

Report from Dagstuhl Seminar 14512

Collective Adaptive Systems: Qualitative and Quantitative Modelling and Analysis

Edited by

Jane Hillston¹, Jeremy Pitt², Martin Wirsing³, and Franco Zambonelli⁴

1 University of Edinburgh, GB, Jane.Hillston@ed.ac.uk

2 Imperial College London, GB, j.pitt@imperial.ac.uk

3 LMU München, DE, wirsing@lmu.de

4 University of Modena, IT, franco.zambonelli@unimore.it

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 14512 “Collective Adaptive Systems: Qualitative and Quantitative Modelling and Analysis”. Besides presentations on current work in the area, the seminar focused on the following topics: (i) Modelling techniques and languages for collective adaptive systems based on the above formalisms. (ii) Verification of collective adaptive systems. (iii) Humans-in-the-loop in collective adaptive systems.

Seminar December 14–19, 2014 – <http://www.dagstuhl.de/14512>

1998 ACM Subject Classification C.2.4 Distributed Systems, D.2 Software Engineering, D.2.4 Software/Program Verification, H.1.2 User/Machine Systems

Keywords and phrases Collective Adaptive Systems, Qualitative and Quantitative Modelling and Analysis, Verification, Humans-In-The-Loop

Digital Object Identifier 10.4230/DagRep.4.12.68

Edited in cooperation with Lenz Belzner

1 Executive Summary

Jane Hillston

Jeremy Pitt

Martin Wirsing

Franco Zambonelli

License © Creative Commons BY 3.0 Unported license

© Jane Hillston, Jeremy Pitt, Martin Wirsing, and Franco Zambonelli

Modern systems are often structured as complex, multi-layered networks of interconnected parts, where different layers interact and influence each other in intricate and sometimes unforeseen ways. It is infeasible for human operators to constantly monitor these interactions and to adjust the system to cope with unexpected circumstances; instead systems have to adapt autonomously to dynamically changing situations while still respecting their design constraints and requirements. Because of the distributed and decentralized nature of modern systems, this usually has to be achieved by collective adaptation of the nodes comprising the system. In open systems exhibiting collective adaptation, unforeseen events and properties can arise, e.g. as side effects of the interaction of the components or the environment. Modelling and engineering collective adaptive systems (CAS) has to take into account such “emergent” properties in addition to satisfying functional and quantitative requirements.



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Collective Adaptive Systems: Qualitative and Quantitative Modelling and Analysis, *Dagstuhl Reports*, Vol. 4, Issue 12, pp. 68–113

Editors: Jane Hillston, Jeremy Pitt, Martin Wirsing, and Franco Zambonelli



DAGSTUHL
REPORTS Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Finding ways to understand and design CAS, and to predict their behaviour, is a difficult but important endeavour. One goal of this seminar was to investigate techniques for modelling and analysing systems that adapt collectively to dynamically changing environment conditions and requirements. In many cases, these models and analysis techniques should not only capture qualitative properties of the system, such as absence of deadlocks, they should also be able to express quantitative properties such as quality of service.

Research on CAS builds on and integrates previous research efforts from several areas:

- Formal foundations and modelling techniques for concurrent systems deal with problems such as enabling and limiting concurrency, access to shared resources, avoidance of anomalies, communication between processes, and estimation of performance.
- Analysis of concurrent systems typically exploits such notions as bisimilarity of different processes or reasons on stochastic properties of systems consisting of many equivalent processes.
- The area of adaptive systems also investigates systems consisting of interacting entities, but is more concerned with the reaction of whole systems or individual actors in a system to a changing environment.

An important aim of this seminar was to combine research from concurrent systems with results from the adaptive systems community in order to develop formalisms for specifying CAS, to increase the scalability of qualitative and quantitative modelling and analysis techniques to large systems, and to apply them to systems that dynamically change their structure or adapt to novel situations.

The seminar was organised with a mixture of talks and working group sessions which facilitated more in-depth discussions and exploration of topics. In this report we include the abstracts of a selection of the presented talks, and three longer contributions compiled after the meeting which seek to reflect the activities of the working groups. The first group, considering modelling, specification and programming for CAS, start their presentation with brief descriptions of four diverse applications developed on the basis of CAS, ranging from national level power management to personal wearable devices. To complement this identification of application domains, the group also catalogued common and contrasting features that can be found in CAS. This consideration highlights the role of physical space in all the considered domains and the urgent need to develop modelling and analysis techniques which reflect this central role played by space. This was key amongst a number of challenges identified by the group in their conclusions. Spatio-temporal aspects were also identified as a key challenge by the second working group who considered verification of CAS. The report from this group outlines the role of verification within the design and management of CAS ranging from seeking to guarantee global emergent behaviour from local specifications to using online verification to drive adaptation. Two specific challenges were explored in more detail, namely handling the inherent uncertainty in CAS, and specification and verification of spatial properties of systems composed of self-organising patterns. The third working group focused on the issues that arise from the recognition that some of the entities within a CAS may be humans and outside technological control, i.e. the design of socio-technical systems. A number of different scenarios are provided to illustrate the difference between socio-technical CAS and ‘technical’ CAS, and the human factors which must be taken into account. To remediate some of the problems identified, the group propose the idea of a general intervention framework, based around the 3I life-cycle – inspection-innovation-intervention. It was foreseen that intervention would be achieved by shaping mechanisms, and the report goes on to describe some possible shaping mechanisms which were considered. To conclude a number of research challenges are discussed.

2 Table of Contents

Executive Summary

Jane Hillston, Jeremy Pitt, Martin Wirsing, and Franco Zambonelli 68

Overview of Talks

Creating Predictable Collective Behaviors with Aggregate Programming

Jacob Beal 72

Algebraic Reinforcement Learning

Lenz Belzner 72

Dynamic change of collaboration patterns: motivations and perspectives

Giacomo Cabri 73

A formal approach to autonomic systems programming: The SCEL Language

Rocco De Nicola 73

Discrete Time Markovian Agents

Marco Gribaudo 74

On bootstrapping sensori-motor patterns for a constructivist learning system in continuous environments

Salima Hassas 74

Challenges for Quantitative Analysis of Collective Adaptive Systems

Jane Hillston 75

Role-based Adaptation

Annabelle Klarl 75

Diversity, Heterogeneity and Dynamics in Collective Systems

Peter Lewis 76

Modelling Collective Adaptive Systems in CARMA

Michele Loreti 76

Stochastic Coordination in CAS: Expressiveness and Predictability

Stefano Mariani 76

On-the-fly Fast Mean Field Model Checking for Collective Adaptive Systems

Mieke Massink 77

Declarative vs. Procedural Approach for SCSP with an Application to an E-mobility Optimization Problem

Ugo Montanari 78

Procedural Justice and ‘Fitness for Purpose’ of Self-Organising Electronic Institutions

Jeremy Pitt 78

LollyScript, a concurrent programming language to ensure that promises are kept

Christophe Scholliers 79

Testing as a useful complement to verification of SOAS?

Hella Seebach 79

How Collective and Adaptive our CAS are?


Nikola Serbedzija 80

Three Behavioural Equivalences for Chemical Reaction Networks <i>Mirco Tribastone</i>	80
Engineering Autonomous Ensembles <i>Martin Wirsing</i>	81
Smart Cities as Heterogeneous Superorganisms: Scenarios and Challenges <i>Franco Zambonelli</i>	81
Working Group Reports	
Modelling, Specification, and Programming for Collective Adaptive Systems <i>Hella Seebach, Lenz Belzner, Marco Gribaudo, Anabelle Klarl, Michele Loreti, Ugo Montanari, Laura Nenzi, Rocco De Nicola, Christophe Scholliers, Petr Tuma, and Martin Wirsing</i>	82
Verification of CAS <i>Luca Bortolussi, Giacomo Cabri, Giovanna Di Marzo Serugendo, Vashti Galpin, Jane Hillston, Roberta Lanciani, Mieke Massink, Mirco Tribastone, and Danny Weyns</i>	91
Humans-in-the-Loop in Collective Adaptive Systems <i>Jake Beal, Peter Lewis, Stefano Mariani, Jeremy Pitt, Nikola Serbedzija, and Franco Zambonelli</i>	102
Participants	113

3 Overview of Talks

3.1 Creating Predictable Collective Behaviors with Aggregate Programming


Jacob Beal (BBN Technologies – Cambridge, US)

License  Creative Commons BY 3.0 Unported license
© Jacob Beal

Practical collective adaptive systems typically comprise many different interacting elements, with a great degree of heterogeneity in the activities required of any given element and the capabilities of different elements. This tends to make engineering such systems extremely difficult, and proving properties about the engineered systems effectively impossible. Recently developed methods in aggregate programming, however, offer the possibility of creating “operator algebras” in any collective adaptive system created using a fairly general API is proved by construction to have desirable adaptive properties such as self-stabilization, scalability, and toleration of network perturbation. These methods thus offer a path to engineering systems that exhibit implicit, safe, and ubiquitous adaptivity.

3.2 Algebraic Reinforcement Learning

Lenz Belzner (LMU München, DE)

License  Creative Commons BY 3.0 Unported license
© Lenz Belzner

This talk expands on support for decision making in autonomous adaptive systems by identifying proper qualitative state abstractions through quantitative (i.e. statistical) analysis of environment data sampled at system runtime. The TG relational reinforcement learning algorithm [1] learns relational decision trees by statistical evaluation of runtime data. Here, qualitative state abstractions to be analyzed statistically are specified manually and a-priori.

The talk introduces a quantifiable metric for adaptation in the context of learning systems to allow for quantitative evaluation of adaptation. By identifying operators for model modification and evaluation, the relation of relational reinforcement learning to evolutionary programming is shown. An approach for automatic extraction of relevant qualitative abstractions via algebraic term generalization with the ACUOS system [2] is presented.

References

- 1 Driessens, K., Ramon, J., Blockeel, H.: *Speeding up relational reinforcement learning through the use of an incremental first order decision tree learner*. In: Machine Learning: ECML 2001. Springer (2001) 97–108
- 2 Alpuente, M., Escobar, S., Espert, J., Meseguer, J.: *Acuos: A system for modular acu generalization with subtyping and inheritance*. In Ferm, E., Leite, J., eds.: Logics in Artificial Intelligence. Volume 8761 of Lecture Notes in Computer Science. Springer International Publishing (2014) 573–581

3.3 Dynamic change of collaboration patterns: motivations and perspectives

Giacomo Cabri (University of Modena, IT)

License © Creative Commons BY 3.0 Unported license
© Giacomo Cabri

Joint work of Cabri, Giacomo; Capodieci, Nicola; Zambonelli, Franco; Puviani, Mariachiara
Main reference G. Cabri, N. Capodieci, “Runtime Change of Collaboration Patterns in Autonomic Systems: Motivations and Perspectives,” in Proc. of the 2013 27th Int’l Conf. on Advanced Information Networking and Applications Workshops (WAINA’13), pp. 1038–1043, IEEE, 2013.
URL <http://dx.doi.org/10.1109/WAINA.2013.82>

Today’s complex distributed systems must adapt to the unexpected execution conditions they face, in an autonomous way. This requires not only the adaptation feature at component level, but also the capability of adapting at the system level, modifying the collaboration pattern among components [3]. In this talk I will introduce the scenario, motivate the need for collaboration pattern changes at runtime [1], and propose some approaches to enact them, one based on formal models, one based on roles [4], and one bio-inspired [2].

References

- 1 Giacomo Cabri and Nicola Capodieci. *Runtime Change of Collaboration Patterns in Autonomic Systems: Motivations and Perspectives*. In *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, IEEE, Piscataway, NJ, USA, 1038–1043.
- 2 Nicola Capodieci, Emma Hart and Giacomo Cabri. *Artificial Immune System driven evolution in Swarm Chemistry*. Proceedings of The Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems SASO 2014.
- 3 Franco Zambonelli, Nicola Bicchieri, Giacomo Cabri, Letizia Leonardi and Mariachiara Puviani. *On self-adaptation, self-expression, and self-awareness in autonomic service component ensembles*. In *Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference on*. IEEE, IEEE, Piscataway, NJ, USA, 108–113.
- 4 Mariachiara Puviani and Giacomo Cabri and Letizia Leonardi. *Enabling Self-expression: the Use of Roles to Dynamically Change Adaptation Patterns*. Proceedings of the Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshop(SASOW 14), London, UK, 8-12 September 2014.

3.4 A formal approach to autonomic systems programming: The SCEL Language

Rocco De Nicola (IMT Lucca, IT)

License © Creative Commons BY 3.0 Unported license
© Rocco De Nicola

The autonomic computing paradigm has been proposed to cope with size, complexity and dynamism of contemporary software-intensive systems. The challenge for language designers is to devise appropriate abstractions and linguistic primitives to deal with the large dimension of systems, and with their need to adapt to the changes of the working environment and to the evolving requirements. We introduced a set of programming abstractions that permit to represent behaviours, knowledge and aggregations according to specific policies, and to support programming context-awareness, self-awareness and adaptation. Based on these abstractions, we described SCEL (Software Component Ensemble Language), a kernel language whose

solid semantic foundations lay also the basis for formal reasoning on autonomic systems behaviour. To show expressiveness and effectiveness of SCEL's design, we presented a Java implementation of the proposed abstractions and showed how it has been exploited for programming a robotics scenario used as a running example for describing features and potentials of our approach.

3.5 Discrete Time Markovian Agents

Marco Gribaudo (Politecnico di Milano, IT)

License © Creative Commons BY 3.0 Unported license
© Marco Gribaudo

Markovian Agents is a formalism that has been used to model large systems composed by interacting entities. Agents interact using a mechanism based on what is called “Induction”: the states in which neighbor agents are, influences the transition rates. The concept is quite natural in continuous time, and it is supported by strong theory coming from mean-field analysis and spatial Poisson processes. The transition to discrete time however is not trivial, and opens new questions and new possibilities.

3.6 On bootstrapping sensori-motor patterns for a constructivist learning system in continuous environments

Salima Hassas (University Claude Bernard – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Salima Hassas

Joint work of Mazac, Sébastien; Armetta, Frédéric; Hassas, Salima
Main reference S. Mazac, F. Armetta, S. Hassas, “On Bootstrapping Sensori-Motor Patterns for a Constructivist Learning System in Continuous Environments,” in Proc. of the 14th Int'l Conf. on the Synthesis and Simulation of Living Systems (ALIFE'14), pp. 160–167, MIT Press, 2014.
URL <http://dx.doi.org/10.7551/978-0-262-32621-6-ch028>

The theory of cognitive development from Jean Piaget (1923) is a constructivist perspective of learning that has substantially influenced cognitive science domain. Within AI, lots of works have tried to take inspiration from this paradigm since the beginning of the discipline. Indeed it seems that constructivism is a possible trail in order to overcome the limitations of classical techniques stemming from cognitivism or connectionism and create autonomous agents, fitted with strong adaptation ability within their environment, modelled on biological organisms. Potential applications concern intelligent agents in interaction with a complex environment, with objectives that cannot be predefined. Like robotics, Ambient Intelligence (AmI) is a rich and ambitious paradigm that represents a high complexity challenge for AI. In particular, as a part of constructivist theory, the agent has to build a representation of the world that relies on the learning of sensori-motor patterns starting from its own experience only. This step is difficult to set up for systems in continuous environments, using raw data from sensors without a priori modelling. With the use of multi-agent systems, we investigate the development of new techniques in order to adapt constructivist approach of learning on actual cases. Therefore, we use ambient intelligence as a reference domain for the application of our approach.

3.7 Challenges for Quantitative Analysis of Collective Adaptive Systems

Jane Hillston (University of Edinburgh, GB)

License © Creative Commons BY 3.0 Unported license
© Jane Hillston

Main reference J. Hillston, “Challenges for Quantitative Analysis of Collective Adaptive Systems,” in Proc. of the 8th Int’l Symp. on Trustworthy Global Computing (TGC’13), LNCS, Vol. 8358, pp. 14–21, Springer, 2014.

