www.ascens-ist.eu

ASCENS

Autonomic Service-Component Ensembles

D2.2: Second Report on WP2

Models for Collaborative and Competitive SCEs, and Distributed Implementation of Connectors

Grant agreement number: **257414**
Funding Scheme: **FET Proactive**
Project Type: **Integrated Project**
Latest version of Annex I: **Version 2.2 (30.7.2011)**

Lead contractor for deliverable: **UNIPI**
Author(s): **Saddek Bensalem & Jacques Combaz (UJF-Verimag), Michele Boreale (UDF), Roberto Bruni (ed.) & Andrea Corradini & Fabio Gadducci (ed.) & Ugo Montanari & Matteo Sammartino (UNIPI), Alberto Lluch Lafuente & Andrea Vandin (IMT), Giacomo Cabri (UNIMORE), Diego Latella & Mieke Massink (ISTI), Mirco Tribastone (LMU)**

Reporting Period: **2**
Period covered: **October 1, 2011 to September 30, 2012**
Submission date: **November 12, 2012**
Revision: **Final**
Classification: **PU**

Project coordinator: **Martin Wirsing (LMU)**
Tel: **+49 89 2180 9154**
Fax: **+49 89 2180 9175**
E-mail: **wirsing@lmu.de**

Partners: **LMU, UNIPI, UDF, Fraunhofer, UJF-Verimag, UNIMORE, ULB, EPFL, VW, Zimory, UL, IMT, Mobsya, CUNI**

SEVENTH FRAMEWORK PROGRAMME

Executive Summary

This deliverable reports on the WP2 activities that have been conducted during months 13–24.

In Task 2.1 we further pursued the investigation on the theoretical foundations of connectors and their distributed implementation, the latter issue in particular for the BIP formalism. We introduced abstract models for those resource-aware and stochastic calculi we presented last year, and we freshly tackled the issue of tractability of markovian process algebras.

In Task 2.2 we started to focus on the soft constraints paradigm: on a new operator for the modular description of preference domains and on the use of constraints for increasing the expressiveness of programming languages. Moreover, we adapted some mechanisms inspired by the *Negotiate. Commit, Execute* (NCE) schema in order to address some foundational aspects of long-running transactions based on compensations. The aim is to suggest novel constructs to languages used for programming ensembles. We also report on a specification mechanism for the formal description of the conceptual framework for "white-box" adaptation introduced during months 1-12.

In Task 2.3 we finalized some of the previous works on the modelling of emerging behaviors in autonomic systems: those on game models for Grid systems and on coalgebraic techniques for dynamic systems. At the same time, we kept on investigating about issues related to game semantics for agents, focusing on energy trading and metering scenarios.

The deliverable is organized around the above structure of themes. A concluding section offers some general remarks on the overall satisfaction of the objectives, on the influences from and towards other work packages, and some comments on the foreseen developments for the forthcoming year.

Contents

1 Introduction

The three tasks of Work Package 2 span the whole duration of the Project; indeed, during months 13–24, we have been working on all of them. We summarize below our contributions to the package: a more detailed description for the contributions to each tasks, as well as a larger reporting of each line of research, can be found in subsequent sections. For each of the tasks the activities can be roughly divided along either two of three main strands of research.

1.1 Task 2.1 (Resource-aware operational models)

The overall aim of the task is concerned with "the study of resource-aware infrastructures and networking middleware modeled in terms of advanced components, glues and connectors which can support different levels of guarantees, reliability, dynamicity and integration to heterogeneous components."

Along those lines, also for the second year we further pursued the two main lines of research that we identified in the first year. Concerning the strand on *Foundations of resource-aware connectors*, we investigated on the general paradigm that we proposed during months 1-12 to overcome the fragmentation in the different notions and terminologies involving connectors. Our main accomplishment has been the ability to address also reconfigurable systems, an enhancement we envisioned since the beginning of our research. We further tackled the specific case of our main component framework, based on BIP. With respect to the first year, we advanced the results on its distributed implementation, now being optimized and fully addressing also priorities.

The activities in the second strand concerned the *Foundations of resource-aware languages and models*. It supersedes the strand *Advanced models of networking middleware* of Deliverabale 2.1: as for the first year, the emphasis was on the development of novel frameworks based on process calculi, yet with a specific interest towards the development of abstract equivalences. So, we proposed a functorial semantics for the *Network Conscious π-calculus* (NCPi), a network-aware extension of classical π-calculus that we introduced in the first year: its syntax allows expressing the creation and the activation of connections. Also, we further enhanced the uniform framework for stochastic calculi proposed in the first year, moving from SOS-like language definitions to a coalgebraic setting. Finally, a new contribution summarises the results on a behavioural relation for Markovian process algebras.

1.2 Task 2.2 (Adaptive SCs: building emergent behavior from local/global knowledge)

This task is concerned with the "develop[ment of] robust mathematical foundations for interaction scenarios [...] address[ing] models that can favor a mixture of static and dynamic analysis tools [...]".

At the end of the second year, the work on this task has now three different facets. Oon the strand titled *Negotiate, Commit, Execute mechanisms* we pursued the investigation on the *Negotiate, Commit, Execute* (NCE) paradigm, focusing on some foundational aspects of long-running transactions based on compensations to be used for programming ensembles. We defined a small-step operational semantics for the so-called concurrent Sagas language and devised a first-order dynamic logic for compensable processes. In particular, we propose a concurrent extension of dynamic logic that allows us to distill the hypothesis under which the correctness of compensable programs can be ensured.

We then introduced a novel strand of research on soft constraints. Under the title *Enhancements of conceptual models for autonomicity*, we at first consider a novel operator for the modular presentation of preference domains. Then, we offer a brief report on the different applications of the soft constraint paradigm as a specification technique: for adding expressiveness to process calculi, for a more flexible representation of knowledge, and finally for the modelling of the automotive case study. All these works are fully addressed in different deliverables of the second year.

On a more foundational issue, we further pursued the first year's work on the identification of *Conceptual models for autonomicity*. More specifically, in order to characterize the core nucleus identifying emergent behavior, during months 1-12 we developed a framework for adaptivity conceived around a prominent role of *control data*: computational data that are managed to enact adaptive behaviors. We report here the work towards a possible formalization of such conceptual notion. In particular, we present a suitable definition of *adaptable transition system*, our attempt to distill a formal model for adaptivity, to be used for the specification of properties related to the adaptive behaviour of a system.

1.3 Task 2.3 (Modeling SCEs with collaborative and competitive behavior)

The research on this last task has the ambition of "develop[ing] a theory combining as much as possible the flexibility and compositionality of computer science semantic models with the expressiveness of models developed by economic theorists", possibly with an in-depth analysis of the "adapt[ion] and re-use in this context many [of] coalgebraic tools well-known for ordinary transition systems".

Also this task has two different prongs. We first report on the application of *Game paradigms for service composition*. At first, a summing up of the investigation of a repeated non-cooperative job scheduling game is reported, whose players are Grid sites and whose strategies are scheduling algorithms, showing whether different strategies may reach a Nash equilibrium or not. Also, the application of adaptation in service composition has been further investigated. Last year, a peer-to-peer energy management scenario was modeled by exploiting an instance of the minority game (a family of games where the choice of the minority wins), where players have the chance to trade energy for purchasing to and for selling from a number of different actors. Now, the ability to automatically retrieve data to be used in adaptive and collaborative aspects, such as those affecting the consumption of electricity, has been further included.

We then describe more foundational contributions on *Coalgebraic techniques for dynamical systems*, pushing the coalgebraic view of weighted automata (WA). This is part of an ongoing research that aims at re-using in the WA context those coalgebraic tools from ordinary transition systems, in turn leading to coinductive reasoning and algorithms for equivalence checking and minimization in WA. Our ultimate goal is to develop a framework integrating classical models of computation and control theory: coalgebras and WA have proven useful towards achieving this integration.

