should allow facing one of the key challenges in avoiding centralised control: limiting the overhead due to the interactions between ACEs needed to reach an agreement and to plan the right organisation. The implementation of a “semantic time to live” (STT) is needed to avoid that each time an ACE advertise its goal this is forwarded to any other ACEs without any filtering. This STT works on the following assumption: if an ACE receives a GA related to its GN, it would be proper to forward that information to neighbour ACEs.

Although some preliminary conceptual and simulated experiences show the potentials of the identified GN/GA protocol, further studies are needed to demonstrate its practical implementation feasibility and its performances, and to possibly identify more suitable solutions to avoid flooding and enforce effective matches. Also, we plan to evaluate how and to which extent the propagation of GA can be supported by knowledge networks, i.e., by having algorithms for knowledge propagation and diffusion be exploited also for the propagation, diffusion, and evaporation of GA advertisements.

## 4.3 Self-organized ACE Aggregation

Self-organisation is one of the key design principles underpinning the autonomic management of large populations of ACEs, because it is the only practical alternative to central control. However, engineering such self-organizing behaviours presents a variety of challenges, which we are addressing in two complementary ways: *(i)* by identifying and documenting local rule-sets capable of promoting the desired collective behaviour in a population of autonomic service components, a notion that we generically call "aggregation" and that includes both clustering, differentiation, and synchronization, in groups of ACEs; *(ii)* by conducting a quantitative analysis of macroscopic dynamics in a large population of autonomic elements, i.e., by using modelling and simulation techniques to extrapolate from the local rules governing individual behaviour. The value of the above methodology has been confirmed by the studies performed so far in CASCADAS. Experiments have indeed revealed that extremely simple local rules (candidates for expressing global self-organized aggregation behaviours in ACEs) had "hidden" properties susceptible to lead to inoperable global configuration, and thus a principled study of system properties has to be an integral part of the process of engineering the behavioural repertoire of individual components. The main conclusion is that, if the objective is to design a set of local rules capable of scaling up to a large, fully decentralised population of such components, failing to conduct such a preliminary, model- or simulation-based investigation can have catastrophic consequences, even when small-scale experimental deployment revealed no unwanted properties.

Coming to the specific of the algorithms studied in the first year of CASCADAS, the study has mostly concerned clustering algorithms for the autonomic, self-organized management of resources in large colonies of ACEs, i.e., algorithms to have ACEs devote to provide specific functionalities being able to self-aggregate with other ACEs so as to properly serve requests with the needed quality. Among a variety of scheme studied, our findings lead us to recommend using "on-demand" clustering algorithm whenever constraints dictate that only strictly local messaging between first neighbours (i.e., information transfer without any dedicated messaging infrastructure - either centralised or distributed - or forwarding capability) is available. This will of course not always be the case for ACEs, but can be regarded as the most challenging scenario for a rule-set relying on self-organisation to promote the emergence of the desired system configuration (due to the strict locality of information and absence of explicit long-range interactions) and so constitutes the ultimate test of robustness.

In practice, this means that whenever operating under these extreme conditions, an ACE following the "on-demand" clustering algorithm (see Figure 5) would initiate a rewiring procedure as soon as it detects a discrepancy between its "Goal Needed" (GN) and "Goal Achievable" (GA) lists of required/available functionalities. This situation can result from many different events like, e.g., the breaking of an existing collaborative link (GA $\rightarrow$ GN), the submission of a new type of request (additional GN), a change in the local load (GA $\rightarrow$ GN, due to a surge in demand leading to the current collaborative relationships being no longer able to absorb the corresponding workload). Depending on the circumstances, the initiator can choose one or more of its first neighbours (ACEs with which it has an existing relationship) as match-makers, and the constraint on the conservation of the total number of links can be relaxed or not. But fundamentally, our work demonstrates that successful self-organisation would take place, at a predictable rate, provided that well-identified conditions are met (most importantly in terms of the diversity of "goals types", which must be low compared to population size and of the same order of magnitude as the average node degree). The resulting "ACE aggregates" will reflect the presence of durable complementaries 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. For instance, those ACEs of type "x" frequently needing access to a functionality associated with type "y" will identify this dependency and materialise it by attempting to establish a preferential relationship with an ACE of that type, giving rise to an appropriate number of "x-y" pairs. This can of course 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 complementarity.