URL http://dx.doi.org/10.1007/978-3-319-05119-2_2

Quantitative analysis plays an important role in the design of systems as it allows us to predict their dynamic behaviour. Thus in addition to the functional properties that can be assessed by qualitative analysis we can also investigate properties such the timeliness of response and the efficient and fair access to resources. However the scale of collective adaptive systems imposes serious challenges on the usual approaches to quantitative analysis which are based on discrete state representations. In this talk I will talk about these challenges and explain how in some circumstances making a fluid approximation of the discrete state space can be beneficial.

3.8 Role-based Adaptation

Annabelle Klarl (LMU München, DE)

License © Creative Commons BY 3.0 Unported license
© Annabelle Klarl

Joint work of Klarl, Annabelle; Hennicker, Rolf


A self-adaptive component keeps track of its individual and shared goals, perceives its internal state as well as its environment, and adapts its behavior accordingly. Based on these characteristic features, we propose a pragmatic methodology to develop self-adaptive systems from specification to design. We specify the system’s adaptation logic by adaptation automata. A design model refines the specification by adding application logic and providing an architecture. We take inspiration from the autonomic manager pattern [2] where an adaptation manager is employed on an adaptable component to control appropriate behavioral adaptation in response to observations of the environment. To realize the architecture of the autonomic manager pattern, we make use of the Helena modeling approach [1] to encapsulate the manager, sensors of the environment, and different behavioral modes of the component into roles applied to the component. The system design therefore gets structured into self-contained roles providing a clear architecture separating adaptation logic and application logic.

References

- 1 Rolf Hennicker and Annabelle Klarl, Foundations for Ensemble Modeling – The Helena Approach, in Specification, Algebra, and Software, ser. LNCS, vol. 8373. Springer, 2014, pp. 359–381.
- 2 Mariachiara Puviani, Giacomo Cabri, and Franco Zambonelli, A taxonomy of architectural patterns for self-adaptive systems, in International C* Conference on Computer Science and Software Engineering. ACM, 2013, pp. 77–85.

3.9 Diversity, Heterogeneity and Dynamics in Collective Systems

Peter Lewis (Aston University Birmingham, UK)

License  Creative Commons BY 3.0 Unported license
© Peter Lewis

Diversity plays an important role in many natural and engineered systems. In this talk, I will describe two different forms of diversity present in engineered collective systems: (i) heterogeneity (genotypic/phenotypic diversity) and (ii) dynamics (temporal diversity). I will discuss these forms of diversity through two qualitatively different case studies (smart camera networks and particle swarm optimisation). The analysis shows that both forms of diversity can be beneficial in very different problem and application domains, and can indeed impact more than the ability of the collective to adapt. I will end by raising some questions regarding how to engineer effective diversity in collective systems.

3.10 Modelling Collective Adaptive Systems in CARMA

Michele Loreti (University of Firenze, IT)

License  Creative Commons BY 3.0 Unported license
© Michele Loreti

Joint work of Bortolussi, Luca; De Nicola, Rocco; Galpin, Vashti; Gilmore, Stephen; Hillston, Jane; Latella, Diego; Loreti, Michele; Massink, Mieke

In this talk we present CARMA, a language recently defined to support specification and analysis of collective adaptive systems. CARMA is a stochastic process algebra equipped with linguistic constructs specifically developed for modelling and programming systems that can operate in open-ended and unpredictable environments. This class of systems is typically composed of a huge number of interacting agents that dynamically adjust and combine their behaviour to achieve specific goals. A CARMA model, termed a collective, consists of a set of components, each of which exhibits a set of attributes. To model dynamic aggregations, which are sometimes referred to as ensembles, CARMA provides communication primitives that are based on predicates over the exhibited attributes. These predicates are used to select the participants in a communication. Two communication mechanisms are provided in the CARMA language: multicast-based and unicast-based. In the talk, we first introduce the basic principles of CARMA and then we show how our language can be used to support specification with a simple but illustrative example of a socio-technical collective adaptive system.

3.11 Stochastic Coordination in CAS: Expressiveness and Predictability

Stefano Mariani (Università di Bologna, IT)

License  Creative Commons BY 3.0 Unported license
© Stefano Mariani

Joint work of Omicini, Andrea; Mariani, Stefano

Recognising that (i) coordination is a fundamental concern when both analysing and modelling CAS, and that (ii) CAS often exhibit stochastic behaviours, stemming from probabilistic and time-dependent local (interaction) mechanisms, in this talk we argue that (a) measuring

expressiveness of coordination languages, and (b) predicting behaviours of stochastic systems based on coordination models are two fundamental steps in the quest for designing well-engineered CAS. As a concrete ground where to or discussion, we describe some of our current works as well as our ideas for further research.

3.12 On-the-fly Fast Mean Field Model Checking for Collective Adaptive Systems

Mieke Massink (CNR – Pisa, IT)

License © Creative Commons BY 3.0 Unported license

© Mieke Massink

Joint work of Latella, Diego; Loreti, Michele; Massink, Mieke

Main reference D. Latella, M. Loreti, M. Massink, “On-the-fly fast mean-field model-checking,” in Proc. of the 8th Int’l Symp. on Trustworthy Global Computing (TGC’13), LNCS, Vol. 8358, pp. 297–314, Springer, 2014.

URL http://dx.doi.org/10.1007/978-3-319-05119-2_17

Typical self-organising collective systems consist of a large number of interacting objects that coordinate their activities in a decentralised and often implicit way. Design of such systems is challenging and requires suitable, scalable analysis tools to check properties of proposed system designs before they are put into operation. Model checking has shown to be a successful technique for the verification of distributed and concurrent systems, but in the context of collective systems we need these techniques to be highly scalable. Model checking approaches can be divided into two broad categories: global approaches that determine the set of all states in a model M that satisfy a temporal logic formula F , and local approaches in which, given a state s in M , the procedure determines whether s satisfies F . When s is a term of a process language, the model-checking procedure can be executed “on-the-fly”, driven by the syntactical structure of s . For certain classes of systems, e.g. those composed of many parallel components, the local approach is preferable because, depending on the specific property, it may be sufficient to generate and inspect only a relatively small part of the state space. Recently global stochastic model-checking approaches for collective systems have been explored, combining fast simulation and fluid approximation in a continuous time setting, for example in the work by Bortolussi and Hillston. In this presentation we explore the use of on-the-fly techniques in this direction in a discrete time setting. We first present an efficient, probabilistic, on-the-fly, PCTL model checking procedure that is parametric with respect to the semantic interpretation of the language. The procedure comprises both bounded and unbounded until modalities. The correctness of the procedure is shown and its efficiency has been explored on a number of benchmark applications in comparison with the global PCTL model checker PRISM. We then show how to instantiate the procedure with a mean field semantics to verify bounded PCTL properties of selected individuals in the context of very large systems of independent interacting objects. The asymptotic correctness of the procedure is shown and some results of the application of a prototype implementation of the FlyFast model-checker will be presented.

References

- 1 Latella, D., Loreti, M., Massink, M.: On-the-fly fast mean-field model-checking. In: Abadi, M., Lluch-Lafuente, A. (eds.) Trustworthy Global Computing – 8th Int’l Symp., TGC 2013, Buenos Aires, Argentina, August 30–31, 2013, Revised Selected Papers. LNCS, vol. 8358, pp. 297–314. Springer (2014), http://dx.doi.org/10.1007/978-3-319-05119-2_17
- 2 Latella, D., Loreti, M., Massink, M.: On-the-fly probabilistic model-checking. In: Proceedings 7th Interaction and Concurrency Experience ICE 2014. EPTCS, vol. 166 (2014)

3.13 Declarative vs. Procedural Approach for SCSP with an Application to an E-mobility Optimization Problem

Ugo Montanari (University of Pisa, IT)

License © Creative Commons BY 3.0 Unported license
© Ugo Montanari

Main reference N. Hoch, G. V. Monreale, U. Montanari, M. Sammartino, A. Tcheukam Siwe, “From Local to Global Knowledge and Back,” in M. Wirsing, M. Hölzl, N. Koch, P. Mayer (eds.), “Software Engineering for Collective Autonomic Systems – The ASCENS Approach,” LNCS, Vol. 8998, pp. 185–220, Springer, 2015.

URL http://dx.doi.org/10.1007/978-3-319-16310-9_5

Large optimization problems tend to be overly complex to solve and often a globally optimal solution may be impossible to find. For this reason specific strategies are needed to solve them. We propose an approach for the coordination of declarative knowledge – that is the exact specification of the complete optimization problem – and of procedural knowledge – that is the specific knowledge about subproblems and their, possibly approximated, solution strategies. We consider Soft Constraint Satisfaction Problems (SCSPs) and we introduce a formalism, similar to a process calculus, for their specification. Cost functions are associated to terms and form a model of such specification, where operators are interpreted as optimization steps. We compare our approach with Courcelle’s approach for efficient monadic second-order evaluations on tree composable graphs. We apply our approach to a problem studied in the ASCENS e-mobility case study, for which we provide a model in terms of cost functions. The procedural part concerns heuristic choices about which dynamic programming strategy should be employed and how different ad-hoc approximation heuristics could be applied.

3.14 Procedural Justice and ‘Fitness for Purpose’ of Self-Organising Electronic Institutions

Jeremy Pitt (Imperial College London, UK)

License © Creative Commons BY 3.0 Unported license
© Jeremy Pitt

In many multi-agent systems, it is a commonplace requirement to distribute a pool of collectivised resources amongst those agents. One way to address typical problems, like unrestrained resource access, or to ensure some desirable property, like fairness, is for the agents to mutually agree a set of rules to self-organise and self-regulate the distribution process. We propose a framework for measuring the ‘fitness for purpose’ of such a set of rules, as represented in the Event Calculus. This framework encapsulates metrics for principles of participation, transparency and balancing, as derived from various conceptions of procedural justice. We define a metric for the empowerment aspect of the participation principle, and report some experimental results which show how this metric can reveal an inherent ‘fairness’ or ‘unfairness’ in the distribution of (institutionalised) power over time, and inform decision-making or rule-adaptation accordingly. We conclude with some discussion of how procedural justice can be used for analysis of collective adaptive systems.

3.15 LollyScript, a concurrent programming language to ensure that promises are kept

Christophe Scholliers (Free University of Brussels, BE)

License  Creative Commons BY 3.0 Unported license
© Christophe Scholliers

It is difficult to reason about the synchronisation between a set of concurrently executing tasks. One of the main reasons for this complexity is that the amount of possible states a system can be in increases exponentially with the amount of concurrent tasks that are executing at the same time. Programming languages abstractions can help the programmer to prune the state space by eliminating those states that lead to inconsistencies. In this talk we will focus on the use of promises and futures to structure the asynchronous communication between two tasks. Unfortunately, in current systems the use of promises can easily lead to deadlocks. For example, the programming model can not ensure that all the promises in the system will be resolved. In this talk we present a concurrent programming language with a linear type system to statically verify the correct use of promises in concurrent programs.

3.16 Testing as a useful complement to verification of SOAS?

Hella Seebach (Universität Augsburg, DE)

License  Creative Commons BY 3.0 Unported license
© Hella Seebach

Joint work of Seebach, Hella; Nafz, Florian; Eberhardinger, Benedikt; Reif, Wolfgang

Self-organization and adaptivity are important techniques for building flexible and robust systems. Though, applying verification and testing is crucial for their acceptance. We propose techniques (software engineering guideline, coalition formation, compositional reasoning, verified result checking, etc.) for the construction and partial verification of self-organizing resource-flow systems. These techniques allow for example to reason about global properties by verifying single agent properties. In this talk, I want to discuss in which way new techniques for testing SOAS can be a complement to further extend the quality assurance of our partially verified system.

References

- 1 Hella Seebach, Florian Nafz, Jan-Philipp Steghöfer, Wolfgang Reif *How to Design and Implement Self-organising Resource-Flow Systems*. Organic Computing – A Paradigm Shift for Complex Systems, Autonomic Systems, Birkhäuser, Springer
- 2 Florian Nafz, Jan-Philipp Steghöfer, Hella Seebach, Wolfgang Reif. *Formal Modeling and Verification of Self-* Systems Based on Observer/Controller-Architectures*. Springer-Verlag, Berlin, Heidelberg, 2013
- 3 Benedikt Eberhardinger, Hella Seebach, Alexander Knapp, Wolfgang Reif. *Towards Testing Self-Organizing, Adaptive Systems*. Proceedings of the 26th IFIP WG 6.1 International Conference (ICTSS 2014), Lecture Notes in Computer Science (accepted), Springer

3.17 How Collective and Adaptive our CAS are?

Nikola Serbedzija (FhG FOKUS – Berlin, DE)

License © Creative Commons BY 3.0 Unported license
© Nikola Serbedzija

Main reference N. B. Serbedzija, “Autonomous Systems and Their Impact on Us,” in R. de Nicola, R. Nennicker (eds.), “Software, Services, and Systems – Essays Dedicated to Martin Wirsing on the Occasion of His Retirement from the Chair of Programming and Software Engineering”, LNCS, Vol. 8950, pp. 662–675, Springer, 2015.

URL http://dx.doi.org/10.1007/978-3-319-15545-6_37

The talk examines the computing principles inspired by the nature (in a broader sense) and explores the design and deployment issues of technical systems that interfere with individuals and/or societies (placing humans directly into the processing loop). Due to inevitable impact that smart, mobile and web technologies have on individual and social behavior, inclusion of humanities in the early design phase is condition sine qua non. The talk further explore the possibilities of enriching current collective adaptive approaches with some concepts from social sciences and psychology.

3.18 Three Behavioural Equivalences for Chemical Reaction Networks

Mirco Tribastone (University of Southampton, UK)

License © Creative Commons BY 3.0 Unported license
© Mirco Tribastone

Joint work of Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin

Chemical reaction networks (CRNs) can be seen as a compact language for parallel computation, where the output of an algorithm is given by the concentration of the species at some point in time according to an underlying semantics based on continuous-time Markov chains (CTMCs) or on ordinary differential equations (ODEs).

Using a species-as-process analogy, we study behavioural equivalences over species of a CRN inspired by traditional approaches in established models of computation such as labelled transition systems. We define three equivalences in the Larsen-Skou style of probabilistic bisimulation that identify a partition of the species such that the dynamics of a CRN can be described only in terms of the equivalence classes. In Exact Fluid Lumpability, equivalent species have the same ODE solutions when starting from identical initial conditions. In Differential Species Bisimulation, each equivalence class represents the exact sum of the ODE trajectories of its member species. In Markovian Species Bisimulation, a partition over species identifies an exact aggregation in terms of ordinary lumpability over the states of the underlying CTMC.

For each equivalence relation we develop an efficient partition-refinement algorithm for computing the coarsest aggregations. Using a prototypal implementation, we find significant reductions in a number of models of biological processes available in the literature.

3.19 Engineering Autonomous Ensembles

Martin Wirsing (LMU München, DE)

License © Creative Commons BY 3.0 Unported license
© Martin Wirsing

Joint work of Wirsing, Martin; Hölzl, Matthias; Tribastone, Mirco; Zambonelli, Franco
Main reference M. Wirsing, M. M. Hölzl, M. Tribastone, F. Zambonelli, “ASCENS: Engineering Autonomic Service-Component Ensembles,” in Proc. of the 10th Int’l Symp. on Formal Methods for Components and Objects (FMCO’11), LNCS, Vol. 7542, pp. 1–24, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-35887-6_1

Today’s developers often face the demanding task of developing software for ensembles: systems with massive numbers of nodes, operating in open and non-deterministic environments with complex interactions, and the need to dynamically adapt to new requirements, technologies or environmental conditions without redeployment and without interruption of the system’s functionality. Conventional development approaches and languages do not provide adequate support for the problems posed by this challenge.