2 On Task 2.1. Resource-aware operational models

The advances occurred during months 13-24 in Task 2.1 concerns two lines of research, both of them started during the first year. One strand concerns the *Foundations of resource-aware connectors*: it addresses the fragmentation linked to the many different notions and terminologies involving connectors, with the aim of developing a suitable paradigm for distributed and reconfigurable connectors. The other strand focused on *Foundations of resource-aware languages and models*: it supersedes the strand *Advanced models of networking middleware* in Deliverable 2.1, putting its emphasis on the development of novel specification frameworks based on process calculi.

2.1 Strand on "Foundations of resource-aware connectors"

It is largely acknowledged that software architectures are essential for mastering the complexity of systems and easing their analysis: they allow a separation between the detailed behavior of components and their overall coordination, which is often expressed by constraints that define possible interactions between components. The term *connector* has been coined within the area of component-based software architectures to denote entities that can regulate the interaction of a collection of components [PW92]. Indeed, component-based design relies on the separation of concerns between coordination and computation: the components are loosely coupled sequential computational entities that come equipped with a suitable interface (e.g. comprising the number, kind and peculiarities of communication ports) and the connectors can be regarded as (suitably decorated) links between the ports of the components. Semantically, each connector can impose suitable constraints on the communications between the components it links together (e.g. handshaking, broadcasting, multicasting). Then, the evolution of an ensemble can be seen as if played in rounds: at each round, the components try to interact through their ports and the connectors allow/disallow some of the interactions.

Recent years witnessed the development of different frameworks for the specification of connectors. A rigorous foundation is crucial for the analysis of coordinated distributed systems, thus motivating an increasing interest about the modeling of (different classes of) connectors. To this end, in the first year we focused on sketching a reference paradigm for unifying the different views so far advocated for connectors: we overviewed and compared some notable theories of connectors, as well as defined some mutual embeddings and planned some possible enhancements. However, despite their variety, more powerful classes of connectors are needed: due to the high dynamicity of autonomic component ensembles, connectors need to be empowered with mechanisms for resource- and network-awareness (the behavior of a connector may depend on the links it is tied to, e.g. for optimizing the routing of messages), as well as adaptation, reflection and reconfigurability. In the second year we started to address the issue of connectors for reconfigurable systems, i.e, systems in which the possible interactions among components are not fully defined at design time and may change during run-time. In order to foster the generality of our proposal, also this year we pursued the same overview and comparison aims with respect to the reconfigurable extensions of our reference paradigm.

Among the current proposals for dynamic architectures, needed for modeling reconfigurable systems or systems that adapt their behavior to changing environments, the BIP (Behavior, Interaction, Priority) framework is based on a semantic model encompassing composition of heterogeneous components. BIP uses an expressive set of composition operators, parametrised by a set of multiparty interactions and by priorities. The main focus of the second year work has been on the development of a methodology for transforming BIP models with priorities into distributed models that are directly implemented, as well as optimized by exploiting the knowledge of the system. The target model consists of components that represent processes and Send/Receive interactions that represent asynchronous message passing. Correct coordination is achieved through additional components implementing conflict resolution and enforcing priorities between interactions.

2.1.1 From static to dynamic connectors

Recent years have witnessed an increasing interest about a rigorous modelling of (different classes of) connectors [PW92]. This has led to the development of different mathematical frameworks that are used to specify, design, analyse, compare, prototype and implement suitable connectors. Our previous efforts have been focused at unifying different frameworks, in particular, the BIP component framework [BBS06], Petri nets with boundaries [Sob10] and the algebras of connectors [BLM06, ABC^{+}09] based on the tile model [GM00, Bru99]. In [BMM11b] we have shown that BIP without priorities, written BI(P) in the following, is equally expressive to Petri nets with boundaries. Thanks to the correspondence results in [Sob10, BMM11a], we can define an algebra of connectors as expressive as BI(P), where a few basic connectors can be composed in series and parallel to generate any BI(P) system. The algebra is a suitable instance of the tile model, which we propose as unifying framework for the study of connectors. In fact, the generality of the tile model as a semantic framework has been already witnessed in [MR99, BM99, FM00, BMR01b, KM01, GM02, BMM02].

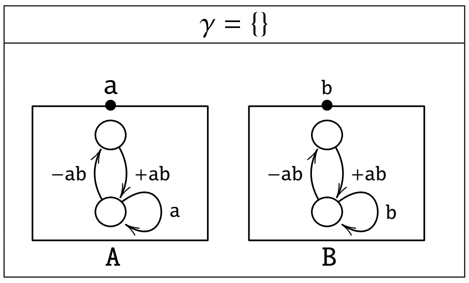

Figure 1: A simple reconfigurable BIP system.

All above approaches deal with systems that have static architectures, i.e., systems in which the possible interactions among components are defined at design time and remain unchanged during run-time. Nevertheless, when shifting to connectors for reconfigurable systems or systems that adapt their behavior to changing environments, the situation is less well-understood. In fact, we still lack a general and uniform theory for dynamic connectors. In order to contribute to the development of such a general approach, we planned to study the extension of the three frameworks mentioned above (i.e., BI(P), Petri nets with boundaries and the algebra of connectors) with different degrees of dynamicity. As a first step, we focused on the reconfigurable version of BI(P).

Reconfigurable BI(P). A reconfigurable BI(P) system allows for the dynamic modification of interactions among components, i.e., the set of available interactions changes as a side-effect of the synchronization between different parts of the system. For instance, consider the system depicted in Figure 1, which consists of two components: A and B. Components A and B exhibit just one port each (named a and b respectively). Initially, there is no allowed interaction in the system (in fact, the set of allowed interactions γ is empty). Nevertheless, a joint action between components A and B may add a new interaction. This takes place when both A and B perform the action with label +ab (the sign + indicates that the interaction ab has to be added to the set of allowed interactions). Once the interaction ab has been added to γ, the components A and B may interact as in any static BI(P) system by using the synchronization ab (i.e., a synchronization takes place when A executes a and B performs b). Analogously, the components can jointly decide to remove the allowed interaction ab by executing the action −ab (when $\mathtt{ab} \in \gamma$).

Mapping Reconfigurable BI(P) to Petri nets with boundaries. It can be shown that reconfigurable BI(P) systems can be represented as ordinary Petri nets with boundaries. This can be achieved by mapping components as in the ordinary case after considering reconfiguration actions such as +ab and −ab as additional ports of the components (this is shown in Figure 2 as $[\![\mathtt{A}]\!]$ and $[\![\mathtt{B}]\!]$). In addition, any interaction that can be added/removed dynamically from the system is mapped to an additional Petri net component (see the component $[\![\mathtt{ab}]\!]$ in Figure 2). Such component keeps track of whether the interaction is currently available or not. Finally, the glue between the different parts of the system is encoded into the component $[\![\gamma]\!]$. This net regulates which synchronizations can take place. For instance, the transition ab in $[\![\gamma]\!]$ captures the fact that the action ab only can take place when component $[\![\mathtt{A}]\!]$ performs a, $[\![\mathtt{B}]\!]$ performs b and the synchronization ab is enabled (this is represented by the fact that the net $[\![\mathtt{ab}]\!]$ may perform the action ab.

This mapping shows that the reconfiguration capabilities provided by reconfigurable BI(P) do not increase the expressive power of BI(P) (since any reconfigurable BI(P) system can be represented as a Petri net with boundaries, which in turn can be described as an ordinary BI(P) system). In fact, reconfigurable BI(P) only provides a more compact representation of ordinary systems.