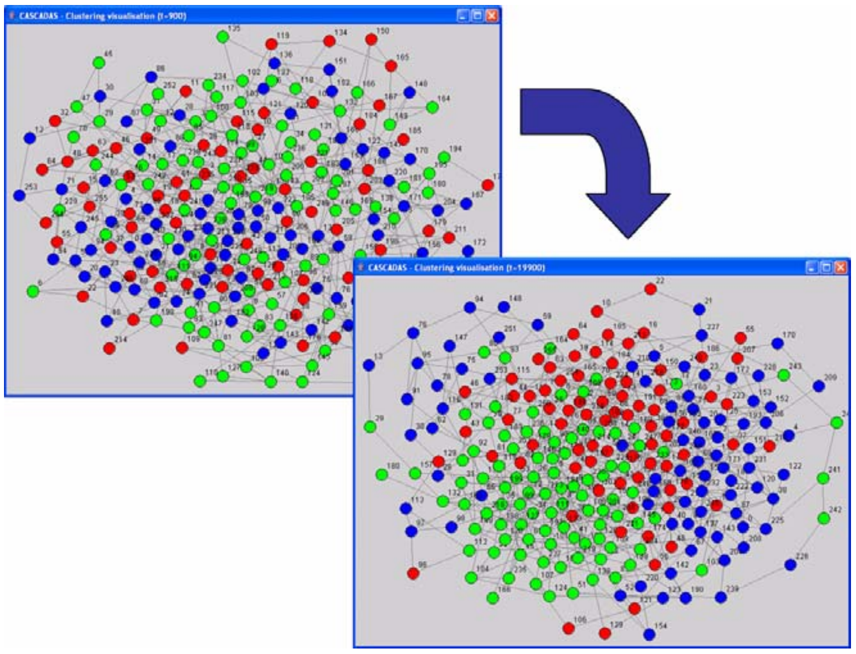
CASCADAS - Clustering visualisation (t=900)
CASCADAS - Clustering visualisation (t=19900)

**Figure 5. Self-organised aggregation through local rewiring ("on-demand" clustering). Simulation example featuring three color-coded classes of ACEs providing different functionalities.**

The natural next step for our studies is to include deliberate changes of local characteristics into the "algorithmic toolkit" and investigate their effect on system properties. The underlying concept is that individual autonomic components are not necessarily predetermined to accomplish one (or a limited subset of) task(s). There are many ways in which autonomic elements will be able to tune their internal state so as to adapt to their environment. For instance, an operating system may be able to choose whether to load or clear individual software modules, therefore determining the ability of a processing unit to perform certain tasks. Finally, synchronization, which can be regarded as a special case of aggregation, will also need to be part of any usable toolkit designed to engineer self-organising systems. The overall objective is to develop decentralised techniques of coordinating the activity of individual components so that their activity cycles reflect their interdependency. An obvious example would be the emergence of de-synchronization between units sharing the same resource, so as to minimize conflicts in the absence of central orchestration. In the context of pervasive computing, where battery-powered devices are playing a central role, the emergence of synchronization between units that cannot afford to be "always-on" will also be critical to the emergence of workable partnerships.

## 4.4 Knowledge Networks

CASCADAS considers ACEs as entities that needs to access contextual knowledge and adapt their behaviour (whether individual or collective) accordingly. In CASCADAS, we envision knowledge will be accessed by ACEs on the basic of a specific flavour of the GN/GA protocol, i.e., a "Knowledge Needed"/"Knowledge Available" protocol, and in the assumption that the Knowledge Available with be provided by specific ACEs in charge of managing contextual data. However, as stated earlier in this paper, CASCADAS also considers that there must be an evolution from a model of simple context-awareness, in which services are given access to isolated pieces of contextual data, to a model of "situation-awareness", in which services are given access to properly elaborated and organised knowledge (i.e., knowledge networks) representing, in much more expressive yet still simple to be exploited, comprehensive knowledge related to a "situation".

The construction of a single knowledge network capable of mirroring the universal situational knowledge is illusionary, because of the large amount of information available, and because of the different needs (in terms of knowledge representation/aggregation) that may be exhibited by different applications and services. Accordingly, CASCADAS considers a reference architecture for knowledge networks in which a multiplicity of application-specific knowledge networks can be built from the same raw data and can co-exists to serve the different needs of different applications and services, i.e., can provide different knowledge views to requesting ACEs, as from Figure 6. In different knowledge networks, data can be organized along spatial and/or temporal relations, we well as along any desired semantic relations. Although the identification of the knowledge reference architecture can be considered as a research result per se, more tangible research results has been achieved in trying to turn the abstract architecture into a practical reality. In particular, with regard to: *(i)* the definition of detailed specifications (fitting the reference architecture) for knowledge network components and their aggregation, together the implementation of a simple knowledge network software for testing these concepts; *(ii)* the study of several algorithms for knowledge aggregation and distribution, and their evaluation via simulation. The details of these two research threads can be found in [Bau06] and [BicMZ07] respectively; here we only summarize the key points.

