Deliverable 6.2

Part A: "Experimental QoS Evaluation in Autonomic Network Environments"

Status and Version:	Final	
Date of issue:	30.04.2008	
Distribution:	Public	
Author(s):	Name	Partner
	R. Lent, L. Hey, G. Sakellari	ICL
	I. Stavrakakis	NKUA
Checked by:	A. Manzalini	TI

Table of Contents

1 Introduction 3

1.1 Purpose and Scope 3

1.2 Document History 3

1.3 Document overview 4

1.4 Reference Material 4

2 Evaluation 8

2.1 Evaluation Areas and Metrics 9

2.2 Evaluation Parameters 11

2.2.1 Markov models for the creation of workloads 11

2.2.1.1 Arrival processes 12

2.2.1.2 Popularity distributions 13

2.2.2 Random graph models for representing the topology of Services, SEEs, and ACEs 15

2.3 Evaluation Methods 18

2.3.1 Analytic methods. 18

2.3.1.1 Markov Decision Theory: optimising the behaviour of a single intelligent agent that interacts with a random environment 19

2.3.1.2 Game Theory: optimising the behaviour of an intelligent agent that interacts with multiple intelligent agents 21

2.3.2 Computer Simulation 21

2.3.2.1 Event driven simulation 22

2.3.2.2 Component Simulators 23

2.3.2.3 Real Time and Hybrid Approaches 23

2.3.2.4 Testbed experimentation and open-network testing 23

2.3.2.5 Experimental methods in the literature of mobile ad hoc networks 24

3 Conclusions 26

1 Introduction

1.1 Purpose and Scope

A main goal of CASCADAS is to identify and to create a general-purpose abstraction that will facilitate the development of future autonomic and situation-aware services. This abstraction will be realised in the form of the autonomic communication element (ACE), a component-based model that will be used as a basic building block to construct services by allowing them to autonomously organise and adapt to the context in which they operate. The systems constructed from ACEs are expected to be of great complexity, creating dense environments of heterogeneous computing devices where communication and computation will be possible as a result of a high degree of inter-ACE collaboration, yet made easily manageable and usable by the autonomic properties offered by the system.

The purpose of D6.2A is to explore technical evaluation approaches to ACEs and their interactions to gain valuable feedback about the system's operation. This study complements the evaluation of socio-economic aspects of CASCADAS' outputs covered in D6.1B. A set of measures can describe the system in a quantitative way to forecast or directly determine its performance (e.g. measure quality-of-service metrics), and can give directions to other work packages for correcting or improving parts of the system. Likewise, specific requirements from other WPs will be evaluated, for example, by obtaining specific measures showing the effects of particular changes in the system.

The document covers the elementary tools that will be used in the project for investigating the behaviour and performance of autonomic systems. The tools are based on prototyping and test-bed experimentation, as the purpose of CASCADAS is to create an open-source autonomic toolkit. Analytic and simulation methods are also covered in the document as there are other useful tools that can guide the toolkit design. Most measures will be related to the quality of service (QoS) offered by the ACE system under study by evaluating the QoS level at which goals can be fulfilled.

1.2 Document History

Version	Date	Authors	Comment
0.1	10/06/2006	Ricardo Lent	Initial document
0.2	04/10/2006	Ricardo Lent	Updated after Berlin's meeting
0.3	09/10/06	Laurence Hey	Various updates
0.4	16/11/06	Ricardo Lent	Section 3 removed, Section 1 updated
0.5	20/11/06	Ricardo Lent	NKUA contribution added
0.6	21/12/06	Ricardo Lent	Update after WP leaders meeting
0.7	27/03/07	Ricardo Lent	Implementing comments of 1st ESR
Final	30/04/08	Antonio Manzalini, Ricardo Lent	Implementing comments of 2nd ESR

1.3 Document overview

The document discusses evaluation methods that are applicable to autonomic network systems, focusing principally on experimentation. The discussion analyses evaluation areas and relevant metrics (related to quality-of-service) that can be observed to gain insight into the operation of a system, as well as elaborating on various significant parameters that control experiments.

1.4 Reference Material

[1] D. M. Lucantoni, "New Results on the Single Server Queue with a Batch Markovian Arrival Process," Commun. Statist.- Stochastic Models, vol. 7, no. 1, pp. 1-46, 1991

[2] Mitzenmacher, Michael, "A Brief History of Generative Models for Power Law and Lognormal Distributions," Internet Mathematics, I (2003), 226-251.

[3] Neuts, M. F., Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach, The Johns Hopkins University Press, Baltimore, 1981.

[4] Eramilli, A. and J. Wang, J., "Monitoring Packet Traffic Levels", Proceedings IEEE Globecom '94, pp. 274-280, 1994.

[5] P. Erdos and A. Renyi, "On the evolution of Random Graphs," Publications of the Mathematical Institute of the Hungarian Academy of Sciences, Vol. 5, pp. 17-61, 1960.

[6] T. Blass, "The Man Who Shocked the World: The Life and Legacy of Stanley Milgram," Basic Books, March 2004.

[7] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, "Random Graphs with Abritrary Degree Distributions and Their Applications," Physical Review E, Vol. 64, 026118, July 2001.

[8] B. Bollobas, "Random Graphs," Academic Press, 1985.

[9] R. Hekmat, "Fundamental Properties of Wireless Mobile Ad Hoc Networks," Ph. D. Thesis, Technische Univewrsiteit Delft, 2005.

[10] R. Albert and A. L. Barabasi, "Statistical Mechanics of Complex Networks," Review of Modern Physics, Vol. 74, pp. 47-97, January 2002.

[11] D. J. Watts and S. H. Strogatz, 1998, Nature (London), 393, 440.

[12] A.-L Barabasi and R. Albert, 1999, Science 286, 509.

[13] A.Vogel, B.Kerherve, G.VonBochmann and J. Gecsei, "Distributed multimedia and QoS: A survey", IEEE Multimedia, vol. 2, no. 2, pp. 10–19, 1995.

[14] L. C. Wolf, C. Griwodz, and R. Steinmetz, "Multimedia communication", Proceedings IEEE, vol. 85, no. 12, pp. 1915–1933, 1997.

[15] E. Bertino, T. Catarci, A. K. Elmagarmid, and M. S. Hacid, "Quality of service specification in video databases", IEEE Multimedia, vol. 10, no. 4, pp. 71–81, 2003.

[16] J. P. Thomesse, "Fieldbuses and quality of service", in The 5th Portuguese Conference on Automatic Control, Annecy; France, 2002, pp. 10–14.

[17] Gilbert Held, Frame relay networking, Wiley, Chichester ; New York, 1999.

[18] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet architecture: An overview", Tech. Rep., IETF Informational RFC 1633, Jun 1994.

[19] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (rsvp)", Tech. Rep., Version 1 Functional Specification, IETF RFC 2205 (Proposed Standard), Updated by RFCs 2750, 3936, Sep 1997.

[20] J.Wroclawski, "The use of RSVP with IETF integrated services", Tech. Rep., Internet proposed standard RFC 2210, Sep 1997.

[21] L. Zhang, S. Deering, D. Estrin, and S. Shenker, "RSVP: A new resource reservation protocol", IEEE Network, vol. 7, no. 5, pp. 8–18, 1993.

[22] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service", Tech. Rep., Informational RFC 2475, Updated by RFC 3260, Dec 1998.

[23] Harry G. Perros, An introduction to ATM networks, Wiley, Chichester [England] ; New York, 2002.

[24] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture", Tech. Rep., RFC 3031 (Proposed Standard), Jan 2001.

[25] P. J. Welcher, "Introduction to MPLS", Aug 2000.

[26] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas, "LDP specification", Tech. Rep., RFC 3036 (Proposed Standard), Jan 2001.

[27] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta, "MPLS label stack encoding", Tech. Rep., RFC 3032 (Proposed Standard), Updated by RFC 3443, Jan 2001.