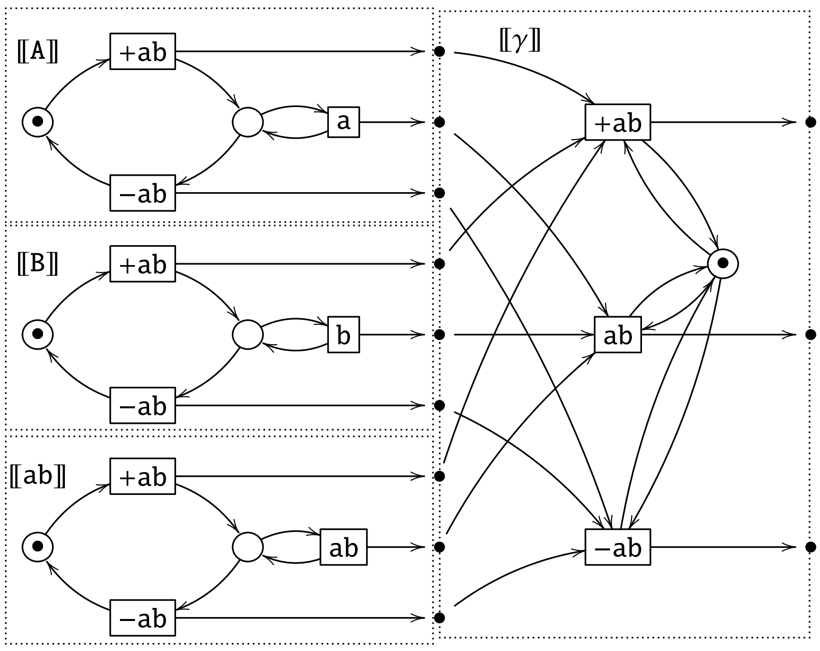

Figure 2: Mapping of a Reconfigurable BI(P) system into a Petri net with boundaries.

Novelty and future works. Note that the interfaces of components in reconfigurable BI(P) are static, i.e., the set of available ports in every component is fixed. As a consequence, the set of all possible interactions in a system are determined at design time (despite the fact that they can be enabled/disabled at run-time). Our next step is to explore situations in which the interfaces of the components may change dynamically (i.e., support the dynamic creation/elimination of ports). This requirement also imposes the necessity of handling interactions that can be created/removed dynamically. We take as motivating example the actual behavior of real servers, which interact with different clients by keeping separate sessions. In these cases, a session starts as a consequence of a client requesting for a connection. After the initial connection, each client synchronizes with the server by using a dedicated, private channel until the client decides to disconnect from the server. In addition, this example requires the modeling of the concurrent behavior of components. In order to deal with this class of systems we will consider an extension of BI(P) in which components may change their interface and exhibit concurrent behavior (they resemble the reconfigurable Petri nets of [AB09, BS01]). We also plan to study the mappings between the model of Petri nets with boundaries and the algebra of connectors.

2.1.2 Distributed implementation of BIP

During these last years, we have been focusing on distributed implementation for models defined using the BIP framework [BBBS08], which is based on a semantic model encompassing composition of heterogeneous components. The behavior of components is described as automata extended by arbitrary data and associated operations in C. BIP uses an expressive set of composition operators for obtaining composite components from a set of components. The operators are parametrised by a set of multiparty interactions between the composed components and by priorities, the latter used to specify different scheduling mechanisms between interactions.

Transforming a BIP model into a distributed implementation consists in addressing three issues

1. enabling concurrency: components and interactions should be able to run concurrently while respecting the semantics of the high-level model;

2. conflict resolution: interactions that share a common component can potentially conflict with each other;

3. enforcing priorities: when two interactions can execute simultaneously, the one with higher priority must be executed.

We developed a general method based on source-to-source transformations of BIP models with multiparty interactions leading to distributed models that are directly implemented [BBJ$^+$10, BBJ$^+$12]. This method has been later extended to handle priorities [BBJ$^+$10] and optimized by exploiting knowledge [BBQS12]. The target model consists of components representing processes and Send/Receive interactions representing asynchronous message passing. Correct coordination is achieved through additional components implementing conflict resolution and enforcing priorities between interactions.

In particular, the conflict resolution issue has been addressed by incorporating solutions to the committee coordination problem [CM88] for implementing multiparty interactions. Bagrodia [Bag89] proposes solutions to this problem with different degrees of parallelism. The most distributed solution is based on the drinking philosophers problem [CM84], and it has inspired the approaches of Perez et alii [PCT04] and Parrow et alii [PS92]. In the context of BIP, a transformation addressing all the three challenges through employing centralized scheduler is proposed in [BBBS08]. Moreover, in [BBJ$^+$10], we propose transformations that address both the concurrency issue by breaking the atomicity of interactions and the conflict resolution issue by embedding a solution to the committee coordination problem in a distributed fashion.

Distributed implementation of priorities is usually considered as a separate issue, and solved using completely different approaches. For example, in [BBJ$^+$10] priorities are eliminated by adding explicit schedulers components and more multiparty interactions. This transformation leads to potentially much more complex models, having definitely more interactions and conflicts than the original one. In [BBPS11, BPS10] the focus is on reducing the overhead for implementing priorities by exploiting knowledge. Yet, these approaches make the implicit assumption that multiparty interactions are executed atomically and do not consider conflict resolution. In a similar line of work, [BBQS12] aims at detecting false conflicts, that is, statically detected but never occurring at execution. However, this method still relies on conflict resolution protocols, at least for states where no false conflicts exist.

In this year work we proposed a combined implementation of the two coordination mechanisms: multiparty interactions and priorities. We proposed an intermediate model and transformations towards fully distributed models dealing adequately with both of them. The contribution is twofold.

First, we introduced an alternative observation-based semantic model for BIP. This model is general enough to encompass priorities and multiparty interactions and, moreover, to capture knowledge-based optimization as in [BBQS12]. Observation-based semantics reveals two types of conflicts occurring between interactions, which can be handled using different conflict resolution mechanisms.

Second, this model is used in an intermediate step of a transformation leading to a distributed implementation. We showed that observation conflicts that usually follow from the encoding of priorities could be dealt with more efficiently than structural conflicts, due to the sharing of components between multiparty interactions. We extended the counter-based conflict resolution protocols of Bagrodia in order to handle these types of conflicts. These extensions have been fully implemented.

We proposed different methods of generating a distributed implementation for multiparty interactions with observation. The model ensures enhanced expressiveness as the enabling conditions of an interaction can be strengthened by state predicates of components that are not directly involved in that interaction. It encompasses modeling of priorities, which are essential for modeling scheduling policies. We proposed a transformation from a model with observation into an equivalent one with interactions. The transformation consists in creating events making visible state-dependent conditions.

Novelty and future works. Expressing observation by interactions allows the application of existing distributed implementation techniques, such as the one presented in [BBJ$^+$12]. We have proposed an optimization of the conflict resolution algorithm from [Bag89] that takes into account the fact that an observed component does not participate in the observing interaction. Preliminary experiments show significant performance improvement of this optimized implementation method.

Future work directions include the study of knowledge-based techniques [BBQS12] for efficient conflict resolution, in particular by minimizing the set of the observed components for each interaction. We also plan to study optimized implementations of systems with multiparty interaction and observation, for other implementations based on other conflict resolution protocols, such as α-core [PCT04].