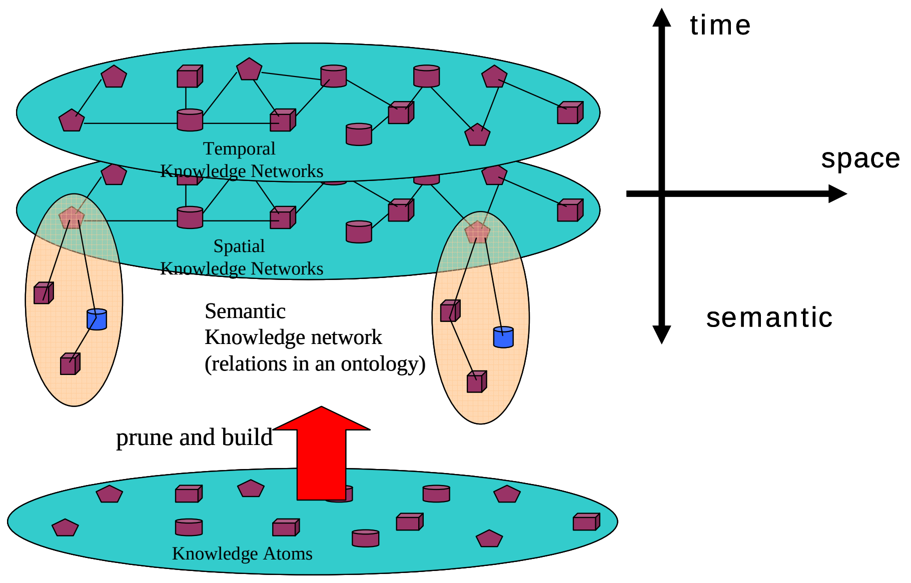
time
space
semantic
Temporal Knowledge Networks
Spatial Knowledge Networks
Semantic Knowledge network (relations in an ontology)
prune and build
Knowledge Atoms

**Figure 6. A Reference Architecture for Knowledge Networks.**

With regard to knowledge network specifications, they have been organised around the basic concepts of knowledge atoms, knowledge containers, and their organisation [Bau06]. Knowledge atoms represent the most basic components of knowledge networks (the atomic data items). A knowledge container is instead a structure capable of encapsulating knowledge at different levels of granularity. The purpose of a knowledge container is to organize multiple knowledge atoms in a semantic, spatial, or temporal fashion, and to providing access to the underlying information. Knowledge containers, being perceivable as composite knowledge atoms, can further be aggregated and organised by high-levels knowledge containers. Knowledge atoms and knowledge containers fits the general reference architecture, thus resulting in an overall scenario for knowledge networks that is coherent with the overall CASCADAS framework and that promote component-ware and self-similarity.  The alpha software developed to test the above concept has the form of a general-purpose Web tool for flexible manipulation of knowledge networks components. The tool, fully based on modern dynamic Web technologies (i.e., XML-RPC), implements the concept of “knowledge repository”, intended as a general server to which data coming from the sensor level (in the form of knowledge atoms) can be collected and aggregated (via knowledge containers) to form any needed knowledge networks that services may need to access via proper “Knowledge Needed” requests.

The study of algorithms for knowledge aggregation has focused on the definition and simulation of advanced self-organisation algorithms to achieve self-aggregation of sensorial data in sensor networks [BicMZ07]. the idea underlying our proposal is that of delegating to the sensor network the execution of distributed gossip-based algorithms that – by continuously running in the network as a sort of background noise with bounded energy costs – can enforce:  *(i)* the adaptable self-partitioning of the network into spatial regions characterized by similar patterns for sensed data, via the self-organization of an overlay network; *(ii)* the distributed aggregation of whatever sensorial data on a per-region basis. As a result of this process (whose effectiveness has been extensively studied via simulation) the sensorial data generated by the sensor network is no longer perceived as a multiplicity of unrelated (and difficult to be analyzed) data items. Rather, the algorithm makes it possible to perceive the sensor network as if it were made up of a more limited number of “macro sensors”, each associated to a well-characterized region of the physical environment (i.e., a region exhibiting a uniform pattern for some specific property such as a light, temperature, etc.). To some extent, the algorithm provides for the automatic construction of a knowledge network aggregating data and facilitating its usage by services.

Despite the very encouraging achievements of the first year, there is still a research and development work to make the idea of knowledge network a general and usable tools for the provisioning of situational information to autonomic services and, specifically, to ACEs. First, there is need to study more general and flexible mechanisms for knowledge manipulation. While the first year has mostly focussed on mechanisms for organisation of knowledge around the spatial dimension, the second year will also focus on the semantic dimension (i.e., organising knowledge network based on semantics relations between components) and on the temporal dimension (i.e., organising knowledge network based on the temporal relations so as to enforce the possibility of providing services with a sort of “predictive situational knowledge”). Second, we expect to notable extend the knowledge network software, making it fully integrated with the ACE-based toolkit currently under implementation, and making it rely on an ACE-based implementation of knowledge atoms and knowledge containers.