[28] C. Semeria, "Migration strategies for IP service growth: Cell-switched MPLS or IP-routed MPLS", Tech. Rep., White Paper, Juniper Networks, Inc., Mar 2002.

[29] B. Davie, J. Lawrence, K. McCloghrie, E. Rosen, G. Swallow, Y. Rekhter, and P. Doolan, "MPLS using LDP and ATM VC switching", Tech. Rep., RFC 3035 (Proposed Standard), Jan 2001.

[30] A. Conta, P. Doolan, and A. Malis, "Use of label switching on frame relay networks specification", Tech. Rep., RFC 3034 (Proposed Standard), Jan 2001.

[31] S. Jamin, S. J. Shenker, and P. B. Danzig, "Comparison of measurement based admission control algorithms for controlled-load service", in Proceedings of the Conference on Computer Communications (IEEE INFOCOM '97), Kobe, Japan, 1997, vol. 3, pp. 973–980, IEEE Computer Society Press.

[32] R. J. Gibbens, F. P. Kelly, and P. B. Key, "A decision-theoretic approach to call admission control in ATM networks", IEEE Journal on Selected Areas of Communications, vol. 13, no. 6, pp. 1101–1114, 1995.

[33] D. Tse and M. Grossglauser, "Measurement-based call admission control: Analysis and simulation", in Proceedings of the INFOCOM '97,Kobe, Japan, 1997, vol. 3, pp. 981–989, IEEE Computer Society Press.

[34] S. Jamin, P. Danzig, S. Shenker, and L. Zhang, "A measurement based admission control algorithm for integrated services packet networks", IEEE/ACM Transactions on Networking, vol. 5, no. 1, pp. 56–70, 1997.

[35] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks", IEEE Journal on Selected Areas in Communications, vol. 9, no. 7, pp. 968–981, 1991.

[36] R. Guerin and L. Gun, "A unified approach to bandwidth allocation and access control in fast packet-switched networks", in Proceeding of INFOCOM'92, Florence, Italy, 1992, vol. 1, pp. 1–12.

[37] E. Gelenbe, X. Mang, and R. Onvural, "Diffusion based statistical call admission control in atm", Performance Evaluation, vol. 27/28, no. Com, pp. 411–436, 1996.

[38] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint admission control: Architectural issues and performance", in Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, 2000, pp. 57–70, Acm.

[39] G. Bianchi, A. Capone, and C. Petrioli, "Throughput analysis of end-to-end measurement-based admission control in ip", in Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, 2000, pp. 1461–1470, IEEE.

[40] V. Elek, G. Karlsson, and R. Ronngren, "Admission control based on endto-end measurements", in Proceedings of IEEE INFOCOM 2000,TelAviv, Israel, 2000, vol. 2, pp. 623–630.

[41] R. J. Gibbens and F. Kelly, "Distributed connection acceptance control for a connectionless network", in Proceedings of the 16th International Teletraffic Congress (ITC 99), Edinburgh, UK, 1999, vol. 2, pp. 941–52.

[42] C. Cetinkaya and E. W. Knightly, "Egress admission control", in Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, 2000, vol. 1, pp. 1471–1480, IEEE.

[42] G. Malkin, "Routing information protocol version 2", Internet RFC 2453, November 1998.

[43] J. Moy, "Open shortest path first version 2", Internet RFC 2328, April 1998.

[44] Z. Wang and J. Crowcroft, "QoS routing for supporting resource reservation", IEEE Journal of Selected Areas in Communications, vol. 14, no. 7, pp. 1228–1234, 1996.

[45] R. Guering and A. Orda, "QoS-based routing in networks with inaccurate information", in Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), April 1997, pp. 75–84.

[46] B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput-competitive online routing", in Proceeding of the 34th Symposium Foundations of Computer Science, April 1993, pp. 32–40.

[47] H. F. Salama, D. S. Reeves, and Y. Viniotis, "A distributed algorithm for delay-constrained unicast routing", in Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), April 1997.

[48] G. Chen and N. Ansari, "Multiple additively constrained path selection", IEE Proceedings – Communications, vol. 149, no. 5–6, pp. 237–241, 2002.

[49] David B. Johnson David A. Maltz, Josh Broch, "Experiences designing and building a multi-hop wireless ad hoc network testbed", Tech. Rep., CMU, 1999.

[50] Henrik Lundgren, David Lundberg, Johan Nielsen, Erik Nordstrom, and Christian Tschudin, "Large-scale testbed for reproducible ad hoc protocol evaluations", in Proceedings of IEEE WCNC'02, 2002.

[51] Daniel Aguayo, John Bicket, Sanjit Biswas, Douglas S. J. De Couto, and Robert Morris, "MIT Roofnet implementation", Tech. Rep., MIT, 2003.

[52] Douglas R. Raichle James T. Kaba, "Testbed on a desktop: strategies and techniques to support multi-hop manet routing protocol development", in Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing, 2001.

[53] S. Sanghani, T. X. Brown, S. Bhandare, and S. Doshi, "EWANT: The emulated wireless ad hoc network testbed", in IEEE Wireless Communications and Networking Conference (WCNC), 2003.

[54] Maximilian Ott, Ivan Seskar, Robert Siracusa, and Manpreet Singh, "Orbit testbed software architecture: Supporting experiments as a service", in Proceedings of IEEE Tridentcom 2005, Feb 2005.

[55] Yongguang Zhang and Wei Li, "An integrated environment for testing mobile ad-hoc networks", in Proceedings of ACM MobiHoc'02, 2002.

[56] R. Lent. A Testbed Validation Tool for MANET Implementations. In Proceedings of the IEEE Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Sep 2005.

[57] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, "GloMoSim: A Scalable Network Simulation Environment," Technical Report 990027, UCLA Computer Science Department

[58] K. Fall and K. Varadhan, "The NS Manual (Formerly NS Notes and Documentation) ", http://www.isi.edu/nsnam/ns/doc/index.html, August 2002

[59] David Johnson, Tim Stack, Russ Fish, Daniel Montrallo Flickinger, Leigh Stoller, Robert Ricci, Jay Lepreau, Mobile Emulab: A Robotic Wireless and Sensor Network Testbed, In *Proceedings of the 25th Conference on Computer Communications (*IEEE INFOCOM 2006*), April 2006*

[60] Pradipta De, Ashish Raniwala, Srikant Sharma, and Tzi-cker Chiueh, MiNT: A Miniaturized Network Testbed for Mobile Wireless Research. *In Proceedings of IEEE Infocom, 2005*

[61] Werner-Allen, G.; Swieskowski, P.; Welsh, M., "MoteLab: a wireless sensor network testbed," *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on* , vol., no.pp. 483- 488, 15 April 2005

[62] Brown, A. B., Hellerstein, J., Hogstrom, M., Lau, T., Shum, P., and Yost, M. P. 2004. Benchmarking Autonomic Capabilities: Promises and Pitfalls. In *Proceedings of the First international Conference on Autonomic Computing (Icac'04) - Volume 00* (May 17 - 18, 2004). ICAC. IEEE Computer Society, Washington, DC, 266-267.

[63] Salehie, M. and Tahvildari, L. 2005. Autonomic computing: emerging trends and open problems. In *Proceedings of the 2005 Workshop on Design and Evolution of Autonomic Application Software* (St. Louis, Missouri, May 21 - 21, 2005). DEAS '05. ACM Press, New York, NY, 1-7.

[64] Puterman, M. L. 1994, Markov Decision Processes: Discrete Stochastic Dynamic Programming. 1st. John Wiley & Sons, Inc.

[65] Bertsekas, D. 1995, Dynamic Programming and Optimal Control, Vols. I and II, Athena Scientific.

[66] Tijms, H. 1986, Stochastic Modelling and Analysis, A Computational Approach, Wiley, New York.

2 Evaluation

Autonomic network environments create evaluation challenges that are normally not present in other types of systems. A set of proper evaluation methods would allow CASCADAS researchers to better understand the dynamics of a system created by ACEs, to detect possible problems in the design and to develop performance improvements. At the same time, a proper set of evaluation methods would permit performance comparison studies with similar approaches investigated in other contexts.