The goal of the ASCENS project is to develop a coherent, integrated set of methods and tools to build software for ensembles. To this end we research foundational issues that arise during the development of these kinds of systems, and we build mathematical models that address them. Based on these theories we design a family of languages for engineering ensembles, formal methods that can handle the size, complexity and adaptivity required by ensembles, and software-development methods that provide guidance for developers.

In this lecture I presented a systematic approach for engineering ensembles including an ensemble engineering process, the SOTA approach to ensemble requirements, the underlying formal model called GEM, the SCEL language for designing ensembles, and techniques for the quantitative analysis of ensembles.

3.20 Smart Cities as Heterogeneous Superorganisms: Scenarios and Challenges

Franco Zambonelli (University of Modena, IT)

License © Creative Commons BY 3.0 Unported license
© Franco Zambonelli

The smartness of future cities will be in their capabilities of working like immense heterogeneous superorganisms, bringing together a very diverse actors, from humans, to robots, to ICT devices of any kind.

Engineering the behavior of such systems for the good of our society as a whole and for the good of each individuals in it will be the key societal challenge of the future, and a source for fascinating research challenges in the area of computer science and collective adaptive systems.

4 Working Group Reports

4.1 Modelling, Specification, and Programming for Collective Adaptive Systems

Hella Seebach, Lenz Belzner, Marco Gribaudo, Anabelle Klarl, Michele Loreti, Ugo Montanari, Laura Nenzi, Rocco De Nicola, Christophe Scholliers, Petr Tuma, and Martin Wirsing

License © Creative Commons BY 3.0 Unported license
 © Hella Seebach, Lenz Belzner, Marco Gribaudo, Anabelle Klarl, Michele Loreti, Ugo Montanari, Laura Nenzi, Rocco De Nicola, Christophe Scholliers, Petr Tuma, and Martin Wirsing

4.1.1 Introduction

Over the last decades we have witnessed a steep growth in the world population. This increase has a vast impact in the large scale on how cities operate. For example, how to route traffic in the city and where to place parking spots in such a way that the individuals commuting time is minimized. On a smaller scale, big events such as festivals have to be able to predict how the crowd will react in case of a major incident. It is needless to say that each of the individuals at a festival are autonomous entities, yet it is surprising to see that certain patterns can be observed from the group as a whole. Systems consisting out of a large number of individuals exhibiting group behaviour are called collective adaptive systems (CAS). While the collective adaptive systems described above consist solely out of humans the idea is that these systems can consist both out of human entities and/or ICT components.

While our understanding of CAS is getting better over time, the field is not widely understood by the big audience. CAS are omnipresent in current society and it is thus essential to be able to provide the correct set of abstraction in order to model, verify and implement them.

In this paper we show four typical domains of CAS (Sec. 4.1.2). Afterwards we give a description of what collective adaptive system are and how they can be characterized (Sec. 4.1.3). Section 4.1.4 shows how CAS can be modelled and implemented on a computer system. From this overview we conclude that each of the non-trivial collective adaptive systems has a vast need to reason over spatio-temporal properties. Surprisingly, most of the modelling and implementation techniques, do not provide spatio-temporal operations as first class entities. This means that programmers must encode these properties themselves which is time consuming and prone to error. We thus argue that in order to better reason about collective adaptive systems it is essential to focus on these operations. We conclude this paper with perspectives on future research and propose a set of challenges for future researchers to tackle elegantly.

4.1.2 Application Domains

Collective adaptive systems can be found in a lot of different domains. Each application domain naturally leads to different characteristics of CAS which will need multiple new enabling technologies. We just discussed four domains in this workshop which perfectly fit for developing and evaluating techniques for CAS.

Power Management Systems

In current power management systems, big power plants are controlled by electric utilities and other organisations in a flat hierarchy. Utilities and companies manage parts of the

overall power system independently from each other. For each of the big power plants, a schedule is created that postulates the output of the power plant at a given time. Schedules are coarse-grained, providing target values in 15 minute intervals. Small power plants and especially DERs (distributed energy resources) under the control of small cooperatives or individuals produce without external control and feed the power produced into the grid. This lack of control by the electric utilities is compensated by powerful controllable power plants. Current plans are to scale the controllable output further by installing more plants, especially flexible gas-powered ones. Geographical distribution is even increasing with the wide-spread installation of DERs such as biogas plants, solar plants, and wind farms. The relative stability in the network is an emergent behaviour. No single entity of the system can provide this stability in the face of load fluctuations, weather changes, and generator outages.

What makes power systems difficult to manage and optimize is the cost of storing energy: since energy consumption varies remarkably with time (day, week and season), the unit cost of production varies also, because less efficient plants are turned on only at peak time. On the other hand, DERs production capacity is also heavily dependent on weather conditions, and thus quite variable in time. Physical storage systems (hydro pumping stations, batteries, e.g. of electric vehicles) are not very practical, thus the best policy is to try to match consumption and production in an integrated or unbundled market. The goal is to make demand active, trying to move it, whenever possible, to slots where energy is less expensive. Therefore, there is a need for a future power grid in which even small power plants, consumers, as well as prosumers (entities that produce and consume power like an electric vehicle) can be controlled or participate in a scheduling scheme or market, since entrance requirements to power markets, such as the lower limit of 100 kW for contracts at the European Energy Exchange (EEX), exclude access for small organisations. Networked measuring equipment must be equipped to allow observing the grid status and make decisions based on current conditions. Power plants and consumers will be networked, too, and provide future production or consumption. Producers, consumers, and prosumers must be combined into groups, e.g., as aggregators [1] or Autonomous Virtual Power Plants (AVPPs) [2],[3],[4] that create schedules to cover a portion of the load (depending on the demand) aiming at: (i) lowering peak energy consumption by exploiting their flexibility; (ii) reducing electricity cost for the whole population of actors; (iii) increasing robustness by locally dealing with load and output fluctuations; and (iv) making a profit. Remaining research challenges comprise the robust autonomous scheduling of large-scale open heterogeneous systems (including spatial and temporal constraints) as well as security, privacy, and safety aspects, amongst others.

Cloud Computing

Contemporary cloud computing platforms rely on server farms that host a number of dedicated workload processing servers together with the necessary networking and storage infrastructure. The infrastructure does not expose the details of virtual server location at the level of individual server racks or individual network ports – these are managed transparently by the platform provider, possibly using mechanisms such as virtual machine migration [5] or software defined networks [6]. In contrast, higher granularity location is exposed – complex application deployment scenarios use it to make sure that both application code and application data is distributed appropriately to accommodate the (often conflicting) requirements on communication efficiency, failure resiliency, cost and other factors.

Although many cloud computing applications adopt somewhat conservative resource management techniques (such as limiting dynamic server allocation to manually selected server farms), many research directions seek to amplify the existing cloud computing benefits

by introducing mechanisms such as cloud federations or ad hoc and opportunistic clouds [7, 8].

On many levels, these directions strengthen the collective adaptive system characteristics of the cloud. There are multiple focus areas for research in the domain of cloud computing. *Efficient resource allocation*: Both the cloud platform and the cloud applications seek to maximize utility and minimize cost by sharing resources (servers, storage, network). Efficient resource allocation is a collective task where multiple adaptive entities (platform components and application components) allocate and release resources to meet their specific requirements. The domain offers research challenges in both cooperative and competitive resource allocation algorithms in presence of changing requirements [9, 10, 11, 12, 13, 14, 15]. In turn, these contain challenges in monitoring and predicting the impact of particular resource allocation, needed to perform allocation decisions. *Robustness against failures*: Especially in an open cloud with voluntary participation, node failures and application failures are expected rather than exceptional situations. The domain requires research into efficient failure resilient algorithms, behaviour modelling in presence of (possibly dependent) failures and other challenging problems [16]. *Security against abuse*: As an open environment with heavy resource sharing, cloud computing exposes many opportunities for abuse. These include not only the more traditional security related issues (virtual machine hijacking, data theft and other), but also the possibility of using the available resources beyond fair share or outright free-loading. There is a need for strategies and mechanisms that would prevent such abuse [17, 18, 19, 20]. *Preventing negative emergence*: The cloud environment incorporates entities whose behaviour is largely automated but rarely fully disclosed or even fully understood. Such an environment is easily prone to emergent behaviour with negative consequences, for example oscillations in the adaptive feedback mechanisms. The domain can benefit from research into preventing, detecting or managing cases of negative emergence.

As another practical benefit of the cloud computing domain, the difficulty of the identified challenges varies with the degree of openness and heterogeneity that is considered – a centralized resource allocation in a closed homogeneous cloud faces different issues than a cooperative distributed resource allocation in an open heterogeneous cloud with voluntary participation. Although multiple projects already started tackling some of the listed challenges [21, 22, 23, 24, 25, 26, 27, 28], there are still many topics waiting for advances.

Telecommunication – LTE Resource Allocation

LTE technology, used in 4G mobile phone communications, employs a channel resource allocation scheme based on orthogonal frequency-division multiplexing. In particular it supports for both time-division multiplexing and frequency-division multiplexing, splitting the communication spectrum in a set of resource blocks.

Each participating device is equipped by multiple antennas, and can transmit on more frequencies at the same time. The total bandwidth available to a device depends on the number of blocks it can use to transmit/receive data. The LTE technology envisage the possibility of dynamically allocating the resource blocks depending on the actual demand, to improve the performances that can be achieved by the single user. The service provider usually operates block allocation in a centralized manner: this however limits the bandwidth that could be achieved since interference might arise from the carrier being shared by different providers. An autonomous solution, where each mobile agent can acquire and release block resources, could improve the available bandwidth overcoming these difficulties. This however is not an easy task due to the characteristics of the wireless medium that is affected by limitations such as the hidden terminal problem [29]. The problem can become even more interesting when the cellular infrastructure is complemented with alternative wireless access

	Homogeneous Heterogeneous	Collaborative Competitive	Low ind. impact High ind. impact				
Cloud	<-----X	-----	-----	Space	Sync / async	Continuous / discrete	Open / closed
Wristband	X	·	·				
4G LTE	<-X	·	-----				
Power grids	<---X	-----	-----				

■ **Figure 1** Categorization of the considered applications.

technologies such as WiFi hotspots [30]. In this way data traffic can be offloaded whenever possible towards such hotspots, at the price of a possible degradation in the quality of service experienced by the users [31, 32]. Preliminary studies of such systems using CAS-based techniques have been proposed in [33, 34]. The key solutions in this direction will also be the basis to the next generation of wireless and cellular communications that exploits more advanced techniques such as *cognitive radio* [35] in which terminals and base stations harvest for unused radio frequencies and spectrum bandwidths to increase their transmission capacity.

Wearable Computational Devices

Due to the advances in hardware technology and the miniaturisation of electronic components it has become feasible to make wearable computational devices. These wearable devices open up opportunities for new and exciting applications. Also in the world of collective adaptive systems this new technology can be exploited. One particular application is the use of wristbands equipped with wireless near field communication. When a large number of people are equipped with such wristbands these wristbands could light up in order to provide additional functionalities to the users. One application of these wristbands could be to make figures at large scale events by lighting up the wristbands at synchronised moments in time. Additionally the same wristbands could be used at mass events to drive people to the exit in case of disaster. Challenging is amongst others the situation of simple, limited nodes (simple communication, limited resources) and the unpredictable position of the nodes.

4.1.3 Common Features and Characteristics of Application Domains

In the workshop, we identified multiple characteristics for CAS. The four mentioned application domains can be categorized against a set of different features that will guide in the selection of the most appropriate modelling technique. Figure 1 summarizes the results.

The first aspect we considered is the type of elements that compose the CAS. The elements can be *homogeneous*: all the cooperating entities can be considered to be identical. This is for example the case of the bracelets in the wristband application, where all the devices are exactly the same. *Heterogeneous* applications are instead composed of agents that are completely different one from the other. A typical example is the power grid scenario, where each producer or consumer is completely different from the others, of course depending on the level of abstraction. Both the cloud and the LTE scenario have some degree of heterogeneity due to the fact that they are composed by different devices, produced by different manufacturers. However all the devices are abstracted by the *role* they are playing in the application: this allows us to consider them homogeneous from a modelling perspective.

The second aspect we discussed is whether the agents are *collaborative* or *competitive*. For

example, the wristband application is a clear example of a collaborative system: each agent cannot gain any advantage by not cooperating with the others. The LTE network is instead an example of a competitive application, where each mobile device tries to acquire all the available bandwidth to improve its communication performance. Cloud computing can be either collaborative or competitive depending on the specific application we are considering. A Big Data application might be competitive to gain more resource to parallelize its execution and to reduce its running time. A Platform-as-a-service job can instead be consolidated with other applications to increase the chance of having idle machines that can be switched off, reducing the total energy consumption. The prosumer in the power management systems are first and foremost competitive to optimize their benefit. But in future energy grid scenarios they have to collaborate in organisational structures to be able to participate for example on the energy market.

The third feature is the impact that a local agent can have on the entire community. In the wristband application it can be minimal, since an agent can at most do not propagate a message and do not properly switch the colour of the bracelet. In the power grid example the impact is instead maximum, since other nodes might be relying on the production or consumption of energy of other participants in the network. 4G LTE might have a limited impact, since most of devices are autonomous. However, the presence of a shared environment (the communication spectrum) can have an impact on a large set of devices in case of malicious signals that could be generated to interfere with the regular transmissions. The particular characterization of the cloud-computing scenario depends on the considered application since it can have either a low impact (as the exclusion of a physical node in an Infrastructure-as-a-service scenario), or a high impact (for example when the shut down of a node in a distributed storage application makes a file no longer accessible).

Other important features that characterize an application from a modelling point of view are the presence of space, the fact of being synchronous or asynchronous, discrete or continuous, open or closed. All the examples that we consider in this work rely somehow on a concept of *space*. Space can be either a physical space or a logical space. For example, both the stadium where the concert is held in the wristband application, or the area where base stations are located in the LTE application, are examples of physical spaces. The nodes interconnected by the distribution network in the power grid application, and the servers and routers in the cloud application, are examples of logical space. In both cases the system topology can be described by a graph, where nodes of the system correspond to nodes on the graph. All the considered applications are *asynchronous*, but they build up some level of *synchronism* to allow the system to reach their goals: depending on the level of abstraction that we want to consider, different modelling techniques can be used to specifically target their synchronous or asynchronous features. Most of the applications are *discrete*, in the sense that they change their state according to events that happens in discrete time instants. The power grid application however, requires a *continuous* time approach since problems can arise and must be handled in few milliseconds. This leads to the requirement of radically different modelling techniques. Finally applications can be considered either *open* or *closed*. The wristband is a classical example of a closed application. Also in this case, depending on the level of abstraction and on the features we are interested to consider, different modelling techniques could be employed.

4.1.4 Methods to approach CAS

We identified three different main approaches to model CAS: systems of systems, autonomy, and aggregation. They are not exclusive; rather they should be regarded as a kind of

dimensions that a particular system or solution exhibits. Afterwards we discussed several modelling techniques for the different levels of abstraction.

Systems of Systems (Roles)

A core characteristic of CAS to be modelled are behavioural and communicational aspects, both on the individual and the collective level. A key challenge here is the specification of organisational, communicational or behavioural collectives. Also, modelling these structures has to take into account reconfiguration of collectives at runtime due to changing situations (i.e. adaptation). The properties stated above lead to the idea of considering systems of systems [36] as an appropriate way of modelling CAS. The collectives may be organized in hierarchical or overlapping ways, also depending on spatial aspects. In the workshop, aggregation and organization based on roles, communication patterns and spatio-temporal properties have been discussed.

Autonomy (Reasoning)

One way to drive adaptation is to provide learning and planning capabilities to individuals and collectives alike. Reasoning and planning provide ways for autonomic system reorganization according to current needs and system goals. In the context of CAS, this gives rise to questions about individual and collective knowledge gathering and transformation as well reasoning capabilities. In especial, if considering systems of systems the question arise how to compare and evaluate systems e.g. on different or equal levels of hierarchies [37] or in different locations leading to different circumstances.