## 4.5 Pervasive Supervision

With the Autonomic Computing Initiative, IBM introduced MAPE (monitor—analyze—plan—execute) as a control paradigm for autonomic systems [KepC03]. MAPE boils down to a feedback control loop that continuously perceives the state of a system and interacts with it. Consider *autonomicity*. If MAPE is used as an architectural paradigm used during the design of a system, and is the element that "adds" autonomicity to the very system, then the data to be gathered during monitoring, the analysis and planning algorithms, and the control functions to be executed can be defined in the development phase of the system. But if the controlled system itself is autonomic, then it is pretty unclear how to define a control loop that deals with a system that is essentially designed to operate without external control. Consider *self-organisation.* Having a system that composes itself in an automatic way from a set of available (but not necessarily pre-defined) components (which might be itself complex, self-organised ensembles), it is by no means clear which data are relevant for control purposes, how to evaluate (or even to define) their state, and how to interact with systems which are dynamically changes their inner composition. The crucial point here is that there is no *a priory* knowledge available to effectively define control purposes and tasks. As functions and structures of self-organised systems *emerge* rather than follow a pre-defined "architecture", associated control functions (and structures) have to *co-emerge*. In summary, the closed control loop paradigm is not sufficient to exemplify autonomic systems.

In the context of the CASCADAS project, and for the start up of the activities related to pervasive supervision, we have tried to analyze and unfold the above issues and identify a reasonable framework for pervasive supervision. In our view, supervision has to be defined as a service, i.e., an activity which is performed by some entity (namely: the supervision system) by executing a certain activity (e.g. improved fault tolerance, SLA validation) on a supervised service. Hence, we may talk about a supervision contract between supervision system and system under supervision. The ability of being supervised (i.e., being perceivable and controllable) is an integral property of a system that is able to commit supervision contracts. However, the extend to which an ACE is willing (allowed, designed) to disclose internals and permit external control depends on the ACE itself, its purpose, service model, security policies, etc.

The generic nature of contractual supervision requires novel mechanisms for system perception and actuation; in particular, there is no definite list of attributes that defines the structure of a supervision contract because of the autonomicity and self-organisation abilities of supervised service configurations. In principle, service configuration need to be open in the sense that behaviour aspects are perceivable and controllable (although we are aware of the security issues raised). On the other hand, complete openness is not what is really needed as information on the level of concrete executions and concrete system states are far too fine-grained for effective supervision. Thus the image that a supervision system maintains about the supervised service configuration is an abstraction, an *operational model* of it. Such a model describes the usage protocol (order of operations/messages, exceptions, states, etc.), the exchanged data and data types, constraints, etc. Concerning this, we may state: *A supervision contract is about the validation of the system under supervision against the operational model associated with a service configuration and the enforcement of this model.*

A basic assumption is that ACE configurations are structured by mechanisms like aggregation, and self-organized clustering or composition. Thus, the operational model of an ACE configuration needs to reflect the structure of this configuration. In order to define notions like composition (or the embedding of a system component into a larger system context) and abstraction (and thus hierarchical aggregation), a key results of the first year of activities has been the development of a rigorous mathematical framework for modelling pervasive supervision. The notion of a "zoom" has been derived, i.e. a local refinement of a system model with respect to a single component abstraction. Zooms form the basis of the definition of hierarchical contingency plans, which are applied on each structural level of the supervised system during the actual supervision activity. Furthermore, the mathematical framework allows has been used to define metrics to assess the "competence" of the supervision system with respect to the supervised systems like effectiveness, timeliness, and appropriateness [Deu06b].

In the upcoming project phases we will address the following issues: (i) the definition of cascaded hierarchical control loops and their implementation in a generic, ACE based software architecture. Using competence metrics, the original steps defined in the MAPE approach are complemented by a "validation and assessment" activity. Furthermore, we will investigate languages and frameworks for the definition of supervision models, abstractions and system compositions. *(ii)* Supervision based on reactive and pro-active planning is in many cases to heavy-weighted to be effective. The employment of "reflex" mechanisms based on local observations and reaction patterns that work in real-time are more desirable in those contexts, but contradicts the requirement of generic supervision discussed above. We will investigate means to derive those "autonomic reflexes" in form of conditional contingency plans from supervision models. *(iii)* Long term supervision, on the other hand, is performed as an off-line activity that takes place in the background of the actual service execution. We have started to develop a framework for the detection of drifts of the "concepts of interests" in the behavioural patterns of a system under supervision and its environment, with the purpose to enable for pro-active adaptation of the system under supervision to changing requirements [Deu06a].

## 4.6 Security

Classical approaches to realize security services such as *confidentiality*, *integrity*, *authentication*, and *non-repudiation*, base on the underlying assumption on a centralized definition of a trust relation is available, i.e. that the set of permitted users is known, and access rights are unambiguously defined. Autonomic communication services are characterised by the involvement of heterogeneous entities without centralised organisation or control, ranging from single users to entire institutions, to coalitions of institutions that do not necessarily belong to the same organisation, nor share a single authority. Thus, an a-priori trust relation between them is not available. Trust between nodes has to be built through specific mechanisms tailored to the challenging scenario offered by an open environment. An overall security framework for the CASACADAS has to address basic security functionalities to enable secure interactions and at the same time provide advanced security functionalities to enable soft- and self-management of the entire system.