To fulfil this goal, in particular in the context of experimentation, a proper testing environment will need to be designed to allow assessing the autonomic capabilities of ACEs: self-optimisation, self-configuration, self-healing and self-protection, amongst others. Unlike traditional system testing, a perturbation usually needs to be injected into the system under observation to evaluate its adaptation capabilities [63]. Figure 1 illustrates the basic elements involved in an experimental evaluation environment for ACEs, which in most cases can be partially applicable to other evaluation methods:

- System of ACEs. Consists of the ACEs under study and other supporting elements, such as an underlying communication network (e.g. Internet) and knowledge networks. For the experimental evaluation, the system of ACEs will consist of a distributed testbed environment and the software implementing ACE functionality deployed in the nodes (supported by the autonomic toolkit).
- Test manager. Consists of supporting hardware, but principally software to control the execution and monitoring of experiments. The test manager normally uses listed descriptions of the experiments, for example, a temporal description of the workload to be applied to ACEs.
- System input. There are two types of system inputs. The first may come from the test manager, which might generate a synthetic workload (random or pre-defined) to excite the system under study. The other type of input might come from real users testing the experimental system. In any case, it is of relevant importance to allow reproducibility of experiments, so meaningful statistical measures of the system response can be obtained.
- Test output. Consists of evaluating the measures of interest. A test output will typically be in the form of quantitative reports of system performance and autonomic adaptation capabilities. However, outputs could also be in the form of visual representations of the system behaviour, principally for demonstration purposes.
- Perturbations. In autonomic systems, a perturbation is normally required to put to test their autonomic adaptation features. The perturbation will typically come from the test manager as part of the pre-defined set of instructions describing the steps of the experiment or they might come from user input. To make experiments meaningful, perturbations will need to be correlated to response metrics and test outputs. Another form of perturbation might come from the dynamics of the system itself, because of the system use. For example, inherent failures might occur in the network without any external input and workload might concentrate and saturate parts of the network because of the internal behaviour of the system.

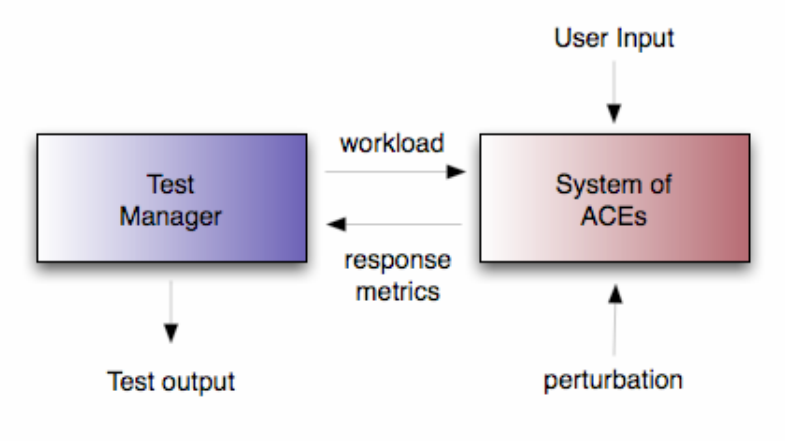

Figure 1: Evaluation of a system of ACEs

2.1 Evaluation Areas and Metrics

The study of autonomic networks requires gaining insight into the inherent behaviour of the system, which is critical both to ensure correctness and to verify autonomicity. Particularly,

1. Evaluation of adaptation and the ability to detect and organise knowledge. This could be expressed in terms of aggregation dynamics and their correlation with the goal achievable (GA) and goal needed (GN) values of ACEs. Similarly, such an evaluation can consider an investigation of the system stabilisation; that is, the time the system takes to reach a stable state (e.g. when adapting to a new context), and the reaction time.
2. A study of the fairness of the system, which should demonstrate the degree at which, under the same conditions, any user could potentially receive a similar level of service.
3. An evaluation of the system's sensitivity and existence of erratic behaviours that may arise from structural errors in the system, erroneous implementations or incorrect decisions from stale information. For example, networks of ACEs may display oscillations in aggregation decisions; that is a quick and ineffective change, back and forth, in the selection of an aggregation peer.
4. An estimation of the scalability of the system in terms of the number of users, ACEs, physical nodes and services it can support.

Another area of interest for autonomic communication systems is the evaluation of system performance. Performance can be measured in terms of:

1. The level at which goals are achieved by the system. Goals can be expressed in terms of quality-or-service (QoS) metrics. A large number of measurable QoS metrics exist, and the following are common metrics found in the literature of communication networks:

    - **Latency or delay:** The sum of transmission times, queuing delays, propagation and processing times along all hops of a given path. Delay is an additive metric, thus, the delay of a path is the sum of the delays on each of its hops and links. In most cases propagation, transmission and processing delays are negligible compared to queuing delay.

- **Packet Loss:** Losses usually occur due to corruption or congestion, the latter being the most common source in modern networks. Other causes are physical problems (e.g. cable failure) and security attacks (e.g. denial of service attacks). Packet loss is a multiplicative metric, so that the overall loss of a path is the product of the loss of each individual hop and link.
- **Jitter** is the variation of delay among packets of the same connection. Depending on the network protocol used, sudden changes in jitter may cause packet losses. Jitter is particularly critical in voice applications and it is minimised by buffering at the destination node. Most voice-over-IP (VoIP) endpoint devices have buffers to compensate and smooth out the network jitter. However, they increment the end-to-end delay, so that they are usually effective for small jitters (less than 100 ms). Jitter is an additive metric.
- **Available Bandwidth** is the minimum bandwidth that a communication requires or the spare bandwidth of the network that can be used by existing or new users. The bandwidth of a path as a whole is determined by the link with the minimum available bandwidth, which makes it a concave metric.
- **Hop count** is the number of nodes a packet visits before reaching its destination. It is a common metric in routing policies that find the shortest path. Hop count is an additive metric.
- **Reliability** is the ability to continue providing service for a client even during network failures or attacks. There are several ways to measure reliability. It can be associated to the physical resilience of the links (e.g. type of cable, transport medium, etc.) It can also be associated with the packet loss probability due to long-term congestion of the network. Reliability is a multiplicative metric.
- **Security** has been of great concern, in particular for military communications. However, nowadays it is becoming more prevalent in the civilian world as electronic communications continue to be integrated to our daily activities (e.g. online transactions). Security is a concave metric.
- The **cost** of a communication is determined by the total cost for utilising the links and nodes along the path. Cost is an additive metric.

The evaluation of QoS metrics should consider systems at rest and under load; that is, systems with no traffic other than that generated by the applications under study and systems that carry external traffic from other applications.

2. Autonomic benefits. A major objective in the creation of ACEs is to offer an alternative means for existing mechanisms to create services with a significant reduction in development and maintenance costs. Certain elements of such reductions are measurable and important to understand, to effectively justify the benefits introduced by ACEs.
3. Complexity introduced by the system of ACEs. The shortcoming of introducing new mechanisms is that inevitably new complexities would be brought in, resulting in extra overhead. Overhead is in general a measurable factor that can be evaluated by the time required to execute processes required to maintain the system or by storage requirements. Overhead can also be expressed as the number of communication packets required ensuring a specific behaviour of the system.
4. Evaluation of application specific characteristics. An evaluation of pertinent measures of applications relying on a system of ACEs is also of applicable importance, as

applications constitute the main purpose of the system. A description of particular factors for use cases of interest is described in D6.2B.