Aggregation (Quantification)

CAS may consist of extreme numbers of individuals. Also, these numbers may be unknown at design-time and/or runtime. Quantitative analysis approaches identify and abstract away symmetries and structure of the system in order to allow for efficient computation of system properties. While this scalability is highly desirable, it comes at the cost of specializing towards a particular problem or situation – quantitative approaches are strongly coupled to the way a CAS is modelled. Thus, they should drive modelling approaches as well as respect any abstractions made when modelling CAS. Most of the techniques in this field rely on mean field solutions [38, 39]: the system is studied by considering variables that counts the number of elements in the same state, and by studying their evolution using a set of ordinary differential equations. The mean field approximation basically states that a large number of objects that randomly evolve tend to show a deterministic mean behaviour as their count tends to infinity. On this assumption, many higher level modelling techniques based on process algebra [40, 41], or Markovian agents [42] have been developed.

Each of the approaches already provides solutions for problems studied under particular aspects. What remains a mostly open challenge is the combination of modelling and solutions from the different perspectives as well as the integration of spatio-temporal aspects in the mentioned techniques. For example, modelling and collective organization formalisms have to (a) provide methods for integration of reasoning in the modelling process and (b) allow for autonomous, goal- or situation-based reconfiguration. On the other hand, reasoning has to account for structural changes, and has to infer about the collective structure. Also, it seems an interesting challenge how different quantitative approaches could be instrumented autonomously based on current system configuration and accounting for autonomous reconfiguration at runtime.

Modelling CAS at different level of abstractions

Different languages have been proposed or used to support modelling, analysis and deployment of CAS. Some of these are general purpose languages, like Java, that, while providing appropriate API, can be used to program and deploy systems also on complex distributed infrastructures. Others languages are domain specific [43, 44, 45, 46, 47] and are equipped with syntactic constructs specifically thought for modelling relevant aspects of CAS. These domain specific languages are typically more oriented to specification than to deployment and provide formal and automatic tools that can be used to support analysis.

This variety of tools and languages can be used by a CAS designer to tackle the system modelling at different level of abstractions. Indeed, each language can be used to describe and analyse a system from different perspective. However, to take a real advantage from this plethora of tools formal links between the considered formalism are definitively needed. The links, that can be rendered in terms of *model transformation*, will be first of all used to relate the different models described in the different languages. These relations will be then instrumental to use the results of analysis performed in a given language to update/improve the other models.

4.1.5 Conclusion and Open Challenges

After we discussed the different application domains for CAS and considered suitable modelling and specification techniques we identified a set of challenges that must be tackled before CAS will be integrated in today's or future ICT systems. As one main result the working group came to the conclusion that spatio-temporal properties are of great value for the modelling and implementation of CAS but are not yet appropriately integrated in the available methods. One concrete challenge: Investigation of the design of CAS learning spatio-temporal requirements. The idea is to optimise the specification and implementation of the space features in the model in such a way that the satisfiability of a spatio-temporal property is maximised. As we know, the verification of global properties on CAS is often an intractable task from a computational point of view. For this reason, such properties will have to be decomposed in a set of local requirements in the optimisation process.

Further the participants of CAS need methods to reason about local versus global or even conflicting goals of the system. These decisions strongly depend on the organisational structures and the presence or absence of central institutions in the CAS. One concrete challenge: When describing/implementing CAS two aspects are crucial: The specification of the actual actions that the different components have to perform (behavioural specification) and the specification of the goal that the single components or the collectives have to achieve (goal specification). Usually these two kinds of activities are performed by taking advantage of very different tools, the former are performed with classical imperative programming language while the latter rely on declarative specifications. The foreseen challenge is the reconciliation of these two approaches to, e.g., able to take decisions about the next actions after having measured how far the goal is and what is the best choice to get closer to it.

Quantitative approaches for modelling and analysis of CAS help to meet the challenge of state space explosion if considering large-scale CAS. Beside the mean field solutions outlined in the previous section, new physically inspired techniques could be applied. One example could come from fluid dynamics, leading to *fluid approximations to CAS modelling*. If we consider a very large number of agents, densely packed, their evolution can be approximated as the motion of a fluid. To give an idea, let us imagine that agents can evolve through a finite set of modes $i \in \{1, \dots, M\}$. Let us also focus on two-dimensional space where agents

can evolve. The state of whole system at time t can be characterized by a set of functions $p_i(x, y, t)$ that describes the density of agents (measured in agents per unit area) in state i at position (x, y) . The evolution of $p_i(x, y, t)$ can be described by a set of partial differential equations, similar to the one used by the mass continuity law in fluid dynamics. From the state density $p_i(x, y, t)$ several performance metrics can be derived. These measures can be used to assess several properties, such as for example determining if an emergent behaviour can appear, and which could be the convergence rate as function of the parameters of the model.

References

- 1 Peeters, E., Belhomme, R., Batlle, C., Bouffard, F., Karkkainen, S., Six, D., Hommelberg, M.: ADDRESS: scenarios and architecture for active demand development in the smart grids of the future. In: Electricity Distribution-Part 1, 2009. CIRED 2009. 20th International Conference and Exhibition on, IET (2009) 1–4
- 2 Steghöfer, J.P., Anders, G., Siefert, F., Reif, W.: A system of systems approach to the evolutionary transformation of power management systems. In: GI-Jahrestagung. (2013) 1500–1515
- 3 Bremer, J., Rapp, B., Sonnenschein, M.: Encoding distributed search spaces for virtual power plants. In: Computational Intelligence Applications In Smart Grid (CIASG), 2011 IEEE Symposium on. (April 2011) 1–8
- 4 Becker, B., Allerdig, F., Reiner, U., Kahl, M., Richter, U., Pathmaperuma, D., Schmeck, H., Leibfried, T.: Decentralized energy-management to control smart-home architectures. In Müller-Schloer, C., Karl, W., Yehia, S., eds.: Architecture of Computing Systems – ARCS 2010. Volume 5974 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2010) 150–161
- 5 Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: Proceedings of NSDI 2005, USENIX
- 6 Casado, M., Koponen, T., Ramanathan, R., Shenker, S.: Virtualizing the network forwarding plane. In: Proceedings of PRESTO 2010, ACM
- 7 Grozev, N., Buyya, R.: Inter-cloud architectures and application brokering: Taxonomy and survey. Software: Practice and Experience (2014)
- 8 Kirby, G., Dearle, A., Macdonald, A., Fernandes, A.: An approach to ad hoc cloud computing
- 9 Warneke, D., Kao, O.: Exploiting dynamic resource allocation for efficient parallel data processing in the cloud. IEEE Transactions on Parallel and Distributed Systems **22**(6) (2011)
- 10 Ishakian, V., Sweha, R., Bestavros, A., Appavoo, J.: CloudPack. In: Proceedings of MIDDLEWARE 2012, Springer
- 11 Wang, J., Hua, R., Zhu, Y., Wan, J., Xie, C., Chen, Y.: RO-BURST: A robust virtualization cost model for workload consolidation over clouds. In: Proceedings of CCGRID 2012, IEEE
- 12 Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Journal of Future Generation Computing Systems **28**(5) (2012)
- 13 Li, J., Shuang, K., Su, S., Huang, Q., Xu, P., Cheng, X., Wang, J.: Reducing operational costs through consolidation with resource prediction in the cloud. In: Proceedings of CCGRID 2012, IEEE
- 14 Dong, J., Jin, X., Wang, H., Li, Y., Zhang, P., Cheng, S.: Energy-saving virtual machine placement in cloud data centers. In: Proceedings of CCGRID 2013, IEEE

- 15 Raghavendra, R., Ranganathan, P., Talwar, V., Wang, Z., Zhu, X.: No “power” struggles: Coordinated multi-level power management for the data center. In: Proceedings of ASPLOS 2008, ACM
- 16 Khalifa, A., Azab, M., Eltoweissy, M.: Resilient hybrid mobile ad-hoc cloud over collaborating heterogeneous nodes. In: Proceedings of COLLABORATECOM 2014, IEEE
- 17 Velloso, P.B., Laufer, R.P., de O. Cunha, D., Duarte, O.C.M.B., Pujolle, G.: Trust management in mobile ad-hoc networks using a scalable maturity-based model. *IEEE Transactions on Network and Service Management* **7**(3) (2010)
- 18 Huang, D., Zhang, X., Kang, M., Luo, J.: MobiCloud: Building secure cloud framework for mobile computing and communication. In: Proceedings of SOSE 2010, IEEE
- 19 He, Q., Wu, D., Khosla, P.: SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks. In: Proceedings of WCNC 2004, IEEE
- 20 Buchegger, S., Boudec, J.Y.L.: Self-policing mobile ad-hoc networks by reputation systems. *IEEE Communications* **43**(7) (2005)
- 21 Bernard, Y., Klejnowski, L., Hähner, J., Müller-Schloer, C.: Towards trust in desktop grid systems. In: 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, CCGrid 2010, 17-20 May 2010, Melbourne, Victoria, Australia. (2010) 637–642
- 22 Bulej, L., Bureš, T., Horký, V., Keznlk, J.: Adaptive deployment in ad-hoc systems using emergent component ensembles: Vision paper. In: Proceedings of ICPE 2013, ACM
- 23 Mayer, P., Velasco, J., Hennicker, R., Puviani, M., Tiezzi, F., Pugliese, R., Keznlk, J., Bureš, T. In: *The Autonomic Cloud*. Springer (2015)
- 24 Amoretti, M., Grazioli, A., Senni, V., Zanichelli, F.: Towards a formal approach to mobile cloud computing. In: Proceedings of EUROMICRO PDP 2014
- 25 Sebastio, S., Amoretti, M., Lluch-Lafuente, A.: A computational field framework for collaborative task execution in volunteer clouds. In: Proceedings of SEAMS 2014, ACM
- 26 Celestini, A., Lluch-Lafuente, A., Mayer, P., Sebastio, S., Tiezzi, F.: Reputation-based cooperation in the clouds. In: Proceedings of IFIP TM 2014
- 27 Vassev, E., Hinchey, M., Mayer, P.: Formalizing self-adaptive clouds with knowlang. In: Proceedings of ISoLA 2014
- 28 Klarl, A., Mayer, P., Hennicker, R.: HELENA@Work: Modeling the science cloud platform. In: Proceedings of ISoLA 2014
- 29 Tsertou, A., Laurenson, D.I.: Revisiting the hidden terminal problem in a csma/ca wireless network. *Mobile Computing, IEEE Transactions on* **7**(7) (2008) 817–831
- 30 Trullols, O., Fiore, M., Casetti, C., Chiasserini, C., Ordinas, J.B.: Planning roadside infrastructure for information dissemination in intelligent transportation systems. *Computer Communications* **33**(4) (2010) 432–442
- 31 Han, B., Hui, P., Kumar, V.A., Marathe, M.V., Pei, G., Srinivasan, A.: Cellular traffic offloading through opportunistic communications: A case study. In: Proceedings of the 5th ACM Workshop on Challenged Networks. CHANTS '10, New York, NY, USA, ACM (2010) 31–38
- 32 Han, B., Hui, P., Kumar, V.S.A., Marathe, M.V., Shao, J., Srinivasan, A.: Mobile data offloading through opportunistic communications and social participation. *IEEE Transactions on Mobile Computing* **11**(5) (May 2012) 821–834
- 33 Gribaudo, M., Manini, D., Chiasserini, C.: Studying mobile internet technologies with agent based mean-field models. In: *Analytical and Stochastic Modelling Techniques and Applications – 20th International Conference, ASMTA 2013, Ghent, Belgium, July 8–10, 2013*. Proceedings. (2013) 112–126
- 34 Chiasserini, C., Gribaudo, M., Manini, D.: Traffic offloading/onloading in multi-rat cellular networks. In: Proceedings of the IFIP Wireless Days, WD 2013, Valencia, Spain, November 13-15, 2013. (2013) 1–7

- 35 Mitola, J., Maguire, G.Q., J.: Cognitive radio: making software radios more personal. *Personal Communications, IEEE* **6**(4) (Aug 1999) 13–18
- 36 Sage, A.P., Cuppan, C.D.: On the systems engineering and management of systems of systems and federations of systems. *Inf. Knowl. Syst. Manag.* **2**(4) (December 2001) 325–345
- 37 Schiendorfer, A., Steghöfer, J.P., Reif, W.: Synthesis and abstraction of constraint models for hierarchical resource allocation problems. In: *Proc. of the 6th International Conference on Agents and Artificial Intelligence (ICAART)*. Volume 2.
- 38 Benaim, M., Boudec, J.L.: A class of mean field interaction models for computer and communication systems. *Perform. Eval.* **65**(11-12) (2008) 823–838
- 39 Bobbio, A., Gribaudo, M., Telek, M.: Analysis of large scale interacting systems by mean field method. In: *Fifth International Conference on the Quantitative Evaluation of Systems (QEST 2008)*, 14-17 September 2008, Saint-Malo, France. (2008) 215–224
- 40 Bortolussi, L., Hillston, J., Latella, D., Massink, M.: Continuous approximation of collective system behaviour: A tutorial. *Perform. Eval.* **70**(5) (2013) 317–349
- 41 Hayden, R.A., Stefanek, A., Bradley, J.T.: Fluid computation of passage-time distributions in large markov models. *Theor. Comput. Sci.* **413**(1) (2012) 106–141
- 42 Cerotti, D., Gribaudo, M., Bobbio, A.: Markovian agents models for wireless sensor networks deployed in environmental protection. *Rel. Eng. & Sys. Safety* **130** (2014) 149–158
- 43 De Nicola, R., Loreti, M., Pugliese, R., Tiezzi, F.: A formal approach to autonomic systems programming: The SCEL language. *TAAS* **9**(2) (2014) 7
- 44 Feng, C., Hillston, J.: PALOMA: A process algebra for located markovian agents. In Norman, G., Sanders, W.H., eds.: *Quantitative Evaluation of Systems – 11th International Conference, QEST 2014, Florence, Italy, September 8–10, 2014. Proceedings*. Volume 8657 of *Lecture Notes in Computer Science.*, Springer (2014) 265–280
- 45 Alrahman, Y.A., Nicola, R.D., Loreti, M., Tiezzi, F., Vigo, R.: A calculus for attribute-based communication. In: *Proceedings of the ACM Symposium on Applied Computing, SAC 2015, ACM* (2015) To appear.
- 46 Bortolussi, L., Nicola, R.D., Galpin, V., Gilmore, S., Hillston, J., Latella, D., Loreti, M., Massink, M.: Caspa: a collective adaptive stochastic process algebra. In Bertrand, N., Tribastone, M., eds.: *Proceedings Twelfth International Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2015, April 2015. EPTCS* (2014) To appear.
- 47 Viroli, M., Damiani, F.: A calculus of self-stabilising computational fields. In Eva Kühn, Pugliese, R., eds.: *Coordination Models and Languages – 16th IFIP WG 6.1 International Conference, COORDINATION 2014, Held as Part of the 9th International Federated Conferences on Distributed Computing Techniques, DisCoTec 2014, Berlin, Germany, June 3–5, 2014, Proceedings*. Volume 8459 of *Lecture Notes in Computer Science.*, Springer (2014) 163–178

4.2 Verification of CAS

Luca Bortolussi, Giacomo Cabri, Giovanna Di Marzo Serugendo, Vashti Galpin, Jane Hillston, Roberta Lanciani, Mieke Massink, Mirco Tribastone, and Danny Weyns

License © Creative Commons BY 3.0 Unported license

© Luca Bortolussi, Giacomo Cabri, Giovanna Di Marzo Serugendo, Vashti Galpin, Jane Hillston, Roberta Lanciani, Mieke Massink, Mirco Tribastone, and Danny Weyns

Verification is the process of assessing how well a system meets a specification or requirement. A variety of approaches have appeared in the literature, ranging from model checking to

static analysis of source code and theorem proving. In this working group, we primarily focused on verification based on model checking [2, 13], in which a state-based description of the system is assessed with respect to a property expressed in an appropriate specification language, like a temporal logic. In particular, we considered the challenges that arise in the model checking of Complex Adaptive Systems (CAS), which are systems comprised of a large number of heterogeneous agents which, interacting together, produce a complex array of collective behaviours.