Classical security solutions aim on malicious entities that try to break the proper system operation and intentionally cause damage. On the other hand, a self-interested entity does not intend to directly damage the overall system functioning, but is unwilling to spend its resources (bandwidth, storage, energy, etc.) on behalf of others. From this kind of misbehaviour, a new class of problems, that we group as non-cooperation problems, arises. Lack of cooperation has gained much relevance in the context of self-organising systems and several techniques akin to game theory (GT) and mechanism design (MD) have been proposed to counter this new type of threat. GT is used for predicting the outcomes of such interactions, whereas MD is used for creating the appropriate contexts that will lead selfish users to behaviours that are in alignment with the preferences of some "designer", which typically assumes the role of leveraging and cultivating the so-call "social-interest" (or "good", or "utility", or "well being"). Details on this approach can be found in [Lao06].

Another approach in the field of soft security mechanisms, which is also called social control mechanism, is based on trust and reputation systems. Trust and reputation systems are useful tools for deriving provision trust, protecting in that way the relying parties from malicious or unreliable service providers. Provision trust is users' knowledge about the reliability of authenticated parties, or the quality of goods and services that they provide. In other words, provision trust describes the relying party's trust in a service or resource provider. Due to the creation and maintenance of provision trust in a system, participants rely on "future payment", which guarantees (creates to the participants a sense of security) that the investment the participants have done when they shared their resources, will be returned to them. Therefore, the participants are willing to contribute their own resources, because they feel sure that this effort (and attitude) will be recognized and rewarded. Without provision trust, converge to that behaviour would not be possible.

Based on these two approaches to *self-preservation*, a security architecture for the CASCADAS project has been sketched in the first year of the activities, and it will be further refined in the upcoming project phases. The main purpose is to construct a trust framework so that components can authenticate and interact in a secure way. Since, in an autonomic and distributed environment we cannot rely on a central authority or on a pre-defined secure infrastructure, this trust relationships must be determined based on ad hoc security relationship which can take advantage from interactions established in the past or by means of a third temporal trusted entity or on opportunist and ephemeral way.

We assume that security services are placed in specific ACEs, which implement basic functionalities. Security services can be defined as cryptographic operations or algorithms as well as advanced services for self-preservation and self-management of the system. According to the capabilities of the nodes and on their goal in the network we identify three different classes of security ACEs that have specific and elementary functions to exploit reusability and context adaptation of the ACE itself. The first class of "security" ACEs consists of nodes that do not participate into the creation of a security service as aggregators since these components do not have sufficient computational capabilities to handle the orchestration of the execution of cryptographic functions. This function is proper of the second class of components that have the role of providing basic cryptographic functionalities to implement security into the CASCADAS framework. The third class identifies the advanced components that will provide mechanisms for self-management and self-preservation of the system. Their role will be to monitor and to self-preserve the CASCADAS network from internal or external attacks.

## 5 Putting Autonomic Services at Work

CASCADAS models and tools promise to be useful for a wide variety of application scenarios, ranging from smart environment, urban computing, wide-area P2P computing and distributed resource sharing. To better direct efforts without loosing on generality, we have decided to focus on two complimentary classes of application scenarios, very different from each other: services for pervasive computing and distributed auction systems.  These two scenarios will be useful to show – via the implementation and the demonstration of specific services within – that CASCADAS and all associated tools to be integrated in the framework promise to be general-purpose and effective.

## 5.1 Innovative Services for Pervasive Computing

The pervasive computing scenario envisioned in CASCADAS considers an environment which is densely enriched with sensorial and computational capabilities. In particular, sensor networks as well as RFID tags are embedded in the environment, and can act both as sources of environmental data as well as a sort of environmental computing infrastructures. Sensors can interact with each other in an ad-hoc way, with some users nearby, and can possibly (not necessarily) be connected to some “sink-server” that can be used to collect data. RFID tags can be accessed by nearby users and devices to read and/or store data. Humans too populate the above scenario, live in it and interact with each and with each other. It is expected that users carry with them some kind of mobile devices (e.g., smart phones and/or PDAs). Via such portable devices, users can be given access to the information produced by nearby sensors and tags, can possibly interact with each other in an ad-hoc way (e.g., via bluetooh or WiFi), and can possibly access to the Internet via some wireless connection.