2.2 Strand on "Foundations of resource-aware languages and models"

Resource-aware calculi are languages with an explicit notion of computational resource and with primitives for resource allocation. Explicit, run time allocation of new resources is essential for adaptivity and autonomicity, as studied in ASCEns, since the additional resources, which can possibly include new knowledge, can dynamically improve the behavior of agents when needed. A paradigmatic example is the creation of new communication channels in the π-calculus. These calculi can be elegantly modeled using *presheaves*, that express the association between a collection of resources c and the set of programs $P(c)$ using those resources. This allows for a coalgebraic characterization of allocation along steps of computation. With the purpose of devising a general modeling framework for resource-aware calculi, we generalized the techniques presented in [FT01] and we applied them to Network Conscious π-calculus (see Deliverable 2.1 and [MS12a, MS12b]). The next steps will be studying of models for calculi with *constraints* as resources, where knowledge can be naturally represented via constraint systems, and investigating the applicability of our framework to cloud computing systems.

To integrate qualitative descriptions with quantitative ones in a uniform way within a single mathematical framework, in months 1–12 we introduced a uniform framework for the definition of Stochastic Process Calculi (SPC). The framework provided a simple and elegant solution to several issues typical of this class of languages, and it had been applied to the definition of significant fragments of major SPC proposed in the literature. In the second year of the project we addressed the notion of bisimulation for SPC from a coalgebraic point of view. A correspondence result is provided stating that bisimularity coincides with the behavioral equivalence of the associated functor. As generic examples, the concrete existing equivalences for the core of the stochastic process algebras PEPA and IML are related to the bisimulation of specific instances of the framework, providing, via the correspondence result, coalgebraic justification of the equivalences of these calculi.

To tackle the problem of state-space explosion in Markovian process algebra, suitable notions of behavioural equivalence (e.g. [HHM98]) have been introduced that induce *lumping* (e.g. [Buc94]) at the underlying continuous-time Markov chain (CTMC), establishing an exact relation between a potentially much smaller aggregated chain and the original one. For massively parallel systems, however, lumping techniques may not be sufficient to yield a computationally tractable problem. In many cases, even fluid approximations based on ordinary differential equations (ODEs), as e.g. in [Hil05], may be too large for feasible analysis. This contribution summarises the results of [TT12], which studies a behavioural relation for process algebras with fluid semantics, called *projected label equivalence*, which is shown to yield an *exactly fluid lumpable model*, i.e., an aggregated ODE system which can be related to the original one without any loss of information. The theory of exact fluid lumpability may be practically used to simplify models that exhibit many copies of identical composite processes, similarly to fluid approximations that are effective with many copies of sequential processes.

2.2.1 Resource-aware models of computation

Resource-aware calculi are languages with an explicit notion of computational resource and with primitives for resource allocation in their syntax. They are particularly suitable to express adaptivity, where manipulation of resources (e.g. control data) is an essential operation. Our work is mainly concerned with models for such calculi. In particular, we consider a class of models based on *presheaves*, that is, functors $\mathbf{C} \to \mathbf{Set}$. Here $\mathbf{C}$ is regarded as a domain of resources, and each $P : \mathbf{C} \to \mathbf{Set}$ as a way of associating a collection of resources c with the set $P(c)$ of programs using those resources. Coalgebras whose states belong to presheaves, instead of plain sets, enable the explicit characterization of resource allocation along steps of computation.

Presheaf models have several advantages. First of all, they allow a clean coalgebraic characterization of resource-aware calculi, which have a non-standard semantics due to the presence of notions such as α-conversion, capture avoiding renamings, freshness... Moreover, from a conceptual point of view, they provide a way of factorizing the model into "reusable" components. Concretely, this is analogous to frameworks based on virtual machines, where data types and their operations are encapsulated in a common layer that can be accessed through a variety of programming languages.

Our main aim is generalizing the presheaf framework, introduced in [FT01] for the π-calculus, in order to deal with non-trivial resources.

NCPi model. Our first case-study in this modelling activity based on presheaves is the *Network Conscious π-calculus* (NCPi), presented in [MS12a, MS12b] and in Deliverable 2.1. It is an extension of the π-calculus with an explicit notion of network: network nodes are represented as atomic names, network links are represented as names parametrized by two other names, telling their sources and targets. Therefore, computational resources for NCPi are nodes and links in a network, and allocation adds new pieces of network. In particular:

1. We defined a category $\mathbf{G}$ of graphs, representing communication networks, and we equipped it with two *allocation endofunctors* $\delta_s, \delta_l : \mathbf{G} \to \mathbf{G}$ that, respectively, add a new vertex and a new link for each ordered pair of vertices. These functors have counterparts in $\mathbf{Set}^{\mathbf{G}}$, Δ_s and Δ_l, defined by precomposition.

2. We gave a *behavioral endofunctor* $B : \mathbf{Set}^{\mathbf{G}_I} \to \mathbf{Set}^{\mathbf{G}_I}$ ($\mathbf{G}_I$ is the subcategory of $\mathbf{G}$ with only monos) employing Δ_s and Δ_l to model bound output transitions, and we proved that B has a final coalgebra. The mono restriction is needed to capture the NCPi transition system and its bisimulations. Then, we characterized the NCPi transition system as a B-coalgebra over a presheaf N_I of processes with injective renamings, showing that its coalgebraic bisimulations are ordinary NCPi bisimulations and the existence of a corresponding HD-automaton [MP05].

3. We derived a behavioral functor $\widehat{B} : \mathbf{Set}^{\mathbf{G}} \to \mathbf{Set}^{\mathbf{G}}$ and a $\widehat{B}$-coalgebra for NCPi from those of the previous point via a standard categorical construction that has been proven to perform *saturation* in the sense of [BKM06]. We then showed that bisimulations on the resulting coalgebra are ordinary bisimulations closed under all renamings, thus are congruences.

Novelty and future works. The definition of the presheaf semantics is an accomplishment of the current year, following the presentation of the syntax and operational semantics of the calculus.

Our next step will be investigating formalisms where *constraints* are regarded as resources. These are relevant for ASCEns, because knowledge has a natural representation as a constraint system, and enriching knowledge amounts to "allocating" new constraints. Once we have a model for constraints and their allocation, we can plug it in many languages, for instance a suitable SCEL dialect.

We also plan to study the applicability of the described framework to the cloud computing case study (see Deliverable 7.2). In fact, since models do not the depend on the "implementation" of resources due to the very structure of presheaves, this framework should be flexible enough to capture the fact that, in cloud computing systems, resource management mechanisms are independent from the language and behavior of user programs.

2.2.2 Bisimulation of State-to-Function LTSs

As reported in Deliverable 2.1, state-to-Function Labelled Transitions Systems (FuTSs) have been proposed as a uniform framework for the formal definition of the operational semantics of Stochastic Process Calculi (SPC) in the SOS style.

We recall the definition of finite support FuTSs. Let $\mathbb{C}$ be a commutative semiring and $\mathbf{FTF}(S, \mathbb{C})$ denote the class of total functions from set S to $\mathbb{C}$ with finite support. In the simplest case, a FuTS over $\mathcal{L}$ and $\mathbb{C}$ is a tuple $(S, \mathcal{L}, \mathbb{C}, \rightarrowtail)$ where S is a countable, non-empty, set of states, $\mathcal{L}$ is a countable, non-empty, set of transition labels, $\mathbb{C}$ is a commutative semiring, and $\rightarrowtail \subseteq S \times \mathcal{L} \times \mathbf{FTF}(S, \mathbb{C})$ is the transition relation. A finite support FuTS $(S, \mathcal{L}, \mathbb{C}, \rightarrowtail)$ is *total* if for all $s \in S, \ell \in \mathcal{L}$ there exists $\mathscr{P} \in \mathbf{FTF}(S, \mathbb{C})$ s.t. $s \overset{\ell}{\rightarrowtail} \mathscr{P}$ and is *deterministic* if the transition relation $\rightarrowtail$ is a function; for such a FuTS, in the sequel, we use the notation (S, θ), where θ is the transition function and $\mathcal{L}$ and $\mathbb{C}$ are assumed known from the context. For the purpose of this deliverable we consider total deterministic finite support FuTSs, since they are general enough as a model for SCPs, including those capturing non-deterministic behaviour.