In our working group in Dagstuhl, we first identified the major issues and more interesting challenges that arise in the process of verification of CAS. Afterwards, we divided in two subgroups to discuss in detail specific topics related to this general framework. In particular, we chose to investigate: (1) the quantitative or stochastic model checking in the presence of uncertainty, and (2) the specifications and logics which capture the spatial arrangements of systems, characterising the impact of those arrangements on collective behaviour. A brief account of each subgroup is given below.

4.2.1 Introduction

In our discussion, we first identified the numerous important issues and challenges that arise when we want to verify a CAS. In the following, we outline just a few of them.

Adaptation and verification

Adaptation is itself an interesting phenomenon which could be subjected to verification. In particular, we considered the issue of quantifying how adaptive the system is, and we identified a number of different measures that can be used to validate the adaptation of a CAS. For example:

Speed of adaptation – Once an adaptation is initiated, how rapidly does the system enter a stable behaviour? In this requirement the stable behaviour is not necessarily a single state, but could be a new stable equilibrium with a given probability distribution over a set of states.

Robustness – How often does the system adapt? Is there a danger of “thrashing”, meaning that the system alternates between different adaptations, achieving one, and shortly after pursuing another?

Effectiveness of adaptation – How closely does a system satisfy the revised property or goal after an adaptation?

Verifying global properties based on local behaviours

In many cases, the properties that are of interest to both system developers and system users are global requirements related to emergent behaviours of CAS. But the populations of CAS are comprised of a large number of single entities, whose local behaviour is usually more accessible and intuitive than the collective description of the system (due to the way CAS are implemented). Hence, there is a need for compositional approaches that are able to validate global properties of CAS, building on the verification of local requirements related to single agents or group of individuals. First promising results in this respect were achieved in [6] in the context of quantitative model checking of population models.

Verification in the presence of uncertainty

A characteristic feature of CAS is the uncertainty. For example, the structure of part of the system may be totally unknown or unknown at a particular instant in time. Moreover, the goals and objectives of a single agent may be hidden from the others, possibly due to an ongoing adaptation process. At a finer level of detail, the rates or probabilities that govern the dynamic behaviour of the system may be unknown, or changing in undefined ways, meaning that model of the CAS could be underspecified.

Scalability

Many verification techniques rely upon explicit representation of the state space of the model. In a CAS this is challenging in two respects. Firstly, not all possible states may be known due to future adaptation, as discussed above. Secondly, even if the “complete” state space is known or can be anticipated, the model will typically include too many states to be represented explicitly. Alternatives such as statistical model checking avoid constructing the whole state space at once, but then become computationally very expensive due to the sampling approach that must be adopted, necessitating many simulation runs before a verification can be approximated. An alternative is to use techniques based on fluid or mean field representation of the discrete state space [20, 4, 5, 6], but these approaches are still in their infancy.

Openness

Openness is an inherent property of CAS, as agents may join or leave the system throughout its lifetime. This poses severe challenges for state-based modelling techniques, particularly if there is the possibility that the population of the system grows unboundedly. In these scenarios, care is needed in phrasing the properties to be satisfied by the system. For example, it may be more appropriate to express goals in terms of proportions of agents rather than absolute numbers.

Quantified verification as a driver for adaptation

When the models contain quantitative information about the system, such as information about the timing and likelihood of events, it is possible to assess a system against a property not just in terms of boolean satisfaction but in a more quantified way. This can be viewed as measuring the degree to which a system satisfies a property, or the distance that a system is from satisfying the property. When this form of quantification is added to the verification process it is possible to see how verification can become a driver for adaptation. As the system undergoes adaptation, its progress towards a goal can be explicitly measured.

In recent years several authors have considered quantitative satisfaction of properties. In this framework, when a system is assessed against a property the result is a measure of distance indicating how close the system comes to satisfying the property [23, 21, 18]. For example, if the property is satisfied then the distance is zero. In this framework a system which fails to satisfy a property at distance 0.2 may be considered preferable to a system which fails to satisfy the property at distance 0.8. This approach has been used to conduct sensitivity analysis of model parameters with respect to desirable properties [24] and to seek parameters that bring a model closest to property satisfaction [25, 1]. This latter approach could be deployed to drive adaptation through verification.

Spatial aspects

In many CAS the location of agents, and bounded ranges of communication are an important factor in the design and realisation of the system. Thus it is essential that location and movement are treated as primitives in both modelling and verification techniques developed to support CAS. Often in existing techniques, if handled at all, space is treated only logically, and the relationships between locations are qualitative rather than quantitative. Thus a model may capture that locations are in some sense “adjacent” but not the actual distance between them. However, if agents are operating, for example, in a wireless sensor network, the actual distance between them will determine whether or not they are able to communicate, or the energy cost of doing so. Incorporating detailed spatial information means that model checking must consider spatio-temporal properties, a further level of challenge.

4.2.2 Motivation

To motivate our discussions in the working group we considered possible applications, where some of the discussed issues would have practical relevance. In particular, we identified the following:

Global adaptation: In this application, the adaptation takes place in the environment, while the agents operating within the system keep the same behaviour. An example of this would be a smart grid, where differential pricing is used to stimulate a change in the behaviour of the end users. In this scenario, a collective change in the dynamics of the system is obtained by acting on the environment (lowering the price of the energy during less busy period of the day), while the agents (the end users) keep the same goals and objectives (to buy energy at the cheapest possible price).

Agent adaptation: In these scenarios the agents change their behaviour based on local information, generating an effect on the collective behaviour of the system. An example of this is a peer-to-peer file sharing systems such as BitTorrent [14, 22]. In this application, the end users locally adapt to improve their own quality of service, while the environment, the BitTorrent protocol, remains unchanged. Moreover, the choices made by the single user affect its rate of uploading content to the network, thus altering the behaviour of the whole network.

4.2.3 Subgroup I: Uncertainty in CAS

When there is uncertainty in the behaviour of the system under consideration it makes the task of verifying a system even more challenging. Unfortunately in CAS, the inherent adaptability means that uncertainty is also inherent in the system. Broadly speaking, we identified two distinct approaches:

- Offline verification, before the system is deployed, tries to anticipate the possible range of behaviours that can be encountered.
 - Online verification, conducted while the system is running, reflects the observed behaviour.
- We anticipate that in many CAS both approaches may be needed.

Offline verification

In this case we might assume that a model is available which aims to capture all the possible behaviours, but that some aspect of the model, in particular the parameters corresponding to any given time or mode of behaviour, are unknown. To some extent this is the aim of probabilistic models which treat the parameters of models as random variations, with a

defined distribution function which gives an estimate of the range of possible values and the likelihood of each. Such approaches have long been used in performance and dependability modelling to abstract away from concrete instances of behaviour and capture instead a range of possible behaviours in a statistically meaningful way. However, the uncertainty that we are considering here means that even the random variables characterising the range of behaviours may be unknown, or unknowable. This arises because we are often interested in designing systems satisfying emergent properties. Uncertainty can emerge in many ways; probably one of the simplest ways is to consider models that are structurally defined, but which can have unspecified parameter values p , possibly belonging to a bounded set P . Furthermore, we assume that we are dealing with stochastic models, so that behaviours are satisfied with a certain probability, rather than always or never.

The question of how we can verify properties under such uncertainty is a difficult one. One approach is to compute the probability of satisfaction of a certain property ϕ (for instance, encoded as a linear temporal logic formula) as a function of the parameter values $p \in P$. Let us denote this satisfaction function by $f(p)$. An exhaustive numerical computation (or bounding) of $f(p)$ for $p \in P$ is infeasible even for simple models. Recently, a statistical method was proposed by Bortolussi *et al.* [7], leveraging machine learning ideas to characterise statistically the function f . This approach can be complemented with that of [3], where the authors use a similar statistical scheme to evaluate a robustness measure associated with a property ϕ , and use it for system design purposes.

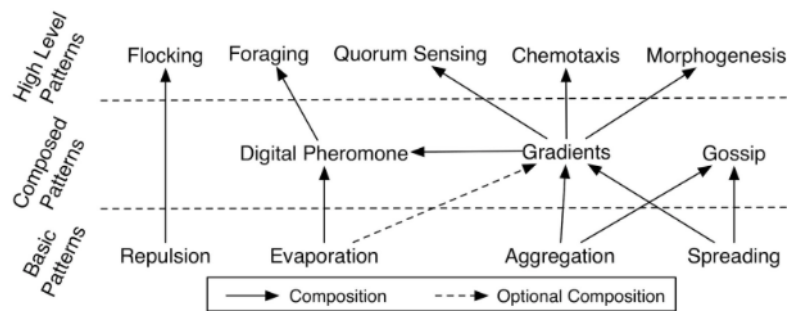
Applying these methods to CAS, however, still remains challenging. On one side, the size of such systems is so large that even fast statistical approaches can fail to provide an answer in a computationally acceptable time. In this respect, decomposition of the system into modules, seems an interesting direction for future work. The challenging problems here are how to identify such modules, and how to combine verification results of modules into a verification procedure (possibly providing bounds on the actual probabilities/robustness scores).

Another challenging problem is how to generalise the statistical approaches of [3, 7] to the case in which parameters are not only taking an unspecified value in a given set P , but they can also change over time (e.g. the arrival rate of customers at a bike or car sharing station will vary according to the time of day). This is a step towards verification of stochastic models of open CAS.

Finally, CAS are often subject to structural uncertainty, not only to parametrical one. In other words, the complete structure of the model itself may not be known. This seems to bring even more challenges to the analysis, particularly because structural changes of a model often result in discontinuous behaviour, which makes it much harder to exploit the statistical tools used in [3, 7] and similar work. Promising work on evolving models has recently been developed by Ghezzi *et al.* [17]. Whilst this was developed considering a specification that evolves as a system develops, it nevertheless has interesting ideas that could be applicable to the problem considered here.

Online verification

The importance of runtime verification has already been recognised by the software engineering community. In the simplest cases this might involve monitoring a system with respect to a formal specification of acceptable behaviour, such as a finite state machine, and raising an alarm if the observed behaviour deviates from the specification. However, systems such as CAS where the operating environment, the user requirements and the system itself are all changing over time, cannot be dealt with in such simplistic approaches. Moreover, functional



■ **Figure 2** Patterns for self-organisation [16].

correctness is no longer considered enough; in [9], the authors argue that quantitative aspects of behaviour must also be verified at runtime for self-adaptive software. In [19], the authors emphasise the need for formalising the adaptation components themselves, which is important to provide guarantees of correctness of the adaptation behaviour and changes of the adaptation logic to handle changing goals. Results from a related Dagstuhl seminar on Assurances for Self-adaptive Systems coined the term *perpetual assurances* [27] as an enduring process where new evidence is provided by combining system-driven and human-driven activities to deal with the uncertainties that the system faces across its lifetime.

In online verification a global model of the system may no longer be available, and if it is it is likely to be too computationally expensive to be used to give answers within the required timescale for runtime decision making. Thus it becomes attractive to develop compositional approaches to verification that allow results from lower levels, i.e. properties relating to one aspect of behaviour or relating to local behaviours, to be composed, building a global view from a number of local views. This may be addressed in a hierarchy, for example, with local verification giving rise to assurances about regional behaviour that can then be composed to give some assertion about global properties. It is likely that increasing levels of abstraction will be needed as we progress up the hierarchy, especially taking efficiency and response time into account. This remains a research challenge; whilst it is clear that boolean algebra provides the basis for composing verification results of qualitative analysis which lead to true/false answers, dealing with quantitative results of verification is more complex.

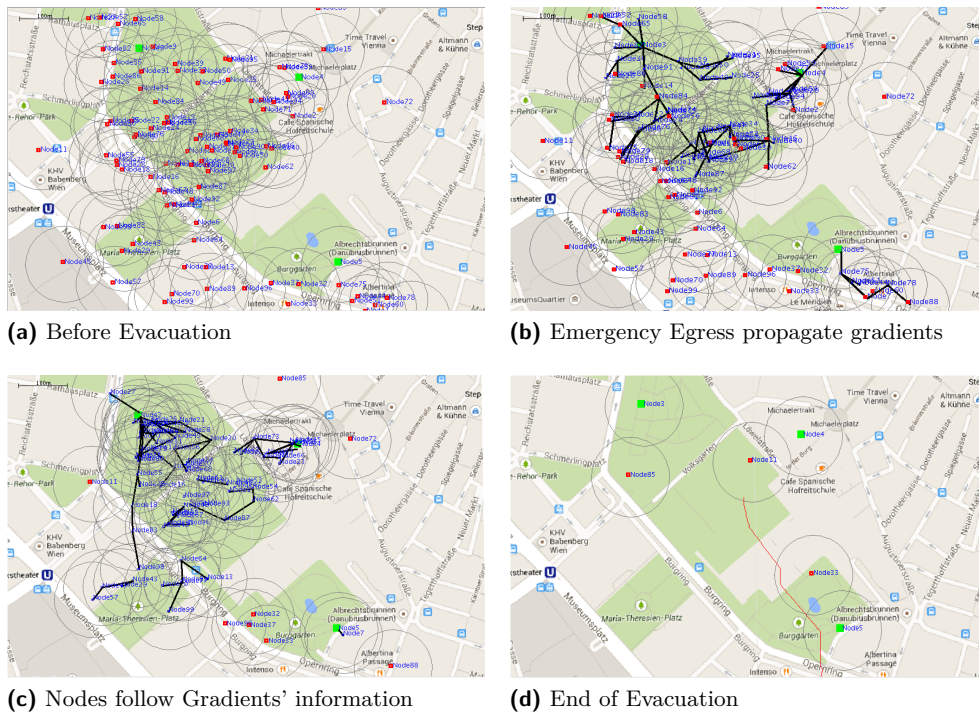
Another interesting approach for future work will be to combine statistical approaches with verification. For example, monitoring and verification could be combined with machine learning techniques to “learn” when a change in the system is likely to lead to an acceptable adaptation, or to guide adaptations in response to changes outside the system.

4.2.4 Subgroup II: Specification and verification of spatial self-organising patterns in CAS

In recent years a number of different frameworks have been proposed for the engineering of CAS. Here we particularly focus on the framework of self-organising patterns proposed in [16] and illustrated in Figure 2. Examples of these patterns include:

Spreading: a copy of the information (received or held by an agent) is sent to neighbours and propagated over the network from one node to another. Information spreads progressively over the system.

Aggregation: information is distributively processed in order to reduce the amount of information and to obtain meaningful information. Aggregation consists of locally applying



■ **Figure 3** Four snapshots of the emergency egress scenario: The emergency exits are the green boxes, the people that have not been reached yet are indicated by red boxes and the circles around people show the radius within which they can get in touch with their neighbours. After some time the dark lines indicate the gradient structure.

a fusion operator to synthesise macro information (filtering, merging, aggregating, or transforming).

Gossip: in large-scale systems, agents need to reach an agreement, shared among all agents, with only local perception and in a decentralised way. Information spreads to neighbours, where it is aggregated with local information. Aggregates are spread further and their value progressively reaches the agreement.

Gradient: information spreads from the location where it is initially deposited and aggregates when it meets other information. During spreading, additional information about the sender's distance and direction is provided: either through a distance value (incremented or decremented); or by modifying the information to represent its concentration (lower concentration when information is further away).