Most observers agree that the above scenario – which is already getting shape – will become increasingly pervasive in a few years, opening the door for the deployment of a wide range of pervasive services. At the infrastructural level one can think at localization services for users, cars, devices, and at a variety of routing services to deliver data and messages across the pervasive network such as services capable to deliver messages at specific locations in the physical environment or to multicast messages at specific groups of nodes or users. At the user level, one can think at various services to query the physical world around, at services that have the world adapt to the user, or at services to alleviate traffic congestion in the cities by involving devices on cars, sensors on streets, computer-based devices in stree lights, etc..

Whatever the specific service one focus on, developing and making available high-quality services in that scenario is very challenging and definitely calls for property of autonomicity and situation-awareness. Just consider the case in which a set of distributed advertising screens in a city have to adapt their content (i.e., decide what commercial clip to show) on the basis of the actual preferences of the users in their proximities (as stored in users’ personal mobile devices).

Service components (i.e., ACEs allocated on screens and on users’ mobile devices) may have to face incredible bursts of usage (hundred thousands of persons all in a place), and incredible load unbalances (during the interval of an exhibition, most people move to bars) some of which totally unpredictable (a specific exhibition at a place has a great success and attract more people than expected). Accordingly, they must be able to properly tune both their internal functioning, calling for autonomic behavior inside ACEs, as described in Subsection 4.1). However, this also calls for adaptable self-organizing behavior at the inter-ace level (as described in Subsection 4.3) to avoid ACEs operations to unbalance computational and communication load, and the capability of continuously monitoring the supervising the overall behavior of the ACEs (as described in Subsection 4.5). The service overall requires the capability for ACEs to interact in a highly dynamic and mobile environment, calling for flexible discovery and interaction protocols (e.g, the GN/GA protocol sketched in Subsection 4.2). Clearly, the amount of data to be processed by ACEs to get decisions may be overwhelming (hundreds of thousands of user profiles and localization information), calling for proper tools for aggregating data and make available to ACEs a pre-digested analysis of the situation (as it can be provided by the knowledge network approaches of Subsection 4.4). Last but not least, the commercial value of advertisement calls for proper security mechanisms (as from Subsection 4.6) to avoid any fraud.

In general, most of pervasive services that one can think of deploying in the sketched scenario challenges traditional service approaches, and will give us a chance to evaluate and demonstrate the effectiveness of the CASCADAS tools under development. Incidentally, the described service for adaptable pervasive advertisement is one of the two that will be most likely implemented to demonstrate CASCADAS results and the effectiveness of the autonomic service framework.

## 5.2 Distributed Autonomic Auction Systems

The second scenario selected by the CASCADAS project considers a future global economy organized around networked auctions between enterprises and between individuals and enterprises [Gel06]. In this scenario, auctions will be completely automated by being conducted by software entities acting on behalf of the user. Auctions might have very short duration in time, with some of them possibly finishing within seconds. In this scenario, delays as small as few milliseconds might cause considerable losses. Moreover, the evolutionary nature of the market often creates situations where the same good is auctioned by more than one auction, some of which under more convenient conditions (consider for instance holiday packages, where the same conditions have different prices based on the offering agency).

The idea of autonomic distributed auctions is that to allow the management of networked auctions in a fully autonomic fashion. Software entities should be made able to manage many concurrent auctions on behalf of the user by autonomically deciding what to do and when, to the extent of fulfilling the overall user aim. In addition, failures by infrastructures, or even unwanted delays in the auctioning process will be avoided by autonomically moving in the background network. For instance, consider a company whose business is in the fast changing market of technology. If the company sells technology goods such as, for instance, mp3 players and game consoles, it is desirable for its warehouse to be able to sustain a fast turnover of goods, where older models will be replaced by newer ones upon release. Autonomic auctions will be an effective way to manage goods in the warehouse, selling older ones and acquiring newer ones without human supervision. The process of acquiring new wares through auctions will be conducted by the software entity in charge of management by acting autonomically based on knowledge of the environment the entity interacts with.

Clearly, the aforementioned scenario requires a robust communication and service infrastructure capable of providing differentiated levels of quality of service to its users and to deliver, in a reliable and timely manner, a large number of data packets. Of course, this scenario will benefit greatly from an autonomic communication framework able to operate in an opportunistic way and that offers self-* features by exploiting situation awareness, as CASCADAS aims to offer. More in particular, the software entities to implement auctions will be realized in the form of ACEs (as from Subsection 4.1) interacting and discovering with each other via the GN/GA protocol (as form Subsection 4.2). In this way, ACEs participants in an auction will be able to internally supervise their own activities and will be able to dynamically discover potential partners in a flexible and semantic way. Also, auction participants will be allowed to virtually move (migrate to a different ACE), and possibly self-aggregate with each other (as from Subsection 4.3), to gain differential advantage and also enabling the a higher-number of concurrent auctions to be handled. The knowledge needed to effectively issue a service migration and properly selected the best target location can be determined by accessing knowledge networks (as from Subsection 4.4), where all relevant information will be properly made available in a compact and efficient way. All of which, properly supported by supervision tools (as from Subsection 4.5) capable of providing self-optimizing and self-healing features at the system-level, and by distributed security tools (as from Subsection 4.6) capable of attack and malicious communications detection.