In this deliverable we address the notion of FuTS bisimilarity, a natural generalization of the standard bisimilarity for Labelled Transition Systems (LTS) and of its extensions to probabilistic systems.

Let $\mathcal{F} = (S, \theta)$ be a FuTS over $\mathcal{L}$ and $\mathbb{C}$; we say that an equivalence relation $R \subseteq S \times S$ is an $\mathcal{F}$-bisimulation if $(s_1, s_2) \in R$ implies $\sum_{s' \in [s]_R} \theta(s_1)(\ell)(s') = \sum_{s' \in [s]_R} \theta(s_2)(\ell)(s')$, for all $s \in S$ and $\ell \in \mathcal{L}$, where $\sum$ is the n-ary extension of the sum of $\mathbb{C}$ and $[s]_R$ is the equivalence class of state s (w.r.t. R); $s_1, s_2 \in S$ are $\mathcal{F}$-bisimilar (notation $s_1 \sim_{\mathcal{F}} s_2$) if $(s_1, s_2) \in R$ for some $\mathcal{F}$-bisimulation R for S. It can be proved (see [LMdV12] for details) that standard process equivalences as *Strong Bisimilarity* for non-deterministic processes, *Strong Equivalence* for the Performance Evaluation Process Algebra (PEPA, [Hil96a]) and *Strong Bisimilarity* of Interactive Markov Chains (IMCs, [Her02]), which are among the most representative SPC, are instances of FuTS bisimilarity. Thus, the latter is a natural generalization of these notions of bisimilarity.

Furthermore, in [LMdV12], we have shown that the FuTS framework naturally fits in the coalgebraic setting of Category Theory. More specifically, for set $\mathcal{L}$ and semiring $\mathbb{C}$ we define a functor $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}$ on category **Set** of sets and function as follows, where by $f^{-1}(y)$ it is intended the set $\{x | f(x) = y\}$:

- for each set X in **Set**, $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(X)$ is the function space of all functions $\varphi : \mathcal{L} \to \mathbf{FTF}(X, \mathbb{C})$, i.e. $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(X) = \mathbf{FTF}(X, \mathbb{C})^{\mathcal{L}}$;
- for each function $f : X \to Y$ in **Set**, $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(f)$ is the mapping $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(f) : \mathbf{FTF}(X, \mathbb{C})^{\mathcal{L}} \to \mathbf{FTF}(Y, \mathbb{C})^{\mathcal{L}}$ where $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(f)(\varphi)(\ell)(y) = \sum_{x \in f^{-1}(y)} \varphi(\ell)(x)$, for all $\varphi \in \mathbf{FTF}(X, \mathbb{C})^{\mathcal{L}}, \ell \in \mathcal{L}$ and $y \in Y$.

A $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}$-coalgebra is, by definition, a pair (X, ρ) where X is a set and $\rho : X \to \mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(X)$ maps each element of x to an function in $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}(X)$. It follows easily from the definitions of FuTSs and of coalgebra that every FuTS over $\mathcal{L}$ and $\mathbb{C}$ is a $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}$-coalgebra. The interesting point here is that functor $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}$ has final coalgebra (Ω, ω) [LMdV12]. This means that we are guaranteed that for any $\mathcal{V}^{\mathcal{L}}_{\mathbb{C}}$-coalgebra $\mathcal{F} = (S, \theta)$—i.e. for any FuTS $\mathcal{F} = (S, \theta)$ over $\mathcal{L}$ and $\mathbb{C}$—there is a *unique* morphism $M_{\mathcal{F}}$ from $\mathcal{F}$ to (Ω, ω). This, in turn, immediately brings to a unique, *standard* notion of equivalence over S, namely the so called *behavioural equivalence*: two states are equivalent if $M_{\mathcal{F}}$ maps them to the same element of Ω, i.e. $s_1, s_2 \in S$ are *behaviourally equivalent* iff $M_{\mathcal{F}}(s_1) = M_{\mathcal{F}}(s_2)$. It is worth pointing out that the nature of this notion of equivalence does *not* depend on the particular FuTS, but only on functor $\mathcal{V}_{\mathbb{C}}^{\mathcal{L}}$ and that the functor provides only *structural* information on the *type* of transitions one chooses to deal with, namely the label set $\mathcal{L}$ and kind of information to associate to states via the transitions, i.e. the semi-ring $\mathbb{C}$. Of course, the specific states which are equivalent will depend on the specific FuTS $\mathcal{F}$ over $\mathcal{L}$ and $\mathbb{C}$ at hand, i.e. on the specific morphism $M_{\mathcal{F}}$.

It turns out that FuTS-bisimilarity *coincides* with *behavioural equivalence* of the functor $\mathcal{V}_{\mathbb{C}}^{\mathcal{L}}$, i.e. for all FuTS over $\mathcal{L}$ and $\mathbb{C}$ the following holds: $s_1 \sim_{\mathcal{F}} s_2$ iff $M_{\mathcal{F}}(s_1) = M_{\mathcal{F}}(s_2)$ [LMdV12]. The above result brings an (additional) argument in support of the concrete notion of FuTS bisimilarity and of its appropriateness. Furthermore, since FuTS bisimilarity coincides with standard Strong Equivalence for SPC (see, e.g., [Hil96a, Her02]), the above result bridges the gap between the original LTS-based definitions of SPC, with related equivalences, on the one end, and our FuTS semantics of the same SPC, with their equivalences and coalgebraic interpretation, on the other end[1].

We plan to further investigate the properties of functor $\mathcal{V}_{\mathbb{C}}^{\mathcal{L}}$ and of its multidimensional extension, for instance in relation with coalgebraic bisimilarity. We also plan to apply our approach to deterministically timed calculi, in particular those based on discrete time, and to extend it in order to cope with continuous deterministic time and with more general randomized, not necessarily Markovian, calculi, in order to pave the way for a general, quantitative extension of SCEL. Furthermore, we would like to investigate on FuTS equivalences other than bisimilarity [BDL10], within the coalgebraic framework (see, e.g., [BG06b, HJS07]).

2.2.3 Exact Fluid Lumpability in Markovian Process Algebra

In the CTMC realm, there is a one-to-one mapping between a state of the underlying labelled transition system and a CTMC state. Similarly, one may think of a specific function, solution to one differential equation of the overall ODE system, as a (continuous) state of the system. A lumpable partition is a partition over the set of the CTMC states such that some requirement over the transition rates is satisfied. In the case of exact lumpability, if it holds that the states in the same partition element are initially equiprobable, then they will be equiprobable at all future time points. This section summarises [TT12], where we introduce the analogous notion of *exact fluid lumpability*: intuitively, a partition over the ODEs of a model whereby two ODEs belonging to the same partition element have indistinguishable solutions, at all time points, *if their initial conditions are the same*. An aggregated ODE model may be defined which only considers a representative ODE for each partition element.

Aggregation may be induced by suitable behavioural relationships at the process algebra level. In PEPA, this may be accomplished by means of a *strong equivalence relation* over the states of a labelled transition system. The set of equivalence classes produced by such a relation represents the partition of the underlying lumped CTMC. Similarly, we define a notion of behavioural equivalence, called *projected label equivalence*, which induces an exactly fluid lumpable partition.