These patterns may be combined to design a system with collective adaptive agents who achieve a desired high-level outcome. For example, consider a scenario of emergency egress in a city. People are assumed to have handheld devices on which they can receive real-time information about the directions to follow to the nearest exit. This information is propagated via their neighbours. However, these devices have only a limited radius for local communication. The idea is therefore to create a dynamic ad-hoc network that aims to eventually reach everyone and provide the required information using dynamic gradients. Figure 3 shows four snapshots of a particular evolution of the system in the initial state and some later times.

When we consider verification there are a number of properties that are of interest in this case study. We mention just a few examples.

- As the mechanism of the communication is dependent on the gradient, it is important to assess its functional properties such as whether all nodes will be reachable.
- By chemotaxis, the agent should follow the gradient through the shortest path. Verification can assess whether all agents will reach the source of the gradient.
- We can also consider quantitative aspects of the induced behaviour such as the speed with which information is spread, the bandwidth consumed and the probability of reaching a certain fraction of people within a certain time.
- Spatio-temporal properties are also of concern such as whether at some future time all reachable nodes receive the gradient information (the spatial dispersion of information at a given time), or conversely at some given location all reachable nodes will receive the gradient information within a given time.
- Invariant properties may be important such as ensuring that the shortest paths are built and that the generated gradient structure does not have loops.

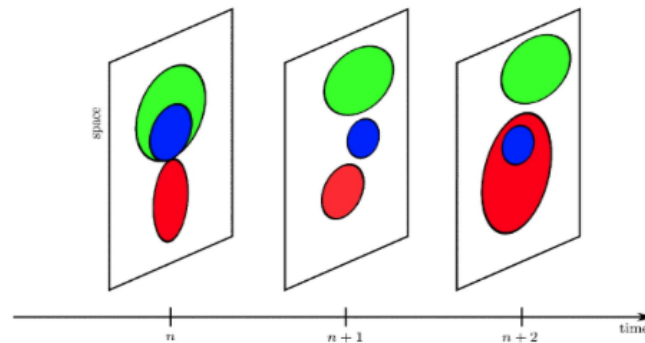
When an application is developed based on such patterns, the details of implementation are often delegated to an agent or service that implements the pattern in question. Therefore it is important that we have the means to independently verify the properties of the supplied patterns (local behaviours) as well as checking the emergent properties at the global level. In particular, it would be hugely beneficial to be able to develop mechanisms for compositional verification to use the verified properties of each pattern to derive the properties of the higher-level patterns, and ultimately properties of the applications build using those patterns. For example, considering the high-level patterns in Figure 2, is it possible to derive the correctness of a flocking pattern in a straightforward manner given the correctness of the repulsion pattern used to implement it?

Spatio-temporal verification via model-checking

Properties of different patterns at the collective level emerge from the coordination at the local level. Considering the example of dynamic gradients used to guide people to the nearest exits in an emergency situation such as that shown in Figure 3 [15], various global properties of the collective mechanism are of interest. For example, as already mentioned, one might like to be sure that all people involved at any time do receive gradient updates on directions within a given time.

Some such properties are spatio-temporal in nature and spatial [12, 10] and spatio-temporal model-checking [11] could be one of the techniques to be considered to automatically verify such properties. Spatio-temporal model-checking is a technique that requires a spatio-temporal model, on the one hand, and a spatio-temporal property, on the other. One way in which the model can be conceived is to consider it as a composition of a Kripke structure (S, T) to model the temporal evolution of the system and a spatial model for each state of the Kripke structure that reflects the spatial situation at the particular time. The latter can also be imagined as a “snapshot” of the spatial situation in a state of the Kripke structure. The spatial model can be conveniently chosen to be a closure space (X, C) , where X is a set of points, and C a closure operator which has its origin in topological spatial logics [26]. Such a choice covers a wide range of specific choices for the spatial structure, encompassing graphs, regular grids and also images similar to those shown in Figure 3.

Spatio-temporal properties address both the evolution in time and the spatial properties of a point in the model, i.e. a point (node) in the space at a particular state in the Kripke



■ **Figure 4** Schematic view of the evolution of a temporal path.

structure. In each possible world there is a different valuation of atomic propositions, inducing a different “snapshot” of the spatial situation which “evolves” over time. This is made clear along a temporal path. A path in the Kripke structure denotes a sequence of digital pictures indexed by instants of time. This is illustrated more schematically in the Figure 4, showing three different states of the Kripke structure at time step n , $n + 1$ and $n + 2$.

Spatio-temporal operators in STLCS (Spatio-Temporal Logic for Closure Spaces) [10] feature the usual Boolean operators (negation, disjunction etc), the CTL path quantifiers A (“for all paths”), E (“exists a path”), which must be followed by path-specific temporal operators XF (“in the next step”), $F1 U F2$, (“eventually $F2$ holds, but until then $F1$ must hold”), where $F, F1$ and $F2$ are STLCS formulas, and the spatial operators closure C and spatial until $F1 S F2$ (“ $F1$ surrounded by $F2$ ”). The two derived temporal operators G (“always”) and F (“eventually”) are also very useful.

Let us proceed with a few simple examples.

- Consider the STLCS formula $EG(\text{green } S \text{ blue})$. This formula is satisfied at a point x in the graph, associated with the initial state s_0 , if there exists a (possible) evolution of the system, starting from s_0 , in which point x is *always*, i.e. in every state in the path, *green* and surrounded by *blue*. The prototype spatio-temporal model-checker described in [10] will return (or colour) all the points x that satisfy the formula.
- A further, more complicated, nested example is the STLCS formula

$$EF(\text{green } S (AX \text{blue})).$$

This formula is satisfied at a point x in the graph associated with the initial state s_0 , if there is a (possible) evolution of the system, starting from s_0 , in which point x is eventually *green* and surrounded by points y that, for every possible evolution of the system from then on, will be *blue* in the next step.

- A simple example concerning the emergency egress case is the formula $AF(!\text{red } S \text{ false})$, where $!$ denotes negation. This formula is satisfied at a point x in the initial state if all possible evolutions of the system eventually reach a state in which there are no red points. Recall that red points correspond to nodes representing people who did not receive the directions to the nearest exit. So when this formula is satisfied it means that there is a point in time at which all people are being updated by the system. The particular expression $!\text{red } S \text{ false}$ is satisfied if none of the pixels are red because the surround operator is a spatial variant of a weak until operator.

This is an area of on-going work and there are a number of open issues for this line of research.

- Are the STLCS spatio-temporal operators sufficient to express the spatio-temporal properties that are relevant to the self-organising patterns used to design and implement CAS?
- If not, which other operators would be needed? Here we can think of operators that address performance aspects, for example to express that the probability that 90% of the people in the emergency egress example have been reached within a certain time-bound T , or more collective aspects such as that a set of points satisfies a certain property.
- What would be the right set of basic operators that on the one hand provide satisfactory expressiveness, or at least cover an interesting class of properties, and on the other hand are such that efficient model-checking algorithms can be found to verify them?
- Which derived operators and property templates are convenient to facilitate formulation of relevant properties in this context?

Much interesting and challenging work remains to find answers to these questions. Furthermore, there are other proposals that consider spatial and spatial-temporal logics. As an example, we would like to mention the spatial signal temporal logic [8]. This is a linear logic for the specification of behavioural properties of continuous signals which has recently been extended with some spatial operators. This logic has been applied in the domain of epidemiology to analyse the spread of a virus.

4.2.5 Concluding remarks

An important aspect of the engineering of CAS is providing evidence that the system requirements are satisfied, despite the uncertainty that may affect the system, its goals, and its environment. In this working group, we primarily focussed on verification of CAS to assess how well a system meets its specification or requirements. The specifics and characteristics of CAS poses several challenges to the problem of verification, including:

- How to express emergent properties of CAS and verify them?
- How to provide evidence in the face of uncertainty, in particular structural uncertainty, where the complete structure of the model of the system may not be known?
- How to enable runtime verification for CAS for which a global model is not available?
- How to enable compositional verification that uses verified properties of patterns at lower levels to derive the properties of higher-level patterns and ultimately properties of CAS built using those patterns?
- How to express spatio-temporal properties relevant for self-organising systems used to design and implement CAS?
- How to blend offline with online verification to provide the evidence that the system requirements are satisfied during the entire lifetime of CAS?

Whilst we enjoyed and benefited from a number of stimulating discussions around the topic of verification of CAS during the Dagstuhl seminar 14512, the time available did not allow us to make any significant developments beyond deepening our understanding and mutual appreciation. Nevertheless, the discussion helped us to hone our ideas and identify a number of exciting topics for future research, several of which are now being actively pursued by members of the working group.

References

- 1 Eugène Asarin, Alexandre Donzé, Oded Maler, and Dejan Nickovic. Parametric identification of temporal properties. In *Runtime Verification – Second International Conference, RV 2011, San Francisco, CA, USA, September 27–30, 2011, Revised Selected Papers*, volume 7186 of *Lecture Notes in Computer Science*, pages 147–160. Springer, 2011.
- 2 C. Baier and J.P. Katoen. *Principles of Model Checking*. MIT press, 2008.
- 3 Ezio Bartocci, Luca Bortolussi, Laura Nenzi, and Guido Sanguinetti. On the robustness of temporal properties for stochastic models. In *Proceedings Second International Workshop on Hybrid Systems and Biology, HSB 2013, Taormina, Italy, 2nd September 2013.*, volume 125 of *EPTCS*, pages 3–19, 2013.
- 4 Luca Bortolussi and Jane Hillston. Fluid model checking. In *CONCUR 2012 – Concurrency Theory - 23rd International Conference, CONCUR 2012, Newcastle upon Tyne, UK, September 4–7, 2012. Proceedings*, volume 7454 of *Lecture Notes in Computer Science*, pages 333–347. Springer, 2012.
- 5 Luca Bortolussi and Roberta Lanciani. Model checking Markov population models by central limit approximation. In *Quantitative Evaluation of Systems - 10th International Conference, QEST 2013, Buenos Aires, Argentina, August 27–30, 2013. Proceedings*, volume 8054 of *Lecture Notes in Computer Science*, pages 123–138. Springer, 2013.
- 6 Luca Bortolussi and Roberta Lanciani. Stochastic approximation of global reachability probabilities of Markov population models. In *Computer Performance Engineering – 11th European Workshop, EPEW Florence, Italy*, volume 8721 of *Lecture Notes in Computer Science*, pages 224–239. Springer, 2014.
- 7 Luca Bortolussi, Dimitrios Millios, and Guido Sanguinetti. Smoothed model checking for uncertain continuous time Markov chains. *CoRR*, abs/1402.1450, 2014.
- 8 Luca Bortolussi and Laura Nenzi. Specifying and monitoring properties of stochastic spatio-temporal systems in signal temporal logic. In *VALUETOOLS 2014*, In Press.
- 9 Radu Calinescu, Carlo Ghezzi, Marta Z. Kwiatkowska, and Raffaella Mirandola. Self-adaptive software needs quantitative verification at runtime. *Commun. ACM*, 55(9):69–77, 2012.
- 10 Vincenzo Ciancia, Stephen Gilmore, Diego Latella, Michele Loreti, and Mieke Massink. Data verification for collective adaptive systems: spatial model-checking of vehicle location data. In *2nd FoCAS Workshop on Fundamentals of Collective Systems*, IEEE Eight International Conference on Self-Adaptive and Self-Organizing Systems, page to appear. IEEE Computer Society, 2014.
- 11 Vincenzo Ciancia, Gianluca Grilletti, Diego Latella, Michele Loreti, and Mieke Massink. A spatio-temporal model-checker, <http://blog.inf.ed.ac.uk/quanticol/technical-reports/>. Technical report, The QUANTICOL project, 2014.
- 12 Vincenzo Ciancia, Diego Latella, Michele Loreti, and Mieke Massink. Specifying and Verifying Properties of Space. In Springer, editor, *The 8th IFIP International Conference on Theoretical Computer Science, TCS 2014, Track B*, volume 8705 of *Lecture Notes in Computer Science*, pages 222–235, 2014.
- 13 Edmund M Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT press, 1999.
- 14 Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, pages 68–72, 2003.
- 15 Jose Luis Fernandez-Marquez and Giovanna Di Marzo Serugendo. Assessment of robustness of spatial structures in crowd steering scenarios. Technical Report SAPERE TR.WP2.2013.07, SAPERE Project, 2013.
- 16 Jose Luis Fernandez-Marquez and Giovanna Di Marzo Serugendo. From self-organizing mechanisms to design patterns to engineering self-organizing applications. In *7th IEEE*

- International Conference on Self-Adaptive and Self-Organizing Systems, SASO 2013, Philadelphia, PA, USA, September 9-13, 2013*, pages 267–268. IEEE Computer Society, 2013.
- 17 Carlo Ghezzi, Claudio Menghi, Amir Molzam Sharifloo, and Paola Spoletini. On requirement verification for evolving statecharts specifications. *Requir. Eng.*, 19(3):231–255, 2014.
 - 18 Thomas A. Henzinger. Quantitative reactive modeling and verification. *Computer Science – R&D*, 28(4):331–344, 2013.
 - 19 M. Usman Iftikhar and Danny Weyns. Activforms: Active formal models for self-adaptation. In *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2014*, pages 125–134, New York, NY, USA, 2014. ACM.
 - 20 Diego Latella, Michele Loreti, and Mieke Massink. On-the-fly fast mean-field model-checking. In *Trustworthy Global Computing – 8th International Symposium, TGC 2013, Buenos Aires, Argentina, August 30-31, 2013, Revised Selected Papers*, volume 8358 of *Lecture Notes in Computer Science*, pages 297–314. Springer, 2013.
 - 21 Radu Mardare, Luca Cardelli, and Kim G. Larsen. Continuous Markovian logics – axiomatization and quantified metatheory. *Logical Methods in Computer Science*, 8(4), 2012.
 - 22 Dongyu Qiu and Rayadurgam Srikant. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 367–378. ACM, 2004.
 - 23 Aurélien Rizk, Grégory Batt, François Fages, and Sylvain Soliman. On a continuous degree of satisfaction of temporal logic formulae with applications to systems biology. In *Computational Methods in Systems Biology, 6th International Conference, CMSB 2008, Rostock, Germany, October 12-15, 2008. Proceedings*, volume 5307 of *Lecture Notes in Computer Science*, pages 251–268. Springer, 2008.
 - 24 Aurélien Rizk, Grégory Batt, François Fages, and Sylvain Soliman. A general computational method for robustness analysis with applications to synthetic gene networks. *Bioinformatics*, 25(12), 2009.
 - 25 Aurélien Rizk, Grégory Batt, François Fages, and Sylvain Soliman. Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures. *Theor. Comput. Sci.*, 412(26):2827–2839, 2011.
 - 26 Johan van Benthem and Guram Bezhanishvili. Modal logics of space. In Marco Aiello, Ian Pratt-Hartmann, and Johan van Benthem, editors, *Handbook of Spatial Logics*, pages 217–298. Springer, 2007.
 - 27 Danny Weyns, Nelly Bencomo, Radu Calinescu, Javier Camara, Carlo Ghezzi, Vincenzo Grassi, Lars Grunske, Poala Inverardi, Jean-Marc Jezequel, Sam Malek, Raffaella Mirandola, Marco Mori, and Giordano Tamburrelli. Perpetual assurances in self-adaptive systems. In *Assurances for Self-Adaptive Systems, Dagstuhl Seminar 13511*, 2014.

4.3 Humans-in-the-Loop in Collective Adaptive Systems

Jake Beal, Peter Lewis, Stefano Mariani, Jeremy Pitt, Nikola Serbedzija, Franco Zambonelli

License © Creative Commons BY 3.0 Unported license

© Jake Beal, Peter Lewis, Stefano Mariani, Jeremy Pitt, Nikola Serbedzija, and Franco Zambonelli

4.3.1 Introduction

Autonomous and autonomic systems have proved highly effective for self-* management of resource allocation in open, distributed computer systems and networks. Examples range from the ‘Ur’ application of autonomic systems in data centre management [8], to autonomous

virtual power plants for autonomous energy supply [1], to self-organising electronic institutions for resource provision and appropriation in multi-agent systems [21].