In summary, autonomic distributed auctions will be an excellent use-case to demonstrate the effectiveness of the CASCADAS framework for wide-area network applications.

# 6 Conclusions and Roadmap

Competition in the telecommunication market and traffic growth will determine the need of enhancing current infrastructure whilst reducing costs in order to maintain the telecommunications business sustainable. In order to match these requirements, Network/Service Providers are evaluating to evolve service platforms introducing service-awareness and autonomics features (e.g. self-healing/self-protection, self-optimization).

The purpose of embracing those research thrusts in the CASCADAS project and bringing that kind of advancement to the area of communication-intensive services is multifold. In the first place, and with the shortest-term outlook, we aim at overcoming service platforms main bottlenecks and simplifying the handling of, interconnection of and interaction with the portfolio of existing communication-intensive services, reducing labor and costs. Furthermore, we aim at facilitating the assembly and management of new types of services that are currently too complicated or costly to implement, because of their complexity and dynamism. In the longer run, we aim to help laying some of the necessary foundations and mechanisms that will enable the construction of a repertoire of innovative services that cannot yet be envisaged in the current communication environment, but that will become part of the fabric of a Connected Society in the years to come.

CASCADAS started from a clear vision of what requirements a general approach to situation-aware and autonomic communication services should meet and of what advances over the state of the art were needed. Also, it has a clear architectural vision to drive the different research activities of the projects. Starting from these solid grounds, activities in the first year of CASCADAS have produced interesting research results, there included:

- the identification of a suitable architecture for general-purpose autonomic communication elements and of a suitable protocol for their interaction/aggregation;
- the study and simulation of several self-organizing algorithms for self-aggregation of services;
- the identification of a general architecture for managing situational data and of several algorithms for knowledge organization and aggregation
- the identification of a general architecture for pervasive supervision and the development of a corresponding mathematical framework;
- the identification of a general security architecture and the study of several innovative soft-security solutions.

In addition to the above, a specific thread of activities has been devoted to study and analyze the potential socio-economical impact of future autonomic communication scenarios, with a specific attention devoted to the autonomic communication services scenarios envisioned by CASCADAS. Such study, which will continue over the full duration of the project, is very precious to direct the activities of the project and to acquire an understanding of the socio-economical implications of its research findings, other than of their scientific implications.

As preliminary and non-well integrated all these results and studies can be, they lay a solid ground for the next phase of the project, which will be specifically focused on integrating these results towards the release of a first functional toolkit for the development and execution of autonomic communication services. In particular, activities in the second year of the project will be organized as follows:

- A first version of the ACE-based toolkit, including basic tools for creating and deploying ACEs and having them interact, and made available to the project researchers;
- Such toolkit will be exploited to implement and release specific libraries of ACEs devoted to implement specific self-aggregation algorithms, knowledge network services, pervasive supervision services, and security services.
- Eventually, at the end of the second year, all these implemented tools will form the first release of the ACE integrated toolkit, whose effectiveness will be demonstrated via the implementation of two simple application demonstrators.

In parallel with the above implementation-oriented activities, CASCADAS researches will continue with more scientifically-oriented activities devoted to study and analyze the basic principles of autonomic and situated communication services. These will include further studies on autonomic component models and their interactions, studies of algorithms for component self-aggregation and for knowledge management, studies on pervasive supervision models, and studies on innovative security solutions and algorithms for autonomic communication services (as discussed in Subsections 4.1 to 4.6).

In summary, we believe that CASCADAS is on the right directions to fulfill its promises and for bringing autonomic and situation-aware communication services to life.

# 7 References

[AlbB02] R. Albert, A. Barabasi, "Statistical Mechanics of Complex Networks", Rev. Mod. Phys. 74(47), 2002.

[AndS04] S. Androutsellis-Theotokis, D. Spinellis, "A Survey of P2P Content Distribution Techniques", ACM Computing Surveys, 36(4):335-371, Dec. 2004.

[BabMM02] O. Babaoglu, H. Meling, A. Montresor, "Messor: Load-Balancing through a Swarm of Autonomous Agents", 1st Workshop on Agent and Peer-to-Peer Systems, 2002.

[Bau06] M. Baumgarten, N. Bicocchi, K. Curran, M. Mamei, F. Zambonelli, M. Mulvenna, "Self-organizing Knowledge Networks for Smart World Infrastructures", 2nd International Conference on Self-organization in Multiagent and Grid Systems, Erfurt (D), Sept. 2006.