2.2 Evaluation Parameters

There are a considerable number of parameters whose selection will affect the evaluation of autonomic communications. Some of the parameters that need special attention when evaluating the system are:

1. **Users:** ACEs are expected to supply the basic building blocks to construct a large number of future applications, which will serve a huge number of users. Consequently, evaluations should consider large sets of users in experiments. Such consideration should include stochastic models for user rates entering and leaving the system and a model for the system residence time. In addition, seasonal effects could also be considered, which would affect the behaviour of the users (e.g. day/night usage), and the response of the system to rare events, such as flash crowds. In an experimentation context, the main workload will come from synthetic sources (e.g. Markov models).
2. **Networks:** ACEs are likely to construct overlay networks operating over existing networks. The actual underlying network topology plays an important role in the quality of ACE communications. Therefore, they should be carefully considered when evaluating the performance of the system. Another important aspect related to the underlying network is node mobility, which is expected to play an even more prominent role in future networks and that will dynamically change the network over time. Network topologies can be generated for evaluation purposes following a certain model (e.g. random and small world topologies).
3. **Failures and attacks:** To add realism to evaluation studies, failures and security attacks should be considered in evaluation models. Failures can be in the form of node malfunctions and network breakdowns (e.g. creating network partitions). In terms of security attacks, denial-of-service attacks should concentrate particular attention as they might cause potential problems in a highly collaborative network of ACEs.

We provide a brief overview of results that can be used for the generation of input parameters for experimental evaluations of the CASCADAS architecture. We start by presenting models for the creation of synthetic workloads (to model requests between CASCADAS entities) and continue with models for the creation of synthetic topologies (to model relationships between CASCADAS entities).

2.2.1 Markov models for the creation of workloads

In the following subsections we present arrival processes and popularity distributions that can be used for generating workload for experimental evaluations. Arrival processes will be required to model several dynamic events in the CASCADAS architectures such as the arrival and departure of new ACEs, messages between different ACEs, request arrival times from end-users, etc. Popularity distributions are useful for modelling relative frequency between similar entities with different identities, e.g., requests for different types of services offered by a given ACE, relative popularity of different ACEs, etc.

2.2.1.1 Arrival processes

The Poisson process is probably the most popular process for modelling arrivals. A stochastic process $\{A(t) \mid t \geq 0\}$ representing the total number of arrivals that have occurred from time 0 to t ($A(0) = 0$) is said to be Poisson with rate λ if the number of arrivals that occur in disjoint time intervals are independent and the number of arrivals in any interval of length τ is Poisson distributed with parameter $\lambda\tau$ i.e. for $t, \tau > 0$

$$P\{A(t+\tau) - A(t) = n\} = e^{-\lambda\tau}\frac{(\lambda\tau)}{n!} n = 0,1,\ldots$$

For a Poisson process, the inter-arrival times are independent and exponentially distributed with parameter λ. The Poisson process may be thought of either as a special case of a renewal arrival process, where the inter-arrival times are assumed to be independent, identically distributed (iid) random variables, or as a special case of Batch Markovian Arrival Process (BMAP), as described later in this section.

Many inter-arrival distributions have a shape not at all similar to the exponential, e.g. they have a maximum for t greater than zero. Such distributions can be represented as sum of a number of exponential stages (phases) with the same intensity; the resulting distribution is a gamma with integer form parameter, denoted as Erlang-k.

More specifically, a random variable *X* has an Erlang-k ($k = 1, 2, \ldots$) distribution with mean k/μ if *X* is the sum of k independent random variables $X_1,\ldots,X_k$ having a common exponential distribution with mean 1/μ. The common notation is $E_k(\mu)$ or briefly E_k. The density of an $E_k(\mu)$ distribution is given by

$$f(t) = \mu\frac{(\mu t)^{k-1}}{(k-1)!} e^{-\mu t}, \quad t > 0$$

The distribution function equals

$$F(t) = 1 - \sum_{j=0}^{k-1}\frac{(\mu t)^j}{j!} e^{-\mu t}, \quad t \geq 0$$

The Erlang-k arrival process is less bursty than the Poisson process (for very large k, inter-arrival times become almost deterministic). At this point it should be noted that several methods have been proposed for the quantitative description of burstiness of Internet traffic [4].

The Erlang distribution is a special case of a class of distributions, referred to as phase type distributions [3]. A non-negative random variable T (or its distribution function) is said to be of phase-type (PH) if T is the time until absorption in a finite-state continuous-time Markov chain.

Formally, a PH distribution with parameter $(\vec{\tau}, \mathbf{T})$, PH($\vec{\tau}, \mathbf{T}$), is the distribution of the time until absorption into state 0 in a Markov chain on the states $\{0,1,\ldots,n\}$ with initial probability vector $(\tau_0, \vec{\tau})$ and infinitesimal generator

$$\begin{pmatrix} 0 & \vec{0} \\ \vec{t} & \mathbf{T} \end{pmatrix},$$

where $\vec{t} = -\mathbf{T}\vec{1}$ and $\tau_0 = 1 - \vec{\tau}\vec{1}$, where $\vec{1}$ is a column vector of 1's.

Phase-type processes are a subclass of Batch Markovian Arrival Processes (BMAPs) that have been introduced in [1]. Consider a two dimensional Markov process $\{N(t), J(t)\}$ on the state space $\{(i, j) : i \geq 0, 1 \leq j \leq m\}$ with an infinitesimal generator Q having the structure,

$$Q = \begin{bmatrix} D_0 & D_1 & D_2 & D_3 & \dots \\ & D_0 & D_1 & D_2 & \dots \\ & & D_0 & D_1 & \dots \\ & & & \dots & \dots \\ & & & & \dots \end{bmatrix}$$

Where $D_k, k \geq 0$ are $m \times m$ matrices, D_0 has negative diagonal elements and non-negative off-diagonal elements $D_k, k \geq 1$, are non-negative and D, defined by $D = \sum_{k=0}^{\infty} D_k$ is an irreducible infinitesimal generator. We also assume that $D \neq D_0$. If $N(t)$ represents a counting process and $J(t)$ an auxiliary state or phase variable then the above Markov process defines a batch arrival process where transitions from a state (i, j) to state $(i+k, l)$, $k \geq 1$, $1 \leq j$, $l \leq m$, correspond to batch arrivals of size k, and thus batch size can depend on i and j.

Intuitively, D_0 can be thought of as governing transitions in the phase process which do not generate arrivals and D_k as the rate of arrivals of size k (with the appropriate phase change). More specifically, assume that the underlying Markov process with generator D is in some state $i, 1 \leq i \leq m$. The sojourn time in that state is exponentially distributed with parameter λ_i. At the end of that sojourn time, there occurs a transition to another (or possibly the same) state and that transition may or may not correspond to an arrival epoch. With probability $p_i(0,k), 1 \leq k \leq m, k \neq i$, there will be a transition to state k without an arrival. With probability $p_i(j,k),\ j \geq 1,\ 1 \leq k \leq m, k \neq i$, there will be a transition to state k with a batch arrival of size j. We therefore have, for $1 \leq i \leq m$, $\sum_{k=1, k \neq i}^{m} p_i(0,k) + \sum_{j=1}^{\infty} \sum_{k=1}^{m} p_i(j,k) = 1$, and with this notation it is clear that $(D_0)_{ii} = -\lambda_i, 1 \leq i \leq m$, $(D_0)_{ik} = \lambda_i p_i(0,k), 1 \leq i, k \leq m, k \neq i$, and $(D_j)_{ik} = \lambda_i p_i(j,k),\ j \geq 1, 1 \leq i, k \leq m$. Thus, D_0 governs transitions that correspond to no arrivals, and D_j governs transitions that correspond to arrivals of batches of size j.

2.2.1.2 Popularity distributions

In this section we review power-law and log-normal distributions, which are two forms of distributions widely used for modelling the frequency or popularity of many man-made and naturally-occurring phenomena, including city sizes, incomes, word frequencies, earthquake magnitudes. A power-law implies that small occurrences are extremely common, whereas large instances are extremely rare.

In [2] some of the basic models that lead to power law and lognormal distributions are reviewed and specifically how small variations in the underlying model can change the result from one to the other is covered. For example, the World Wide Web can be thought of as a graph, with pages corresponding to vertices and hyperlinks corresponding to directed edges.

Empirical work has shown that in-degrees and out-degrees of vertices in this graph obey power law distributions. In the sequel, the definitions of these distributions are provided.