These systems are exemplars of collective adaptive systems (CAS): collective, because it requires cooperative, coordinated, synchronised or orchestrated effort of individual entities acting as a group; and adaptive, because it may change its shape, structure, functions, components, rules etc. at run-time, either reactively in response to changes in the environment, or pro-actively, in anticipation of such changes. The operation of such systems is, not unexpectedly, largely if not completely hidden from human users, and may be considered as being composed of purely ‘technical’ components. It is the speed, frequency and complexity of decision-making that precludes operator intervention, and necessitates these autonomous and autonomic approaches to control.

However, leaving data centre management or high-frequency trading algorithms to their own devices (as it were) is one thing, but it is something different when considering so-called ‘smart’ systems (smart homes, smart grids, smart cities, etc.), where the decision-making has a direct impact on qualitative human concerns, or actively requires the intervention of human agency.

The key question then, is: how can the design principles and operating principles for collective adaptive systems composed purely of ‘technical’ components be successfully transferred to resolve corresponding problems in *socio-technical* collective adaptive systems, i.e. CAS composed of both ‘human’ and ‘technical’ components, and involving human-human interaction, computer-mediated human-human interaction, human-technical component interaction (technical components including sensor, device, software agent, robot, etc.), and technical-technical component interaction. In other words, these are systems with ‘humans in the loop’, in which people interact with an electronically saturated infrastructure, or with each other through an electronically-mediated interface, especially when trying to achieve some collective action or common purpose.

This Working Group was constituted to address this precise question. In three main working sessions, we primarily considered three issues: scenarios which highlight the human factors that have to be taken into account in ‘programming’ socio-technical CAS for smart-* applications; a *general intervention framework* that might be used at design-time, run-time or even both, to account for these factors given the special operating conditions of socio-technical CAS; and finally a preliminary ontology for the specific *shaping mechanisms* which might be used to instantiate the general framework.

This report is structured with summaries of each working session, followed by some observations on research challenges and outcomes, and some concluding remarks. Note that for the purposes of this report, the term *socio-technical CAS* is taken to denote the same type of object as *CAS with “Humans-in-the-Loop”*.

4.3.2 ‘Programming’ Socio-Technical CAS

Modern ICT systems are de facto large-scale socio-technical systems whose goal is to collectively and adaptively behave in certain ways. In this context, the focus of the first discussion sessions was to analyze systematically the peculiar differences that such systems exhibit with respect to traditional ICT systems in engineering their behaviours, starting from existing works with user-centric aspects in the system design and or operation [21, 4, 5, 30, 2, 18, 9, 25, 16]. The session then focused on Humans-in-the-Loop from the perspective firstly of scenarios, and secondly of human factors, i.e. those aspects of human behaviour and psychology which distinguish socio-technical CAS from ‘technical’ CAS.

Scenarios

Smart Cities. It is an increasingly common practice to prefix the qualifier ‘smart’ to particular applications. The applications include smart grids (including smart meters), smart homes, smart cars, and so on. ‘Smart’, in this context, has come to mean systems that are characterised by three ‘i’s: instrumented, interconnected and intelligent. These applications all exhibit collectivity, adaptivity, and indeed other self-* computing properties. The various applications converge in the unifying concept of the Smart City, in which multiple interacting CAS will enable the energy distribution, transportation systems, and other city-wide infrastructure to continuously adapt and cope with the dynamism of the environment (changing traffic conditions, stochastic energy supply and demand, utilisation of common urban spaces, etc.).

However, there is, arguably, a fourth ‘i’ – interactive. All these applications also, at some point, have ‘humans in the loop’ for decision-making and even computation, have people as the recipients, as well as data sources, of a service (for example, participatory sensing applications), or have people’s goals, benefits or values as their focal point (for example, fairness and sustainability in energy or water distribution). Therefore this ecosystem is essentially one of socio-technical applications, and there is a requirement to take the people into account – in short, to recognise that Smart Cities are also places where citizens have to live.

In respect of this, the interleaving of autonomic, adaptive and ubiquitous computing, providing the foundations for self-* computing properties, and human-infrastructure interaction, providing the basis for innovative interfaces, affordances and ergonomics, will be critical in the design, and co-design, of ‘fit for purpose’ ‘user friendly’ socio-technical collective adaptive systems, and sub-systems, for Smart Cities.

Shared Living Spaces. Any shared living space, such as a communal flat, an open-plan office, or even a public space such as a park, require people to share a common space. This is a collective adaptive system: the ambience of the living space is both a goal of and function of the collective; but many aspects of the collective change over time, in particular the human ‘components’, but also the mutually-agreed conventional rules governing the use of the shared space.

Furthermore, violation of (implicitly or explicitly stated) these conventional rules, or social norms, can cause instances of incivility [22]. Such incivility, characterised by a low-intensity form of deviance from accepted norms, can be difficult to detect and resolve, but is also very harmful for the people who experience it regularly. Therefore, it is a pressing problem in both ergonomics and urban planning to reduce the negative side-effects of incivility.

One technological solution that has been proposed for addressing the incivility problem, is MACS (Affective Conditioning System): a system that attempts to avoid, reduce and/or resolve incivility in workplace environment before it escalates into a higher-intensity situation, e.g. conflict or aggression [24]. MACS is intended to emphasise stakeholder engagement and empower collective choice: firstly by avoiding micro-management, as incivility episodes are resolved between stakeholders (i.e. the occupants of the shared space themselves), and only as a last resort by appeal to higher authorities; and secondly by providing social support, through a network of communication and mutual obligations, via the collective selection, monitoring and enforcement of the stakeholders’ own social norms and pro-social processes such as forgiveness [28].

In MACS, the shared living space is envisioned as a *common pool resource* which we seek to manage according to the institutional design principles of Elinor Ostrom [17]. In this

respect, the metaphor we are pursuing is that the (intangible) ‘office ambience’ is a pooled resource which the office occupants can deplete by anti-social behaviour and re-provision by pro-social behaviour. Furthermore, what is (and is not) anti-social behaviour is determined by the occupants themselves – a specific instantiation of Ostrom’s third principle (that those affected by collective choice arrangements participate in their selection). Consequently, MACS implements a voting system for social norms, which allows for those (and only those) admitted to a shared space to vote positively or negatively for a norm. It also allows people to suggest new norms, as the dynamic nature of offices might mean there is a constant need to change norms, so MACS provides support for this process.

The Social Computer. The idea of the *social computer* [11] or *social computation* is for the designers of applications which synthesise the intelligence of human and automated computational units to tackle so-called ‘wicked’ problems [26]. Prototype examples of this type of collective adaptive system can be seen in crowdsourcing applications, such as Amazon’s Mechanical Turk, etc., with people doing the work that computers are not so good at, and computers the work at which people are not so competent.

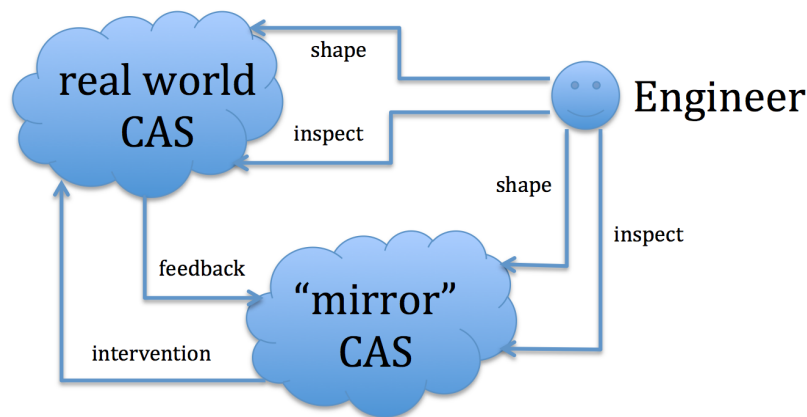
One of the prime examples of the potential of such systems is ReCAPTCHA, which leverages a security requirement to supplement the incompleteness of optical character recognition in digitising text. Ideally, the design and operation of CAS with humans-in-the-loop will thrive on achieving such externalities, i.e. where work required for one purpose will provide beneficial input for another.

Human Factors

Although this WG was composed of scientists and technologists rather than psychologists or sociologists, there was some experience of human-computer interaction and the discussion of (potentially disruptive) human factors identified the following features:

- People engage in micro-level behaviours, actions and decision-making which produces (potentially unexpected) macro-level outcomes, i.e. socio-technical CAS are also complex systems [7];
- Participation and engagement are critical in empowerment and politics but ‘attention’ remains a limited resource [6];
- Population change over times, and with it so do attitudes, cultures, fashions, etc.;
- People have different access to, perception of, and skills with technology;
- People don’t comply or not-comply with policies, they react to incentives implied by the policy [10], or find ways of interpreting a policy so that they consider themselves compliant;
- People are not equivalent to programmable components [11];
- People innovate themselves, and in particular utilise generative technology in unexpected ways [31];
- There is a trade-off between values, incentives and principles, often manifested in the form of social capital [20];
- A ‘spectrum’ of errors is to be expected, from accidents and triage, through to low-grade non-compliance and (regrettably) absolute malice;
- Governance is critical.

One conclusion drawn was that the design, deployment and evaluation of socio-technical CAS should be a multi-disciplinary, if not trans-disciplinary, process; and that the disciplines of psychology (and adaptation) [13, 12, 15], sociology (and collective intelligence) [29] and



■ **Figure 5** 3I Life-Cycle in Design and Monitoring of Socio-Technical CAS.

legal anthropology [3] can all make essential contributions to the study of socio-technical CAS.

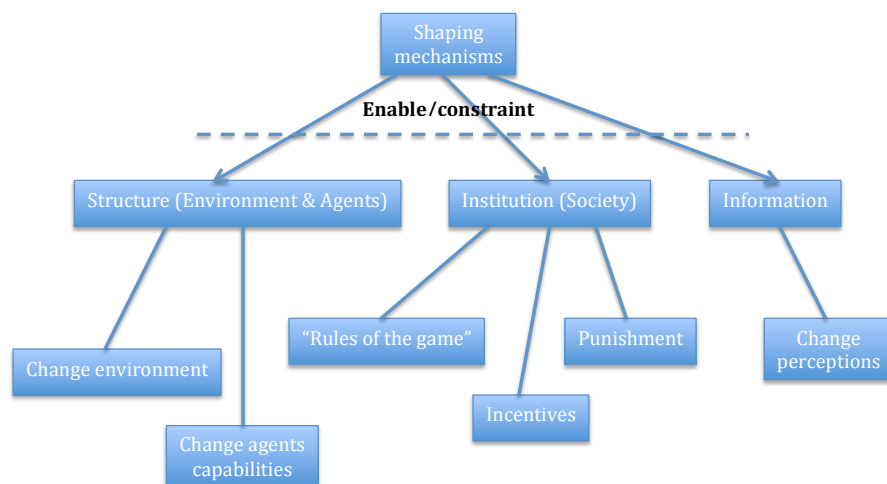
4.3.3 (Towards A) General Intervention Framework

CAS mostly are existing systems we have to simultaneously model (observe) and engineer (build) so as to intervene (influence) on the behaviour of both the collectives and the individuals. Thus, there is a need to embrace (model and engineer) uncertainty, error and unpredictability of behaviours and interactions, with the goal of drawing the boundaries within which individuals as well as collectives can behave and interact, be influenced and influence each other and the CAS as a whole. The main challenge here is that engineers cannot program the individuals (especially if humans); the path to pursue is that, on the other hand, engineers may shape the environment where individuals live to influence the way in which they behave and interact.

When humans enter the picture, it is no longer true that technology is neutral w.r.t. individuals' principles and values and in terms of their reaction to it (emotional, economic, etc.). Further, the impact of technology is also unpredictable and/or uncertain. Thus, there is a need to take into account individuals values at different scales (individuals, collective, CAS) and to recognize that whenever you offer a given technology to people you are simultaneously enabling and constraining (shaping) their capabilities of behaving and interacting within the CAS. The main challenge here is that failure, error, unforeseen emergent phenomena and misbehaviour (such as incivility) are an intrinsic part of the system.

Due to unavoidable uncertainty in modelling CAS and limited intervention opportunities and capabilities for engineers, simulation, model checking, etc. are no longer enough to ensure CAS sustainability. Thus, there is a need for live testing environments, running in a "mirror world" reflecting real-world data, behaviours and interactions while protecting real individuals, continuously providing feedbacks on CAS behaviour and absorbing patches, upgrades, intervention of engineers. The main challenge here is how to conceive, design and deploy these test environments.

The conclusion of the discussion led to the (preliminary) proposal of the the 3I life-cycle (Inspection-Innovation-Intervention) for the the design and operational management and governance of socio-technical CAS (see Figure 5). The idea is that system designers can inspect the behaviour of the system, innovate improvements which are tested in the "mirror" CAS or some sub-system of the CAS, and then make timely interventions in the "actual" CAS.



■ **Figure 6** A Shaping Mechanism Ontology for the General Intervention Framework.

Essentially, though what is required here is *general intervention framework* for socio-technical CAS. The specific mechanisms to shape or make interventions are discussed in the next section.

One further observation of this discussion, though, was that if any socio-technical CAS should be its own testbed, then consideration should be given to instrumentation, observability and running trials for sub-systems. In one sense, this could provide the basis for *evidence-based policy-making*, although how to design and run a controlled double-blind randomised experiment with policies (rather than, say, health treatments) – and not fall foul of ethical requirements – remains an open question.

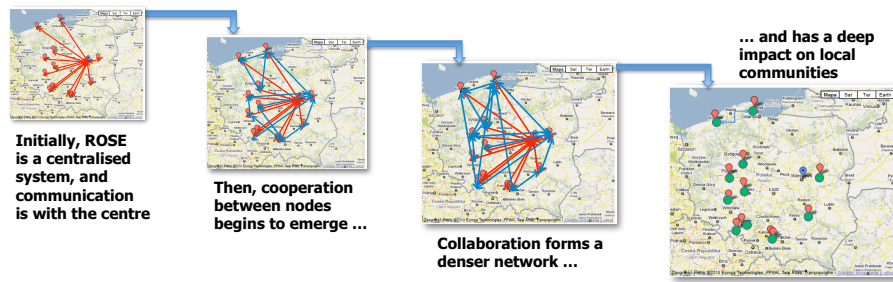
4.3.4 Shaping Mechanisms

The discussion in this session focused on the mechanisms for shaping intervention in the ‘landscape’ of a socio-technical CAS. Throughout, the group were seeking an analogy to the way that a city planner might intervene in traffic shaping by various mechanisms: for example physical mechanisms such as roundabouts (replacing traffic lights) and speed bumps, and policy-based mechanisms such as congestion charging and low-emission zones.

A Preliminary Ontology

Figure 6 illustrates a preliminary ontology of shaping mechanisms to instantiate the general intervention framework for designing, deploying and managing socio-technical CAS at run-time.

Figure 6 identifies three categories of mechanism, environmental (structural), institutional and informational. Environmental mechanisms includes changing the environment or the components. Institutional mechanisms includes changing the conventional rules, the incentives or the punishments (the system of retributive justice dealing with sanctions, or fear of sanctions (deterrence)). One informational mechanism is in the ‘sensory’ apparatus of the CAS. This has been referred to an *interoceptive collective awareness*, i.e. it is a sense that comes from within the collective as a whole, is concerned with the well-being of the collective, and is a precursor to collective action [20].



■ **Figure 7** Emerging Meso-level Structures in Project ROSE.

As an exemplar of a shaping mechanism, we briefly describe the experience of the ROSE project in Poland.