In PEPA, different strong equivalence relations may be merged to obtain possibly coarser partitions. More formally, the transitive closure of the union of several strong equivalence relations *always* induces a strongly lumpable partition. An analogous result holds for Fluid Process Algebra, but contingent to the model being *well-posed*. In such a case, we show that the transitive closure of the union of several projected label equivalences induces an exactly fluid lumpable partition. Furthermore, using this notion, we are able to characterise the relationship between projected label equivalence and stochastic notions of behavioural equivalence for PEPA. We find that, in general, for ill-posed models

[1]See [LMdV12] for details, where the theory is also generalized to multidimensional FuTSs, where different types of continuation functions, based on finite families of different semirings, allow to treat also SPC with different kinds of transitions, like, e.g. the language for IMCs.

Continuous-time Markov Chain Semantics	*Ordinary Differential Equation (Fluid) Semantics*
PEPA model P	Fluid Process Algebra model M, with *label set* $\mathcal{G}(M)$
Labelled transition system $(ds(P), \mathcal{T})$	ODE system of type $\dot{V}_{(H,P)}$, with $H \in \mathcal{G}(M)$
Exact lumpability: partition over CTMC states	Exact *fluid* lumpability: partition over $\mathcal{G}(M)$
Strong equivalence R_i over the set of states $ds(P)$	Projected label equivalence $\approx_{\mathcal{H}_i}$ over $\mathcal{G}(M)$
R_i induces a lumpable partition	$\approx_{\mathcal{H}_i}$ induces an exactly fluid lumpable partition
$(\bigcup_i R_i)^*$ induces a strongly lumpable partition	$(\bigcup_i \approx_{\mathcal{H}_i})^*$ induces an exactly fluid lumpable partition
(for any PEPA model P)	(if the FPA model M is well-posed)

Table 1: Summary of contributions and relationship with concepts related to the CTMC semantics of the Markovian process algebra PEPA. $ds(P)$ is the state space of the labelled transition system, $\mathcal{T}$ its multiset of transitions. The first column gives the novel notions and definitions available in the CTMC realm (from [Buc94] and [Hil96b]), the second one lists the novel contributions of [TT12].

nothing can be said about the stochastic behaviours of processes that are related by projected label equivalence. Instead, if the model is well-posed then projected label equivalence implies a stochastic equivalence, called *semi-isomorphism*, which is, informally, only a slightly weaker notion than relating two processes with isomorphic CTMCs.

Exact fluid lumpability may be used for aggregating models with copies of composite processes.

Novelty and future works Our study is analogous in spirit to that followed for obtaining aggregation of CTMCs induced by process algebra. In Table 1 we relate our contributions to those that are typically employed for aggregation in the CTMC semantics. We refer specifically to PEPA [Hil96b] and its fluid approximation, which we consider in a simplified version called *Fluid Process Algebra*.

The characterisation of label equivalence with respect to semi-isomorphism defines the boundary as to how much we may practically aggregate with our notion exact fluid lumpability. However, we feel that this is not the most that can be done. In this contribution, we have taken the path of considering as *atomic* the sequential components that make up a model; therefore, *label equivalence* was defined as a relation over such atoms. But a fluid atom does not give rise to a single ODE, as it induces as many ODEs as the size of its derivative set. Thus, it is natural to ask whether another behavioural relation could be devised over elements of derivative sets instead. In principle, such an approach might give rise to coarser partitions, hence more aggregated systems, than those that are obtainable through label equivalence. This will be subject of future work.

Our theory is concerned with a form of invariance between models which holds for all time points for which the ODE solution exists. Continuing along the analogy with the discrete-state realm, other topics of future investigation are the characterisation of approximate relations for further state-space reduction, and the study of equivalences which hold in specific points, in particular at equilibrium.

3 On Task 2.2. Adaptive SCs: building emergent behavior from local/global knowledge

In the Introduction we wrote that the research on this task should be concerned with the "develop[ment of] robust mathematical foundations for interaction scenarios [...] address[ing] models that can favor a mixture of static and dynamic analysis tools [...]".

At the end of the second year, the work on this task now shows three different facets. The first one, under the title *Negotiate, Commit, Execute mechanisms*, focuses on some foundational aspects of long-running transactions based on compensations, which are going to be used for programming ensembles. In particular, we have defined a small-step operational semantics for the so-called *concurrent Sagas* language and devised a first-order dynamic logic for compensable processes. In fact, while each activity has its own compensation, the main problem is how to guarantee that if one or more faults occur then the compensations are properly executed so to reach a consistent configuration of the system. We propose a concurrent extension of dynamic logic that allows us to distill the hypothesis under which the correctness of compensable programs can be ensured.

The second facet, *Enhancements of conceptual models for autonomicity*, concerns the use of soft constraints in various parts of the project. Some foundational issues were tackled in terms of adding modularity to the formalism, and some of the applications in the extension of programming languages and the representation of knowledge, fully addressed in other deliverables were reported.

We also further pursued the work of months 1-12 on *Conceptual models for autonomicity*. In particular, moving from the conceptual description of a white-box model for autonomicity, we planned to develop a possible formalization of it. We thus presented a suitable notion of *adaptable transition system*, our attempt to distill a formal model for adaptivity, to be used for the specification of properties related to the adaptive behaviour of a system.

3.1 Strand on "Negotiate, Commit, Execute mechanisms"

The Negotiate, Commit, Execute (NCE) scheme described in the proposal looks appropriate for those scenarios with unpredictable environments, where only part of the static analysis can be reused at run time, and decisions on how to adapt to the environment must be taken at run time. During the second year we have pursued this strand of research by enhancing long-running transactional formalisms to make them applicable to ASCENScomponents and ensembles. Roughly, a goal to be carried out is seen as a transaction. Any fault preventing the reachability of the goal can activate a so called *compensation* mechanism, across the whole ensemble, that has been programmed to undo as much as possible the effect of the wrong computation performed so far. For example, in the case of the robot case study, the compensation can be easily programmed so to bring back each robot in the ensembles along the path performed so far. The main result establishes that if basic activities have a correct compensation we can show the correctness of any compound compensable program. Moreover, we can use dynamic logic to reason about behavioural and transactional properties of programs.

3.1.1 Compensation-based approach to NCE

A first contribution of this strand concerns the definition of a novel small-step operational semantics for a compensation-based workflow language, concurrent Sagas [BMM05], already adopted in the area of business processes modelling.

Long-running transactions (LRTs) in ensembles can be composed by smaller services taken off-the-shelf. One important problem is failure recovery, i.e., the ability to bring a faulty ensemble back to a consistent state. As ensembles may grow large and complex, when a fault occurs the designer has to take several constraints into account: all sibling activities that run unaware of the fault should be stopped and all the activities that were executed before the fault need to be undone in a suitable order. Everything must be done in a distributed way, since each component is responsible for itself.

Concurrency makes the design of large ensembles an error-prone activity: components must be assigned with unambiguous semantics and early validated to detect unwanted behaviour and to suppress as many inconsistencies as possible.

The approach is based on the Sagas language, whose syntax is reported below.

$$\begin{array}{llcl}
(\textsc{act}) & A, B & ::= & a \mid \mathit{skip} \mid \mathit{throw} \\
(\textsc{process}) & P, Q & ::= & A \div B \mid P;Q \mid P+Q \mid P|Q \mid P^* \\
(\textsc{saga}) & S, T & ::= & A \mid [P] \mid S;T \mid S+T \mid S|T \mid S^*
\end{array}$$

Roughly, the basic activities of processes are *compensation pairs* $A \div B$, for A the atomic activity to be done and B its compensation installed when A is successful and to be executed in case a later fault occurs before the completion of the transaction. Transaction boundaries are marked by $[P]$ and form atomic activities of larger Sagas. Both processes and Sagas can be composed in series, in (non-deterministic) choice, in parallel or can be iterated for a finite number of times.