A non-negative random variable X is said to have a power law distribution if

$$\Pr[X \geq x] \propto c\, x^{-\alpha}$$

for constants $c > 0$ and $\alpha > 0$. Roughly speaking, in a power law distribution asymptotically the tails fall according to the power α. Such a distribution leads to much heavier tails than other common models, such as exponential distributions. One specific commonly used power law distribution is the Pareto distribution, which satisfies

$$Pr[X \geq x] = \left(\frac{x}{k}\right)^{-a}$$

for some $\alpha > 0$ and $k > 0$. The Pareto distribution requires $X \geq k$. The density function for the Pareto distribution is $f(x) = \alpha k^{\alpha} x^{-\alpha-1}$. For a power law distribution, usually α falls in the range $0 < \alpha \leq 2$, in which case X has infinite variance. If $\alpha \leq 1$, then X also has infinite mean.

A power law distribution may be obtained by the mechanism that is often called preferential attachment; new objects tend to attach to popular objects. For example, in the case of the Web graph, new links tend to go to pages that already have links. Let us start with a single page, with a link to itself. At each time step, a new page appears, with out-degree 1. With probability α < 1, the link for the new page points to a page chosen uniformly at random. With probability 1−α, the new page points to page chosen proportionally to the in-degree of the page.

A random variable X has a lognormal distribution if the random variable $Y = \ln X$ has a normal (i.e., Gaussian) distribution. The normal distribution Y is given by the density function

$$f(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(y-\mu)^2/2\sigma^2}$$

where μ is the mean, σ is the standard deviation (σ^2 is the variance), and the range is $-\infty < y < \infty$; therefore, the density function for a lognormal distribution satisfies

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\ln x-\mu)^2/2\sigma^2}.$$

The corresponding complementary cumulative distribution function for a lognormal distribution is given by

$$Pr[X \geq x] = \int_{z=x}^{\infty} \frac{1}{\sqrt{2\pi}\sigma z} e^{-(\ln z-\mu)^2/2\sigma^2} dz$$

A lognormal distribution has finite mean and variance, in contrast to the power law distribution under natural parameters. Despite its finite moments, the lognormal distribution is extremely similar in shape to power law distributions, in the following sense: If X has a lognormal distribution, then in a log-log plot of the complementary cumulative distribution function or the density function, the behaviour will appear to be nearly a straight line for a large portion of the body of the distribution.

Lognormal distributions are generated by multiplicative processes [2]. For example, in biology such processes are used to describe the growth of an organism; the random growth of an organism is expressed as a percentage of its current weight, and is independent of its current actual size.

2.2.2 Random graph models for representing the topology of Services, SEEs, and ACEs

This section provides a brief overview of random graph models that can be used for the representation of relationships between different CASCADAS entities, such as ACEs, SEEs, and users. The study of such relationship has been taken place in the context of "network modelling". Network modelling is in the core of any network study since it allows for the understanding of many properties that real networks demonstrate. Irrespectively of the nature of the network (e.g., ad hoc network, social network, the biochemical network, the web pages network etc.), it has been shown that *common properties* are present and a common framework to analise them do exist.

Various models have been proposed in the past that aim to explain the *creation* of networks and their special properties. The first, proposed by Erdos and Renyi, [5], known as the *random graph* or the *ER model*, was until recently the main model considered. The last decade, a lot of research work was initiated in this area when it was observed that networks do not actually follow the random rules of the ER model but rather the (analised later) power-law rule.

The following section presents a brief description of the most well known network models and some of their properties.

Overview:

A network can be represented by a graph *G* consisting of a set of edges *E* and a set of nodes (or vertices) *V*. A edge is a link among two nodes. For most of the cases networks are assumed to consist of *bidirectional* links in the sense that if communication from node u to node v is possible, then communication from node v to node u is also possible.

The Erdos-Renyi or *ER model*, was first introduced and analised by Erdos and Renyi in the early sixties [5]. It is a simple model and has been the basis for the development of the theory of *random graphs*; this is the reason why some times it is also referred to as the *random graph model*. However, recent observations regarding the structure of the networks (not only in the area of telecommunication networks but in a much more diverse spectrum, including, social and biological networks), have shown that the ER model fails to capture a number of interesting properties.

For example, it has been shown that the *diameter* of the network (i.e. the largest shortest path for a pair of nodes in the network), does not always increase as the *network size* (i.e. the number of node in the network) increases. Another observation (closely related to the previous one) is that nodes distances are significantly smaller than predicted by the random graph model for the same network size. Milgram's experiment in 1967 [6], gave a clear demonstration of this property. It presented evidence that it is possible for any person to reach any other person (even someone and he is not aware that exists) within a limited number of common friends (actually six people). The aforementioned property is known today as the *small-world* property. In a network with a small-world property, there is a high probability that there exists a relatively short path between any two nodes, regardless of the network size.

- The ER Model

The ER model, [5], is one of the most studied starting with Erdos and Renyi that introduced it in the early sixties. There are two possible and equivalent ways to construct this model. Suppose that the size of the network is N. At the first step assume that there is only one node in the network and that another node enters the network. Both nodes may have a link among them according to a certain probability p (or may not have a link among them according to

probability $1-p$). Any new node that enters the network (until the number of nodes in the network is equal to N) decides whether there will be a link with any of the nodes already present in the network with probability p. Eventually, when the network is created there will be approximately $pN\frac{N-1}{2}$ edges. If the number of edges is denoted by L, then $E[L]=pN\frac{N-1}{2}$.

The second way to create a network according to the ER model is to start selecting randomly (and independently) the L edges out of the $N\frac{N-1}{2}$ possible edges. It is easy to see that both ways of creating a network according to the ER model are equivalent.

Let d_u denote the *degree* of node u (i.e., the number of neighbour nodes). Under the ER model, d_u has a binomial distribution [8],

$$Pr[d_u=k]=\binom{N-1}{k}p^k(1-)^{N-1-k}\approx\frac{z^k e^{-z}}{k!}, \quad (1)$$

where $z=E[d_u]=p(N-1)$. The second term of the previous equation is a *Poisson distribution*.

It is easy to calculate (in approximation) the average shortest path length among each pair of nodes. A simple way is by considering the fact that any node has on average $p(N-1)$ neighbour nodes. Therefore, after h hops $(p(N-1))^h$ nodes are *reachable*. All nodes are reachable when $(p(N-1))^h\approx N$. Finally, [9,10],

$$E[h]=\frac{\log(N)}{\log(p(N-1))}. \quad (2)$$

Although this is a rough estimation, it is enough to see that the distance among two nodes increases with respect to the logarithm of the network size.

The emergence of the *giant component* is an interesting property of the ER model. In particular, there exists a certain threshold value for the probability p for which a *cluster* is suddenly created throughout the network. This property (also known as *phase transition*) allows for a number of nice conclusions. For example, for values of p smaller than the threshold value, the network has a high probability of being disconnected, whereas for values of p larger than the threshold value, the network has a high probability of being connected. Another example is that the network diameter increases as p increases until the threshold value, after which it starts decreasing again.

- Scale-free Models

Even though there was a feeling that networks are far from random, it was not until the late nineties that new observations came up in a coherent way. Truly, the complex form of networks does not allow for individuals to work tightly on a specific research direction due to the numerous data that need to be collected, processed and analised. However, the widespread availability of computer power at the end of the nineties allowed for processing of the numerous data and finally for a new key observation: the degree distribution of the real networks does not follow the Poisson distribution (indicative of ER graphs) but rather it follows *power-law* with *heavy tail* [7]. In particular, based on experimental studies, it was suggested [7], that a realistic model for real-world networks has the form of,

$$Pr[d = k] \approx k^{-\gamma}, \quad (3)$$

where γ is a constant independent of the network size. It is experimentally found that γ is different for different types of networks.

The scale-free property influences certain properties of the network. For example, in such a network there exist a small number of nodes (called *hubs*) that are connected with a large number of nodes. As a result, in the event of a node failure such a network shows a remarkable resilience. Furthermore, the distances among the nodes are usually short (small-world property) since the paths go through the hubs. It can be argued that the small-world property is "stronger" in scale-free network than in ER networks.