[BicMZ07] N. Bicocchi, M. Mamei, F. Zambonelli, "Self-organizing Spatial Regions for Sensor Network Infrastructures", 2nd IEEE Symposium on Pervasive and Ad-Hoc Communications, IEEE CS Press, Niagara Falls (ON), May 2007.

[BonDT99] E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence: From Natural to Artificial Systems", Oxford University Press, 1999.

[BouSZ05] P. Bouquet, L. Serafini, S. Zanobini, "Peer-to-Peer Semantic Coordination", Journal of Web Semantics, 2(1), 2005.

[CapEM03] L. Capra, W. Emmerich, C. Mascolo, "CARISMA: Context-Aware Reflective mIddleware System for Mobile Applications", IEEE Transactions of Software Engineering Journal, 29(10):929-945, 2003.

[ChoP03] T. Choudhury, A. Pentland, “Modeling Face-to-Face Communication Using the Sociometer”, ACM Conference on Ubiquitous Computing, Seattle (WA), 2003.

[Deu06a] Deussen, P. H., L. Baresi, M. Baumgarten, M. Mulvenna, C. Nugent, K. Curran; Towards Pervasive Supervision for Autonomic Systems; IEEE 2006 Workshop on Distributed Intelligent Systems; Prague, Czech Republic, June 2006.

[Deu06b] Deussen, P. H., “Supervision of Autonomic Systems”, International Conference on Self-Organization and Autonomous Systems in Computing and Communications (SOAS’2006), Erfurt, Germany, Sept. 20, 2006.

[Dil03] S. Dill, R. Kumar, K. Mccurley, S. Rajagopalan, D. Sivakumar, A. Tomkins, "Self-Similarity in the web", ACM Transactions on Internet Technology 2(3):205-223, 2003.

[Est02] D. Estrin, D. Culler, K. Pister, G. Sukjatme, “Connecting the Physical World with Pervasive Networks”, IEEE Pervasive Computing, 1(1):59-69, Jan. 2002.

[EugG02] P. T. Eugster, R. Guerraoui, “Probabilistic Multicast“, Proceedings of the 2002 international Conference on Dependable Systems and Networks, IEEE CS Press, Washington (DC), pp. 313-324, June 2002.

[Gel06] E. Gelenbe, “Analysis of Automated Auctions”, 16th International Symposium on Methodologies for Intelligence Systems, Bari (I), 2006.

[Hoe06] E. Hoefig, B. Wuest, A. Mannella, M. Mamei, B. Catalin Benko, E. Di Nitto, “On Concept for Autonomic Communication Elements”, 1st International Workshop on Modeling Autonomic Communication Environments, Dublin (IR), 2006.

[KepC03] J. Kephart, D. Chess, "The Vision of Autonomic Computing", IEEE Computer, 36(1), 2003.

[Lao06] N. Laoutaris, O. Telelis, V. Zissimopoulos, and I. Stavrakakis. Distributed selfish replication. IEEE Transactions on Parallel and Distributed Systems, 17(12):1401–1413, December 2006.

[LiuP04] H. Liu, M. Parashar, "Component-based Programming Model for Autonomic Applications", International Conference on Autonomic Computing, New York (NY), 2004.

[ManZ06] A. Manzalini, F. Zambonelli, “Towards Autonomic and Situation-Aware Communication Services: the CASCADAS Vision”, 1st IEEE Workshop on Distributed Intelligent Systems, Prague (CZ), June 2006.

[MikM04] M. Mikic-Rakic, N. Medvidovic, "Support for Disconnected Operation via Architectural Self-Reconfiguration", 1st International Conference on Autonomic Computing, New York, NY, USA, 2004.

[Phi04] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, D. Hahnel, “Inferring Activities from Interactions with Objects”, IEEE Pervasive Computing, 3(4):50-57, 2004.

[Rat01] S. Ratsanamy,, P. Francis, M. Handley, R. Karp, "A Scalable Content-Addressable Network", ACM SIGCOMM Conference 2001, Aug. 2001.

[Rav05] N. Ravi, P. Stern, N. Desai, L. Iftode, "Accessing Ubiquitous Services using Smart Phones", 4th IEEE International Conference on Pervasive Computing and Communications (PerCom), Kauai Island (HW), March 2005.

[Tum05] L. Tummolini, C. Castelfranchi, A. Ricci, M. Viroli, A. Omicini, "Exhibitionists and Voyeurs do it better",, Environments for MultiAgent Systems. LNAI 3374, Springer-Verlag, January 2005.

[ZamJW03] F. Zambonelli, N. Jennings, M. Wooldridge, "Developing Multiagent Systems: the Gaia Methodology", ACM Transactions on Software Engineering and Methodology, 12(3):317-370, 2003.

[Zam05] F. Zambonelli, M.P. Gleizes, M. Mamei, R. Tolksdorf, "Spray Computers: Explorations in Self-Organization", Journal of Pervasive and Mobile Computing 1(1):1-20, May 2005.