In the literature, different compensation policies have been defined for parallel processes. These can be classified along two main axis [BBF+05, BKLS10]: centralised vs distributed and interruptible vs non interruptible. During the second year we have defined a flexible, small-step operational semantics via a suitable labelled transition system defined in Plotkin's SOS style that can accommodate for all policies under minor variants of the SOS rule for parallel composition [BK12].

In [BFK12] we introduced a rigorous language of compensable concurrent programs by instantiating the abstract actions of Sagas with multiple assignments and conditional tests. The language is accompanied by a dynamic logic for reasoning about compensation correctness and verification of behavioral properties of compensable programs according to the policy of centralised compensation without interrupts. In this sense we go one step further of e.g. the approach in [VF10, CFV09], where the formulas were mainly concerned with the temporal order of execution of actions in a message-passing calculus with dynamic installation of compensation, by allowing to express properties about the adequacy of the state restored by the compensation after a fault occurred. Furthermore, even though compensation correctness is ensured by construction, our logic allows the verification of strong and weak correctness for compensable programs that contain compensation pairs that are not correct.

As detailed in [BFK12], our dynamic logic differs from previous proposal for the way in which concurrency is handled and for dealing with compensations.

Novelty and future works. The work in [BK12] lays the semantic foundation for the study of ensembles programmed as Sagas, where basic activities can be paired with compensation activities whose execution is delegated to the preferred policy of compensation. For example, in the robot case study it would be immediate to associate with movement activities their opposite movement as compensations and to guarantee by construction that any fault at the level of the ensemble would bring each robot back to its initial position. We plan to extend the semantics to arbitrarily levels of nested Sagas and to experiment with the Sagas language as a high-level language for the early prototyping of components and ensembles, possibly transferring some of the concepts to the level of SCEL.

On the verification side, our research programme leaves as ongoing work the development of a suitable computational model and corresponding logic for allowing a quantitative measure of correctness, so that different kinds of compensations can be distinguished (and the best can be selected) depending on their ability to restore a more satisfactory state than the others can do. Moreover, we would like to develop suitable equivalences over states that can reduce the complexity of the analysis, and facilitate the development of automatic reasoning tools.

3.2 Strand on "Enhancements of conceptual models for autonomicity"

While classical constraint satisfaction problems (CSPs) search for the boolean assignment of a set of variables that may satisfy a family of requirements, its soft variant extends the (possibly ordered) domain of assignments, thus modelling preferences: the aim is to provide an environment where suitable properties could be proven and inherited by all the instances.

Besides flexibility, the use of these formalisms has been advocated for their modularity: suitable operators can be defined, in order to manipulate such structures and build new ones. However, some intuitive construction so far were not covered: it proved e.g. impossible to recast in the soft CSP fold the case of lexicographic orders, i.e., sets whose elements are pairs and position plays a role. Given two partially ordered domains used for variable assignments in the soft CSP framework, some current work reported here shows under which conditions a new one can be built, that precisely corresponds to the lexicographic order of those two. The result allows for a wider application of the soft formalism, and it is going to be pivotal for the intended use of soft constraints in the modelling of ensembles, whose components must often verify constraints on more than one feature, yet the features themselves are equipped with a suitable order.

Moving from its roots on the specification of optimisation problems, (soft) constraints have been adopted for specification purposes into (either declarative or procedural) programming languages. Similarly, their use has been advocated as a tool for knowledge representation and manipulation. Ordinary logic programming is extended with soft constraint (SCLP) by replacing equality predicates with functions yielding values in a partially ordered domain [JL87, BMR01a]. The SCLP framework has been applied to the ASCEns automotive case study via the CIAO system [MM12]. The (soft) concurrent constraint paradigm (SCC) includes a procedural part, in the process calculi style, based on guarded primitives like ask and tell [Sar93, BMR06]. Current work concerns the embedding of the relevant parts of SCC programming into the ASCEns language SCEL. For the purpose of obtaining a flexible way of specifying knowledge representation (KR) in ASCEns, an integration of the constraints paradigm with KnowLang has been proposed [MV12]. Also, the need of modeling scenarios where knowledge is emergent and distributed (meaning that a common ontology is specialized in local KR styles and resolution engines) put forward the work on distributed constraint handling, as witnessed by some preliminary assessment using the PROVA rule language.

3.2.1 Enhancing the soft CSP toolset

Classical constraint satisfaction problems (CSPs) search for the assignment of a set of variables that may satisfy a family of requirements. Constraint propagation (as e.g. represented by local consistency algorithms) embeds any reasoning which consists of explicitly forbidding values or their combinations for some variables of a problem because a given subset of its constraints cannot be satisfied otherwise.

The soft framework extends the classical constraint notion in order to model preferences: the aim is to provide a single environment where suitable properties (e.g. on constraint propagation) could be proven and inherited by all the instances. Technically, this is done by adding to the classical notion of CSP a representation of the levels of satisfiability of each constraint. Albeit appearing with alternative presentations in the literature, the additional component consists of a poset (stating the desirability among levels) equipped with a binary operation (defining how two levels can be combined).

Definition 3.1 *A* semiringal valuation structure *(SVS) is 4-tuple $G = \langle A, \leq, \otimes, \top \rangle$ for $G^{\leq} = \langle A, \leq, \top \rangle$ a join semi-lattice (JSL) with top, $G^{\otimes} = \langle A, \otimes, \top \rangle$ a commutative monoid, and distributivity $a \otimes (b \vee c) = (a \otimes b) \vee (a \otimes c)$ holds.*

Our SVSs are a generalization of valuation structures [SFV95], replacing their total order with a JSL, as well as of c-semirings [BMR97] (also known as absorptive semirings [BG06a]), removing the requirement of a bottom element $\perp$ (which is by construction also an annihilator, i.e., such that $a \otimes \perp = \perp$). The lack of such a (necessarily unique) element is going to be be pivotal in our proposal for modelling constraints whose degree of satisfaction is based on lexicographic orders.

Indeed, it proved impossible to recast in the soft CSP fold the case of lexicographic orders, i.e., sets whose elements are pairs and the position plays a role. Assuming two partial orders $\leq_0$ and $\leq_1$, the associated lexicographic order is

$$\langle a_0, a_1\rangle \leq_l \langle b_0, b_1\rangle \text{ if } \begin{cases} a_0 <_0 b_0 & \text{or} \\ a_0 =_0 b_0 \ \& \ a_1 \leq_1 b_1 \end{cases}$$

with $a < b$ meaning that $a \leq b$ and $a \neq b$. It is easy to see that $\leq_l$ is a partial order. Indeed, it is a JSL if both $\leq_0$ and $\leq_1$ are so, and the latter also is bounded, i.e., it has a bottom $\perp_1$.

Assuming two SVSs G_0 and G_1, a monoidal tensor $\otimes_l$ is also easily defined point-wise. However, the distributivity law usually fails. Such a failure motivated in [HMW09] the introduction of a novel formalism, alternative to the standard soft constraint technology. In our work we follow a more traditional path, proving that under suitable conditions on G_0, also $Lex(G_0, G_1) = \langle A_0 \times A_1, \leq_l, \otimes_l, \langle \top_0, \top_1\rangle\rangle$ falls back to the standard soft CSP fold. Our main result is summed up below.