A simplistic explanation of the power-law distribution with heavy tail is that a small (but non-zero) number of nodes exist that have a large number of neighbours (hubs) and that a large number of nodes have a few (but non-zero) neighbour nodes. There are two models that are well known in the area of scale-free networks and are briefly presented in the following paragraphs.

- The Watts-Strogatz Model

Watts and Strogatz proposed in 1998, [11], a new model, influenced by observations regarding human networks. The most interesting observation was that humans tend to meet people that are known by other people they know. For example, in a social event if someone meets new people that would be most likely because someone has introduced them to him and not because he started randomly introducing himself to others – reality is not as random as the ER model implies.

The model proposed by Watts and Strogatz attempts to incorporate the aforementioned characteristic by boosting up the probability of creating a link between two nodes that share common neighbours. For example, if node u is already a neighbour of node v, then the new node x that just become a neighbour of node u will become neighbour of v with increased probability.

This model consists of two main steps:

- *Start*: Assume a ring network with N nodes such that each node is connected to the K closest nodes ($K/2$ for each side). Assume that $N << K << \ln N << 1$.
- *Random Reordering*: Any edge of the ring is reordered with probability p such that there will be no self-edges or double edges.

It is obvious that for $p = 0$, the second step is not executed and the network remains as the initial ring network. In such a case, the situation presented refers to someone that knows people that are only a few hops away from him. However, this deterministic view is not compliant with real life where people definitely know (at least a few) people that are "many hops away". This is satisfied for values of $p > 0$. Note that for $p = 1$, then the aforementioned model is actually the ER model.

- The Albert-Barabasi Model

The basic idea behind the Albert-Barabasi Model [12], is the exploitation of two observations. First, if a node has many links to other nodes in the network, then it is likely to "attract" more links in the future. In order to explain somehow this observation, think about the reason why a node has initially many links. The most reasonable explanation is because this particular node has some "nice properties" that make the other nodes in the network eager to connect with it.

The second observation made by Albert and Barabasi is that nodes that have entered the network in an earlier time instance than other nodes, do have an advantage regarding the number of links. For example, if two nodes have the same "nice properties" but have entered the network at different time instances, then the first to enter has already attracted a number of links before the second starts attracting links. However, if the "nice properties" of the second are "nicer" than those of the first, it is expected that after some time the second node will have attracted more links that the first one.

This model consists of two main steps:

- *Start*: Assume a small number (m_0) of nodes. Every time a new node is added with m edges ($m \le m_0$) that connect the node to m other nodes of the network.
- *Preferential Attachment*: When the choice for a new connection takes place, the probability Π_u that the new node is connected with node u depends on the number of neighbour nodes of u, d_u,

$$\Pi_u = \frac{d_u}{\sum_{\forall v \in V} d_v}. \quad (4)$$

After t time instances, this procedure results in a network with $N = t + m_0$ nodes and mt edges [10]. Numerical results have shown that for this model $\gamma = 3$.

2.3 Evaluation Methods

The evaluation of autonomic communication systems can be pursued by various methods, experimentation being of particular importance given the pragmatic orientation of CASCADAS. In general, most evaluation studies of ACEs will ponder a mix of the following methods.

2.3.1 Analytic methods.

Analytic methods can provide important behaviour and performance measures from mathematical models of ACEs, specific components utilised by them, or ACE interactions for the creation of services. Analytic methods are traditionally classified as close-form and numerical methods. The first form is mainly applicable to simple models where an explicit

expression can be found to describe the measure of interest in terms of the model construction and parameters. Numerical methods are of wider applicability and consist of a combination of analytic and numerical techniques, often employing numerical analysis for the calculation of the measures. Analytic methods are particularly useful during the design phase of the ACEs and the definition of their interactions, when there are some uncertainties in the final form of the components to be developed. At this phase, parametric analysis can be used to evaluate the model's various factors or to determine parametric sensitivity. Then, considering application scenarios to drive research directions, analytic methods can provide the mechanism to deal with further uncertainties, for example, the workload generated by a number of users. Random variables naturally emerge when uncertainties are present, which can be studied in a stochastic manner (for example using queuing models).
In this section we briefly introduce Markov Decision Theory and Game Theory. These two theoretical tools can be used for optimising the behaviour of ACEs with respect to the environment and/or other ACEs. Markov decision theory is a means of analising which of a series of options should be taken when it is uncertain exactly what the result of taking the option will be but it is known that the future evolution of the system is independent from all previous states other than the current one. Markov decision theory concentrates on identifying the "best" decision option, where the notion of "best" is allowed to have a number of different meanings, of which the most common one is that which maximises the expected benefit to the decision maker.
Game theory is a close relative of decision theory, which studies interactions between self-interested entities. In particular, it studies the problems of how interaction strategies can be designed that will maximise the welfare of an entity in an encounter, and how protocols or mechanisms can be designed that have certain desirable properties. In the same way that decision theory can be claimed to provide a means of making rational decisions under uncertainty, so game theory can be claimed to provide a rational means of analising interactions. Notice that decision theory can be considered to be the study of games against nature, where nature is an opponent that does not seek to gain the best payout, but rather acts randomly.

2.3.1.1 Markov Decision Theory: optimising the behaviour of a single intelligent agent that interacts with a random environment

Markov decision processes (MDP) are simple yet powerful models for sequential decision problems. In these models, it is assumed that there is a state space; at each time the system occupies a certain state, and the decision maker, or controller, has a set of feasible actions for that state that can be applied. At the next time, the state changes according to some probability distribution which depends only on the current state and action, and does not depend on the past. The combination of current state and action also defines an immediate cost to be paid by the decision maker. The target of the decision maker is to prescribe a policy, i.e., a rule for deciding an action based on the current state, so as to minimise some function of the accumulated individual costs.

MDPs are also called controlled Markov chains in the literature, and have a wide range of application areas. In this brief introduction we will discuss only MDPs with a finite number of states and actions, and in a discrete-time context. We will first present the formalism for defining an MDP, discuss the different categories of MDPs with respect to the choice of actions, move on to present the most commonly used notions of cost, and finalise the introduction with some pointers to the computational methods used for solving MDPs.

The definition of an MDP includes the following:

- a state space *S*; we write $I_t = s$ when the process is at state *s* at time $t \geq 0$;
- for every state $s \in S$ a set of feasible actions *A(s)*;
- a set of state transition probabilities, $\{p(s'|s,a), \forall s, s' \in S, \forall a \in A(s)\}$, where *p(s'|s,a)* denotes the probability of observing a transition to state *s'* after having performed action *a* at state *s*, and
- a per-stage (a stage amounting to a single decision) cost function *c(s,a)* capturing the individual cost paid for performing action *a* at state *s*.

A policy *R* prescribes a way of choosing an action for each possible state. Formally a policy can be specified as a collection of probability distributions, $\{q(a|s), \forall s \in S\}$ where *q(a,s)* denotes the probability of choosing action *a* in state *s*. A policy is called deterministic when it always prescribes the same action at the same state, i.e., when *q(a,s)* take only the values 0 and 1; in this case we will denote *a(s)* the action prescribed by policy *R*. A policy that is not deterministic is called randomised. The above definition of policy actually amounts to a stationary policy, i.e., one in which the mapping from states to actions is independent of the current time *t* (when time is discrete it is synonymous to the current observation/control instance). Non-stationary policies that depend on time can also be defined by means of time-dependent transition probabilities $q_t(a|s)$, $t \geq 0$.

Several minimisation criteria can be defined based on different functions of the per-stage individual costs. The most commonly used cost functions for a policy *R* are the following.