Exemplar: Project ROSE

To illustrate the principles and potential of a shaping mechanism for socio-technical CAS, we examine the experience of Project ROSE (Regional Centres of E-learning). The project started in 2004 at the Institute for Psychology of Informatics and Communication, directed by Professor Andrzej Nowak, of Warsaw School of Social Sciences and Humanities. The challenge was to promote the use of ICT, especially the Internet, in education in Poland. However, the rapid advances of ICT usually render any non-evolving educational program obsolete in just a few years. The solution was to create a learning community in the form of an expanding network of teachers that constantly adapt to new developments in ICT.

ROSE was based on the idea that teacher enhancement is a social change process rather than a transfer of knowledge. The Bubble Theory of Social Change [12] specifies how a sustainable social change may be achieved by concentrating on changing fragments of social networks (clusters or bubbles) rather than separate individuals. ROSE is therefore a mixture of face-to-face workshops and Internet mediated interactions. The workshops enabled the teachers to learn to collaborate with each other and to develop trust. From each workshop several individuals were selected as natural leaders to seed the ROSE network. After the initial workshop the training was conducted over the Internet using an e-learning platform. The communication structure resembled a star with the university performing the role of the central hub, and each school being a spoke (see Figure 7).

The leaders in each school initially worked with teachers from their own school but in the next stage schools already in ROSE collaborated with each other in the preparation of programmes for other schools. Meso-level structures (formal groupings with rules, roles, processes, designated groups responsible for decisions in specific areas, etc.; and informal groupings based on friendship circles, interest groups, etc.) emerged as clusters of collaborating schools, local administration and businesses etc. Afterwards, the meso-level structures grew stronger and bigger as more common initiatives were undertaken. The role of the university decreased as the network became increasingly decentralized.

This is a demonstration of using institutions as the shaping mechanism for a socio-technical CAS. This exemplar also again emphasises that disciplines from the social sciences, in this case dynamic social psychology [14], have highly relevant and significant insight to offer the development of socio-technical CAS.

4.3.5 Research Challenges

In the final working session, the WG identified and discussed a number of issues that remain unresolved in devising socio-technical CAS. This includes:

- Dispute resolution mechanisms for the various conflicts that may occur when, for example, members belonging to several communities with incompatible goals or notions of fairness collide;
- The “social ergonomics” that will need to be evaluated and refined: for example, even if the macro-objectives emerging at any one time are fair with respect to a society’s common good, and even if fairness is ensured in the long-term for each individual, this will not necessarily imply an acceptable *experience* for each individual in that society. Users may be willing to trade-off optimality for stability;
- The attention deficit: having algorithmic controls at their fingertips, individuals participating in a group may feel that they have no choice but to engage in a process of continuous negotiation and adaptation to rule-sets and social norms. The system’s affordances would engender an open cycle of societal self-adaptations and frequent change, inducing individual and collective stress and fatigue;
- Data: as observed in [27], the power of Big Data and associated tools for analytical modelling “... should not remain the preserve of restricted government, scientific or corporate élites, but be opened up for societal engagement and critique. To democratise such assets as a public good, requires a sustainable ecosystem enabling different kinds of stakeholder in society”.
- Economic security: how vulnerable would such CAS be to ‘hijacking’, by external parties and what could be the consequences? This especially concerns those socio-technical CAS which have an economic dimension and might see work at the edge but value only in the network or middleware [23];
- Governance security: a model of ‘good governance’ must be installed and maintained, and be equally robust to hostile takeover by minority interests;
- Externalities: are there any collateral costs, as well as benefits that such a system would place on society?

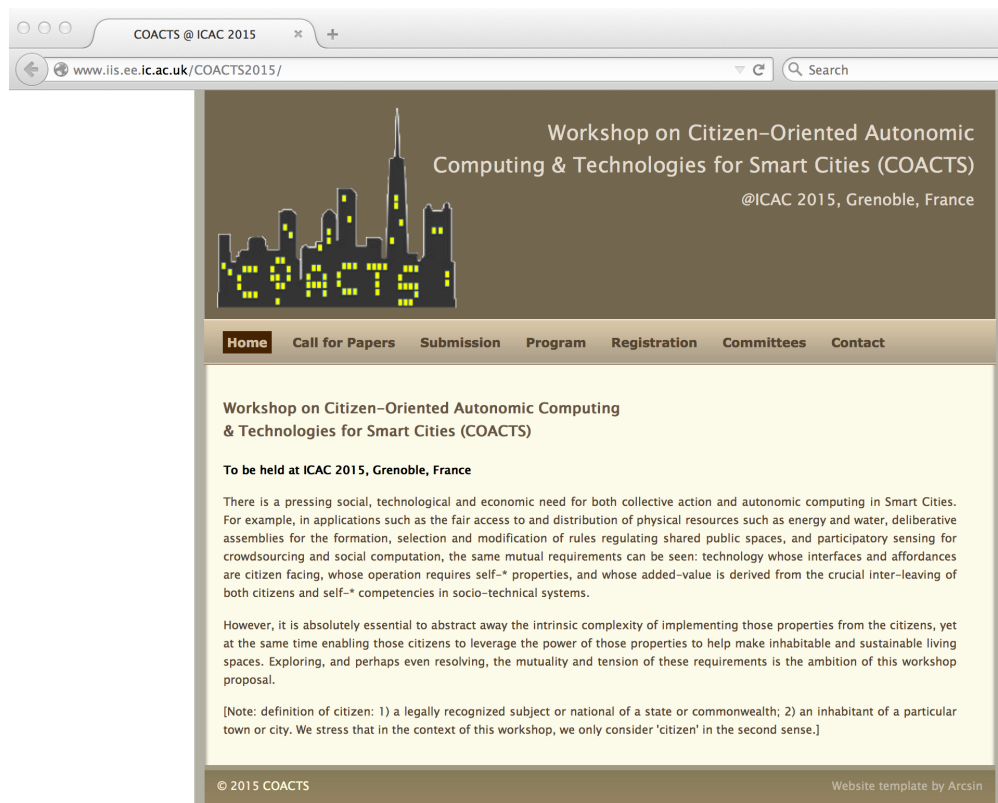
Such questions and the ensuing design requirements must be carefully considered before irreversibly embedding socio-technical CAS in the very fabric of everyday life. Since all possible scenarios cannot be predicted and addressed in advance, the ICT system itself must be sufficiently flexible to enable its evolution in parallel to the society it serves, which is why not just the 3I life-cycle is so important, but so too the integrity of the designers, programmers and managers, highlighting the need for *design contractualism* [19]

4.3.6 Outcomes

At the time of submission, there have been two significant outcomes of the Dagstuhl Seminar and the Working Group on Humans-in-the-Loop in particular.

The first is that the deliberations and discussions of the WG were highly influential in drafting a workshop proposal that has been accepted at ICAC 2015 (International Conference on Autonomic Computing), entitled COACTS (Citizen-Oriented Autonomic Computing and Technologies for SmartCities), see <http://www.iis.ee.ic.ac.uk/COACTS2015/> and Figure 8.

The second is the structure of a paper, co-authored by the WG participants, entitled *Towards a General Intervention Framework for Shaping Socio-Technical Collective Adaptive Systems*, which will be informed by this report.



■ **Figure 8** COACTS at ICAC2015 Website Home Page.

4.3.7 Summary and Conclusions

The WG has made a significant contribution to the Seminar’s goal understanding quantitative and qualitative analysis and modelling of CAS through an in-depth discussion of socio-technical CAS, or *CAS with Humans-in-the-Loop*. In particular, two specific contributions can be highlighted:

- a critical analysis of CAS scenarios with ‘humans in the loop’, and the identification of human factors which need to be taken into account concerning the design and operational principles of such CAS; and
- three innovative proposals for quantitative and qualitative analysis and modelling of socio-technical CAS: (i) a general intervention framework, (ii) the 3I life-cycle, and (iii) the ontology of shaping mechanisms.

As well as identifying some key research challenges, the WG believes that whatever else happens, collective adaptive systems for socio-technical applications with ‘humans in the loop’ need to be engineered properly, deployed responsibly, and managed with ‘good governance’.

References

- 1 Anders, G., Steghöfer, J.P., Siefert, F., Reif, W.: A trust- and cooperation-based solution of a dynamic resource allocation problem. In: SASO. pp. 1–10 (2013)
- 2 Beal, J., Dulman, S., Usbeck, K., Viroli, M., Correll, N.: Organizing the aggregate: Languages for spatial computing. CoRR abs/1202.5509 (2012), <http://arxiv.org/abs/1202.5509>

- 3 Casanovas, P., Pagallo, U., Palmirani, M., Sartor, G.: Law, social intelligence, nmas and the semantic web: An overview. In: *AI Approaches to the Complexity of Legal Systems – AICOL*. pp. 1–10 (2013)
- 4 Chourabi, H., Nam, T., Walker, S., Gil-Garcia, J.R., Mellouli, S., Nahon, K., Pardo, T., Scholl, H.: Understanding smart cities: An integrative framework. In: *IEEE Hawaii International Conference on System Sciences*. Maui (HI), USA (2012)
- 5 Conti, M., Das, S., Bisdikian, C., Kumar, M., Ni, L., Passarella, A., Roussos, G., Troster, G., Tsudik, G., Zambonelli, F.: Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber-physical convergence. *Pervasive and Mobile Computing* 8(1), 2–21 (2012)
- 6 Ferscha, A., Paradiso, J., Whitaker, R.: Attention management in pervasive computing. *IEEE Pervasive Computing* 13(1), 19–21 (2014)
- 7 Gell-Mann, M.: What is complexity? *Complexity* 1(1), 16–19 (1995)
- 8 Kusic, D., Kephart, J., Hanson, J., Kandasamy, N., Jiang, G.: Power and performance management of virtualized computing environments via lookahead control. *Cluster Computing* 12(1), 1–15 (2009)
- 9 Lewis, P.R., Marrow, P., Yao, X.: Resource allocation in decentralised computational systems: an evolutionary market-based approach. *Autonomous Agents and Multi-Agent Systems* 21(2), 143–171 (2010), <http://springerlink.metapress.com/content/h4t672m316678605/>
- 10 López, E.: *The Pursuit of Justice: Law and Economics of Legal Institutions*. New York, NY: Palgrave MacMillan (2010)
- 11 Miorandi, D., Maggi, L.: “Programming” social collective intelligence. *Technology & Society Magazine* 33(3), 55–61 (2014)
- 12 Nowak, A., Lewenstein, M., Szamrej, J.: Bable modelem przemian społecznych (bubbles: a model of social transition). *Swiat Nauki (Scientific American Polish Edition)* 12 (1993)
- 13 Nowak, A., Szamrej, J., Latane, B.: From private attitude to public opinion: a dynamic theory of social impact. *Psychological Review* 97, 362–376 (1990)
- 14 Nowak, A., Vallacher, R., Kus, M., Urbaniak, J.: The dynamics of societal transition: modeling non-linear change in the Polish economic system. *International Journal of Sociology* 35, 65–68 (2005)
- 15 Nowak, A., Rychwalska, A., Szamrej, J.: Social, psychological and technological determinants of energy use. *IEEE Technol. Soc. Mag.* 33(3), 42–47 (2014)
- 16 Omicini, A., Mariani, S.: Agents & multiagent systems: En route towards complex intelligent systems. *Intelligenza Artificiale* 7(2), 153–164 (2013), <http://dx.doi.org/10.3233/IA-130056>
- 17 Ostrom, E.: *Governing the commons: The evolution of institutions for collective action*. Cambridge, UK: Cambridge University Press (1990)
- 18 Pitt, J., Bourazeri, A., Nowak, A., Roszczynska-Kurasinska, M., Rychwalska, A., Santiago, I.R., Sanchez, M.L., Florea, M., Sanduleac, M.: Transforming big data into collective awareness. *Computer* 46(6), 40–45 (2013)
- 19 Pitt, J.: Design contractualism for pervasive/affective computing. *IEEE Technol. Soc. Mag.* 31(4), 22–29 (2012), <http://dx.doi.org/10.1109/MTS.2012.2225458>
- 20 Pitt, J., Nowak, A.: Collective awareness and the new institution science. In: Pitt, J. (ed.) *The Computer After Me*, chap. 11. IC Press (2014)
- 21 Pitt, J., Schaumeier, J., Artikis, A.: Axiomatisation of socio-economic principles for self-organising institutions: Concepts, experiments and challenges. *ACM Trans. Auton. Adapt. Syst.* 7(4), 39:1–39:39 (Dec 2012)
- 22 Porath, C., Pearson, C.: The price of incivility. *Harvard Business Review* 91(1-2), 114 (2013)

- 23 Reich, R.: The sharing economy is hurtling us backwards. Salon (February 2015)
- 24 Santos, M., Pitt, J.: Emotions and norms in shared spaces. In: Balke, T., Dignum, F., van Riemsdijk, M.B., Chopra, A. (eds.) COIN. LNCS, vol. 8386, pp. 157–176. Springer (2013)
- 25 Serbedzija, N.B., Fairclough, S.H.: Reflective pervasive systems. TAAS 7(1), 12 (2012), <http://doi.acm.org/10.1145/2168260.2168272>
- 26 Serugendo, G.D.M., Risoldi, M., Solemayni, M.: The social computer. In: Pitt, J. (ed.) *The Computer After Me*, chap. 11. IC Press (2014)
- 27 Shum, S.B., Aberer, K., Schmidt, A., Bishop, S., Lukowicz, P., Anderson, S., Charalabidis, Y., Domingue, J., de Freitas, S., Dunwell, I., Edmonds, B., Grey, F., Haklay, M., Jelasity, M., Karpistenko, A., Kohlhammer, J., Lewis, J., Pitt, J., Sumner, R., Helbing, D.: Towards a global participatory platform: Democratising open data, complexity science and collective intelligence. *European Physical Journal: Special Topics* 214 (2012)
- 28 Vasalou, A., Hopfensitz, A., Pitt, J.: In praise of forgiveness: Ways for repairing trust breakdowns in one-off online interactions. *Int. J. Hum.-Comput. Stud.* 66(6), 466–480 (2008)
- 29 Wood, L., Büscher, M., van Veelen, J.B., van Splunter, S.: Agile response and collaborative agile workflows. *IJISCRAM* 5(3), 1–19 (2013), <http://dx.doi.org/10.4018/ijiscream.2013070101>
- 30 Zambonelli, F.: Toward sociotechnical urban superorganisms. *IEEE Computer* 45(8), 76–78 (2012)
- 31 Zittrain, J.: *The Future of the Internet – And How to Stop It*. New Haven, CT: Yale University Press (2008)

Participants

- Jacob Beal
BBN Technologies –
Cambridge, US
- Lenz Belzner
LMU München, DE
- Luca Bortolussi
Universität des Saarlandes, DE
- Giacomo Cabri
University of Modena, IT
- Rocco De Nicola
IMT – Lucca, IT
- Giovanna Di Marzo Serugendo
University of Geneva, CH
- Vashti Galpin
University of Edinburgh, GB
- Marco Gribaudo
Politecnico di Milano Univ., IT
- Salima Hassas
University Claude Bernard –
Lyon, FR
- Jane Hillston
University of Edinburgh, GB
- Annabelle Klarl
LMU München, DE
- Roberta Lanciani
IMT – Lucca, IT
- Peter Lewis
Aston Univ. – Birmingham, GB
- Michele Loreti
University of Firenze, IT
- Stefano Mariani
Università di Bologna, IT
- Mieke Massink
CNR – Pisa, IT
- Ugo Montanari
University of Pisa, IT
- Laura Nenzi
IMT – Lucca, IT
- Jeremy Pitt
Imperial College London, GB
- Christophe Scholliers
Free University of Brussels, BE
- Hella Seebach
Universität Augsburg, DE
- Nikola Serbedzija
FhG FOKUS – Berlin, DE
- Mirco Tribastone
University of Southampton, GB
- Petr Tuma
Charles University – Prague, CZ
- Danny Weyns
Linnaeus University – Växjö, SE
- Martin Wirsing
LMU München, DE
- Franco Zambonelli
University of Modena, IT