- Finite horizon undiscounted cost: $C_{s_0} = E\left\{\sum_{t=1}^{W} c(I_t, a(I_t)) \mid I_0 = s_0\right\}$, i.e., the expected accumulated cost when starting from state s_0 at time *t=0*, and letting the process evolve for *W* observation instances according to the policy *R*.
- Infinite horizon discounted cost: $C_{s_0} = E\left\{\sum_{t=1}^{\infty} \gamma^t \cdot c(I_t, a(I_t)) \mid I_0 = s_0\right\}$, which is similar to the previous case with the difference that $W \to \infty$ and individual costs are discounted by γ^t, where $\gamma < 1$ is a discount factor (necessary for avoiding divergence of C_{s_0}).
- Expected average cost per observation instance: $C_{s_0} = E\left\{\lim_{W\to\infty} \frac{1}{W} \sum_{t=1}^{W} c(I_t, a(I_t)) \mid I_0 = s_0\right\}$. For well behaved MDPs, this kind of cost is independent of the initial state s_0 and can be written in a simpler manner as $\sum_{s\in S} \pi(s) \cdot c(s, a(s))$, where π denotes the limiting distribution of the MDP I_t.

Having modelled a problem as a Markov decision problem one needs a method for solving it, i.e., for searching the policy space for a policy that minimises the considered cost function. Since exhaustive enumeration requires checking $|A(s_1)| \cdot |A(s_2)| \cdot \ldots \cdot |A(s_{|S|})|$ policies, which for homogeneous action sets $A(s) = A, \forall s \in S$, is equal $|A|^{|S|}$, hence exponential to the size of the state space, one needs better strategies for obtaining an optimal policy *R**. Some of the most usual techniques for this purpose are the following:

- Casting the Markov decision problem as a Lineal Program (LP): There exists a straightforward mapping from MDP to LP. The resulting LP is of a special type (it is totally uni-modular) and therefore can be solved in polynomial time.
- Using specially developed heuristic methods such as the so called Policy- and Value-Iteration algorithms. These are more powerful techniques based on dynamic-programming and are more appropriate than LP for MDPs with large state and/or action sets (they require keeping less total information in memory at any given point).

For information on these and other solution methods the interested reader is advised to consult the following sources [64-66].

2.3.1.2 Game Theory: optimising the behaviour of an intelligent agent that interacts with multiple intelligent agents

Game theory adds to the model of Markov decision theory the element of multiple interacting intelligent agents, each one of which seeks to minimise its own individual cost. The difference may seem subtle but in fact it is a very important one, and its consequences are far reaching. The interaction between multiple (even two) intelligent agents can create an infinite loop of deviations from one strategy to another one which seems more appropriate as a response to the strategy "played" by the other agent. Such loops do not exist in decision theory, as in this case the environment has no mean (or the will) to "respond intelligently" to the agent's behaviour. Game theory provides the notions and the tools for analising such conflicts between intelligent agents. For an exposition of game theory the interested reader can consult deliverable D4.1 "Requirements, Design, State of the Art for security architecture and economics of security models: game theory and mechanism design reputation and trust management and access control through trust negotiation" and the references therein.

2.3.2 Computer Simulation

During the initial design stages of a scientific or engineering task, the knowledge or resources for direct system experimentation may not be present. Computer simulation offers the opportunity to implement and test mathematical models that approximate the desired end system.

Similarly to analytic methods, computer simulation can be employed to study ACEs, specific components making or used by ACEs, and networks of ACEs. However, computer-based models are useful to study a wider class of systems than analytic methods, which can be limited by mathematical tractability (for example if a closed form solution is not possible). With simulation, the analytic requirements of a study can be reduced by translating analytic complexity into a computer model. A program then represents the model and mimics the behaviour of the system, while allowing the collection of data. Various levels of elaboration can be achieved to represent the system under study, with greater or lesser degree of accuracy.

The trade-off between direct experimentation and computer simulation is between accuracy and reproducibility. When dealing with complex real world systems the simulation environment provides complete control over all external input parameters. However, capturing the nuances of the real world in simulation can be difficult, and very elaborate simulations, presumably the most accurate, may require a large number of calculations, which could need a long time to run.

As compared with direct system experimentation, simulations can proceed in compressed time as opposed to real time. In some cases, the running time of a simulation may exceed the

running time of the experiment on a real system. Furthermore, typical simulations, as well as direct system experimentation, require multiple instantiations of an experiment under the same conditions and parameters to obtain a reasonable good statistical reliability of the analysis. This factor can consume large amounts of time.

Fortunately, computer simulations can be parallelised, so that parts of the simulation can proceed concurrently in different computers. Communication protocols exist to support the construction of parallel programs running in computer clusters. The *de facto* standard is the Message Passing Interface (MPI), which consists of a library of routines for synchronising the computing activities of parallel nodes.

Different types of simulation and simulators exist. Most computer simulators are however discrete event driven.

2.3.2.1 Event driven simulation

The basic components of a computer simulator are the following:

- Entities model anything that may alter the state of the system (for example a data packet sent from one ACE to another). Entities can have a number of attributes, such as an identification number, their arrival time, and data contents.
- Queues keep entities until their processing time. Queues in computer simulation are often simple first-in-first-out (FIFO) queues. However, other types of queues are also useful for specific cases, for instance, priority queues.
- The systems' resources are responsible for processing entities waiting in the queues, so that they are either busy serving one entity or idle.
- External inputs drive the system, and outputs from the system can be measured.

The dynamics of the simulation are controlled by what are known as events. An event is anything that can modify the state of the system (e.g. the arrival or departure of an entity from a queue).

The simulation clock, on the other hand, is handled in one of two ways. Simulations may advance the simulated clock in fixed steps, possibly but not necessarily, simulating real time, or they may advance the clock time in chunks to match the next event in the simulation. Events are kept in a simulation event list, implemented as a priority queue (e.g. heaps). The simulation scheduler is responsible for advancing the simulated clock and executing the next event in the list. Because of the execution, a new event may be generated and inserted into the list. A simulation finishes once a predefined time limit is achieved or when no events are remaining in the event list.

An example of such an event driven computer simulator is Network Simulator 2 (NS2), already popular within the networking research community. The simulator covers a range of infrastructures including wired, wireless, and satellite networking. A large number of protocols are supported, and modules are easily implemented. This makes it possible to investigate the interaction of different protocols within the same network. Developed by the VINT project group, NS2 is an open source application. Programming is split into C++ for packet processing and the TCL scripting language for creating the simulations. It is possible to visualise the outputs using the NAM application. The networked environment envisioned by the CASCADAS project consists of wired and wireless networks. Many examples exist of wired and wireless routing protocols implemented and simulated using NS2. This makes it of significant interest to the CASCADAS project.

2.3.2.2 Component Simulators

Another approach to simulator implementation, not entirely incompatible with the event driven simulators described above, is to divide the system into reusable components, which can be linked using a standardised interface, through which the components can interact. Amongst other things, this facilitates the reuse of code.

This approach is in many ways analogous to the ACEs of the CASCADAS project. The ACEs will themselves have standardised interfaces and functionality, and an approach to simulating networks of ACEs would be to make full use of the ACEs' interfaces and connect them into the simulation environment which provides situational and environmental inputs to the ACEs and implements factors such as mobility and network connections between ACEs.

2.3.2.3 Real Time and Hybrid Approaches

Hybrid approaches of analytic, simulation, and experimental methods are also possible in the study of specific measures. For example, if the simulator operates in real time (that is, the events in the simulation are occurring at times corresponding to those of a real system), the inputs and outputs of a real experiment and the simulation can be combined in order to create a more complex system. This may be useful were equipment prices and budgets are restrictive, and the experimental system needs to be supplemented, or where the simulation is being used to control parameters which would be difficult to control in the real world, such as radio frequency background noise in a wireless networking environment.

2.3.2.4 Testbed experimentation and open-network testing

Although analytic and simulation methods have many advantages, for their construction, they need to rely on assumptions about the real system that may not be realistic enough to obtain accurate results or results that hold beyond the pre-stated conditions. In such cases, real-world experimentation constitutes an authoritative method to evaluate the performance of the system and its behaviour and the principal way CASCACAS' outputs will be evaluated.

There are two possible forms of real-world experimentation with ACEs. The first form is the implementation and deployment of the software components required for the ACEs in a local testbed or distributed testbed, where all nodes are controllable and observable. A testbed is a laboratory model of the intended real system. A second form, which constitutes the logical step forward from the previous one, is a deployment of ACEs on an open network (i.e. the Internet), possible in the form of a specific application with embedded ACEs. In such a case, evaluation can take place at the same time ACEs perform useful work for real users. Such form of testing provides the ultimate form of realism for the system. However, evaluations will need to be restricted to simple performance measurement studies given beyond a testbed, as the system is no longer totally controllable. Furthermore, open-network studies might take years to compete. Testbed evaluation is expected to be an important component in the development of the project and open-network experimentation will be considered after the release of the CASCADAS toolkit. Part D6.1A provided a description of the interconnected testbed that will be used to experimentally evaluate ACEs.

Real system experimentation poses challenges not present in simulation. The most relevant are:

- Complex logistics
  Experimental evaluation is limited by the resources available (e.g. number of computers), which limit the scope of the tests. A dedicated testbed for long-run experiments is sometimes difficult to manage as most testbeds are shared to run experiments for diverse

projects. For example, computers in PlanetLab, a worldwide overlay testbed, are time-shared by many users.

Compatibility issues in developing languages, versioning and operating systems are another source of possible problems, particularly for distributed testbeds of unrelated organisations.

- Traffic controlling
  As the testbed is a model of the real system, adjustments may need to be made to the behaviour of the nodes and links that make up the network. This could be in the form of introducing artificial delays to links in order to simulate longer, more geographically diverse connections, scaling link capacities, or even introducing artificial loss in order to mimic a poor wireless connection.

- Manageability and Repeatability
  Experiments need to be repeated and their parameters adjusted to the same conditions (reset) with every run to obtain good statistical reliability of measurements. For autonomic systems, these factors pose a particularly challenging problem as they would require resetting a quite diverse set of inputs to attain producing a comparable sequence of aggregation decisions every time. Unlike simulations, real-world experiments cannot be reproduced perfectly simply because not all conditions are manageable.

  Consider, for example, the problems in reproducing user activity or the case of a network of ACEs receiving input from a sensor network, which reports the light intensity on a street. It is highly unlikely that sensors will report the same levels over the long period of time required by the several instantiations of an experiment.

  In experiments involving physical mobility of nodes, and with them ACEs, reproducing the exact position of the nodes for each experiment is also difficult to achieve, not only because the technical problems involved, but also because it brings in extra logistic problems to move a large number of nodes. Nevertheless, some experiments may be repeatable to a certain degree of error tolerance to produce consistent results with a limited set of parameters.

- Distributed measurement environment
  In a highly collaborative environment, such as the one expected to emerge from a network of ACEs, most measures will not be available from a single location but rather, they will require acquisition from multiple sites. The distributed nature at hand introduces not only difficulties in the evaluation of measures but also errors. A collection mechanism will be required to collect data, which will most likely pollute measurements.

- Need for security mechanisms in distributed testbeds
  Distributed testbeds that utilised shared networks to interconnect are exposed to security threads. Alternatives exist to obtain interconnection with an acceptable level of security (e.g. by creating a VPN). However, it is important to notice that such approaches introduce an extra overhead in the processing of packets flowing from one testbed to another as security computation need to be in place, which might affect certain the collection of certain measures, for example, when involving response time.

2.3.2.5 Experimental methods in the literature of mobile ad hoc networks

Mobile ad hoc networks (MANET) provide an example of a technology where nodes operate in extreme conditions: short lifetime of nodes, uncontrollable mobility, interference and

communication obstacles, in a way very similar to the envisioned environment of CASCADAS. Therefore, it is of relevant interest to the project take into account the experience of constructing MANET testbeds as reported in literature. Moreover, the MANET testbeds that have appeared in recent years provide an excellent example of the challenges involved in real-world system experimentation.

A small number of approaches deal with testing on real implemented systems without major artificial aids. A representative example of this method was applied by Maltz et al., who conducted a number of trials aimed at demonstrating the operation of DSR on a mobile network [49]. The network consisted of five moving nodes (vehicles) and two stationary nodes. The vehicles approximately followed a pre-planned course around CMU campus. The drawback of this method is that experiments are naturally hard to reproduce as uncontrollable factors play a major role (e.g. electromagnetic conditions or mobiles' speeds). A partial solution aimed at facilitating the creation of reproducible experiments was introduced by Lundgren et al., who suggested the use of choreography scripts [50] with instructions that people may follow to move around. However, the methodology does not offer facilities for large-scale experimentation, as it requires one person per mobile node. Another example of the direct method is Roofnet, a large-scale multi-hop IEEE 802.11b testbed [51] that provides experimental Internet connectivity to a group of users. The nodes of Roofnet do not move, as they are located on top of apartment buildings.

A common problem of the direct method is the lack of reproducibility of experiments, which is a result of the difficulties in controlling node motion and other parameters. Another typical problem is the need for relatively large areas to operate, which is something usually inconvenient. For this reason, many researchers have turned their attention to an indirect method, which inserts some form of artificial component to increase the controllability of parameters.

Another set of methods, which could be classified as indirect methods, do not deal with a totally realistic system as they introduce some kind of artificial aids to deal with the disadvantages of the direct method.

In terms of reducing the physical area for testing wireless networks, a clear example is due by Kaba and Raiche [52], who proposed reducing the wireless range of radio transmitters by attenuating the transmitting power of network cards with the use of external low-gain antennas. By reducing the wireless range of radios, multi-hop conditions may be established within a single room. However, their approach is restrictive as it requires special hardware (i.e. the availability of network cards able to accept an external antenna) and does not solve the issue of reproducibility of mobile experiments.

Sanghani et al. [53] employed a similar approach but recreated mobility with an RF MUX to setup connectivity. An interesting approach is undertaken by the Orbit project [54]. A grid of 20 by 20 grid wireless computers was deployed, where each node can take the identity of a mobile node. The advantage of the approach is that realistic access scenarios can be tested (e.g. hidden node problems) but mobility is restricted to discrete steps and confined to the size of the grid.

A final set of methods introduces simulation into the test process. Certain features of the experiments are simulated while other parts remain as before (emulation), making them a viable and inexpensive alternative for testbeds. A popular and simple approach is to filter packets before they reach the routing module of the node. In these terms, Zhang et al. [55] proposed Mobiemu, a master/slave architecture that emulates the connectivity of a wireless network. A master controller keeps a unique clock that defines the entire network-evolving pace. For this, the master and each slave are given the same instructions to follow (i.e. a scenario file). As the emulation progresses, the master broadcast a packet with its current

simulated timestamp every time its clock reaches a control rule in the scenario file. This packet allows the slaves to know when to execute rules by matching the timestamp received. Neighbours are then added or removed via the *iptables* Linux command to interact with the *netfilter* module installed in the kernel.

A more sophisticated system also based on emulation principles is MTM [56], which provides a realistic radio propagation model that uses raytracing for obstacle support. In addition, to simulate connectivity, other essential parameters are also simulated: energy consumption, node location, etc. Most parameters become available through system's agents to routing protocols that require specific information to operate. For example, location aware protocols may query their agent for local information. The system doubles as a visualisation centre and interactive input for network activities.

3 Conclusions

A principal objective of CASCADAS is the creation of an autonomic toolkit that will enable services offering self-* properties to emerge. The toolkit will allow the instantiation of autonomic communication elements (ACE), which will offer self-* properties to facilitate the creation, execution and provisioning of situation-aware and dynamically adaptable communication services at a certain quality of service. Unmistakably, proper evaluation tools will be required to measure service efficiency.

This document has explored various technical evaluation approaches applicable to ACEs and their interactions to obtain measures of variables of interest. The purpose of this work has been the establishment of a reference framework for the future evaluation of ACEs. Such a framework will contribute to the work of approaching months of other CASCADAS researchers to verify the autonomic properties of an ACE design and to benchmark performance against alternative systems.