Theorem 3.1 *Let G_0 and G_1 be two SVSs such that $\leq_0$ is total and strictly preserved by $\otimes_0$ (i.e., $a < b$ implies $a \otimes c < b \otimes c$). Then, $Lex(G_0, G_1)$ is a SVS.*

A few SVSs... We thus first recall some instances of SVSs. The most straightforward example is the classical boolean algebra $B = \langle\{\perp, \top\}, \implies, \vee, \top\rangle$. Possibly, the most prominent example of a SVS is the so-called tropical semiring $T = \langle\mathbb{N}, \geq, +, 0\rangle$, with $\geq$ the inverse of the usual order notion on natural, thus such that 0 is the top element [Pin98]. It is used used for modelling those problems where a cost has to be minimised, as in shortest path scenarios under the name of weighted constraint satisfaction [LS03]. Its bounded version $T_\top$ is obtained by adding the infinite $\top$ as the bottom element. Otherwise, the monoidal operator can be replaced by the standard natural multiplication, for $M = \langle\mathbb{N}^+, \geq, \cdot, 1\rangle$, for $\mathbb{N}^+$ the positive natural numbers. The complementary segment $[0, 1]$ over the rational numbers (possibly extending to the real ones) define the fuzzy SVS $F = \langle[0, 1], \leq, \cdot, 1\rangle$ [Coo03].

...and a simple example The running example considered in [HMW09] was proposed in order to model a scenario concerning the scheduling of meetings. Among other requirements, each person has to express a possible degree of preference for a given date, and it must moreover explicitly state how much his/her presence is actually crucial for that meeting.

We can revise and enrich a bit the original example, using some of the instances we just defined. We may assume to have three possible meetings to organise, each one of them among five possible dates (of course, any pair of natural numbers would do). Some dates might not be compatible, since they overlap (at last, each date overlaps with itself) or because some people are not willing to have too close meetings. For the three meetings, each date (in fact, also each pair and each triple of dates) has associated a set of possible values, representing e.g. the interest and willingness of the persons to be in. The problem is to find the set of three dates maximising such features.

Summing up, each value has to state how much a person is crucial for that meeting, and its willingness to appear in a given date. Also, we may record the status of a person among the group of attendees of the meetings (e.g., the position in a firm or the level of expertise in a technical team).

The last feature can e.g. be modelled by T, with 0 the top, and it is possibly the most important condition to guarantee. Instead, the statement of the relevance of a person for a meeting can be expressed as B, and its willingness by F. The preference domain for this scenario might then be adequately represented as $H(Lex(B \times F, T_\top))$. Clearly, $B \times F$ is a composable SVS, and $T_\top$ is a bounded SVS; hence, $Lex(B \times F, T_\top)$ is a SVS, to which the power-domain operator H can be applied.

Novelty and future works. The work represents a novel strand of research, loosely inspired by the work on enhancing constraint manipulation in [BG06a]. The proposal represented by SVSs may foster a rethinking of the soft approach, and it should possibly directed towards the description of more general structures such bipolar constraints [BPRV11].

3.2.2 Soft constraints for language extensions and knowledge representations

Moving away from their roots as a specification formalism for optimisation problems, (soft) constraints have been advocated for increasing the expressiveness of programming languages, equipping them with a suitable logic for manipulating data structures. Constraint logic programming (CLP) [JL87] extends logic programming (LP) by embedding constraints in it: term equalities are replaced with constraints and the basic operation of LP languages, unification, is replaced by constraint handling in a constraint system. It therefore inherits the declarative approach of LP, by also offering efficient constraint-solving algorithms. As reported on the Deliverable 7.2 concerning case studies, as well as on the Joint Deliverable JD2.2, the soft CLP framework has been adopted [MM12] as a high-level specification formalism to model and solve the e-mobility optimization problem [KZBS12]. As an example, the contribution shows how to model the optimization sub-problem consisting in finding the best trips in terms of travel time and energy consumption. This problem substantially coincides with the multi-criteria version of the shortest path problem, and it is then used to model the e-mobility optimization problem consisting in finding the optimal journey. In order to actually execute the soft CLP program, CIAO Prolog (www.ciaohome.org) has been adopted.

The (soft) concurrent constraint paradigm (SCC) includes a procedural part, in the process calculi style, based on guarded primitives like *ask* and *tell*, as well as negotiation constructs [Sar93, BMR06]. Current work concerns the embedding of (relevant) parts of SCC programming into SCEL. As reported in Deliverable 1.2, along the lines of the work of cc-pi calculus [BM07] a new dialect of SCEL has been proposed, ccSCEL, for enabling SCC programming via the interaction primitives of SCEL, Indeed, the linguistic constructs of SCEL look convenient, since they are equipped with a logically defined notion of ensemble, for modeling the dynamic interaction taking place during constraint propagation. A suitable scenario is currently investigated for the implementation of a social odometry scenario for self-adapting robots [GCS+08]. In the ordinary execution phase the behavior is guided by the current knowledge of the robot, which however tends to become obsolete. In the constraint propagation phase, the knowledge is procedurally updated on the basis of deduction procedures activated by the interaction with other robots, we could say by additional interaction with the environment.

For the purpose of obtaining a more flexible way of specifying knowledge representation (KR) in ASCEns, an integration of the constraints paradigm with KnowLang, a formal language for knowledge representation in autonomic and self-adaptive systems, has been recently proposed [MV12]. In general terms, soft constraints might be used as a KR technique that will help designers to impose constraining requirements for special liveness properties of an intelligent system, considered as an approximation to our understanding of good-to-have properties. On the KnowLang paradigm this implies a multi-tier specification model that allows for the integration of ontologies together with rules and Bayesian networks. The described approach enriches the language with a technique where knowledge can be represented as special restrictive rules that may require full or partial satisfaction, which are represented as some sort of special liveness properties. In the paper, the integrated approach has been applied to derive an initial KR structure for the marXbot mobile robotics platform. The hierarchical structure of the ontology is reflected in the tree structure (where additionally the branches are connected through a bounded number of nodes) of the (soft) constraint network. This fact permits the application of efficient dynamic programming solution algorithms.

Among the various facets of autonomicity and adaptivity, one of the issues in ASCEns is their presence in service-oriented applications. The need of modeling scenarios where knowledge is emergent and distributed (meaning that a common ontology is specialized in local, possibly different KR styles and resolution engines) put forward the work on distributed constraint handling, addressing the need of requiring as little centralization as possible, as well as open endedness and heterogeneity of components. According to some preliminary work, as reported in Deliverable 1.2, it seems feasible to encapsulate a variety of knowledge representation styles in different sites, and to carry on deduction steps and consistency checks in terms of distributed constraint handling, where the local, specific knowledge interacts with the constraint representation via suitable interfaces. As a case study, a simple ontology representation and a relational database have been introduced. Furthermore, its implementation has shown how a mix of logic programming (employing the PROVA framework) and a general-purpose programming language such as Java can be used to integrate different knowledge representation formalisms. Constraints can be used as interfaces between heterogeneous knowledge, using Java calls for accessing their implementation.

Novelty. All the different proposals referring to the use of soft constraints for language extensions and knowledge representations that were briefly described here represent novel strands of research. We refer to Deliverables D1.2, D3.2 and D7.2 as well as Joint Deliverables JD2.1 and JD2.2 for a better focused presentation of the different features of these proposals and their integration with the overall aims of ASCEns.

3.3 Strand on "Conceptual models for autonomicity"

While the meaning of "adaptivity" may seem intuitively obvious, various communities use different and incompatible definitions of the term, and surprisingly few generally applicable, precise definitions are available. *A fortiori*, no widely accepted formal model for adaptivity exists.

During the first year we presented [BCG+12] a conceptual notion of "white-box" adaptation, conceived around a prominent role of *control data*: computational data that govern the execution and are conveniently managed to enact adaptive behaviors. White-box adaptation is therefore concerned with how the adaptation process is achieved, not with the environments to which the system can adapt. Building on that methodological proposal, we present a notion of *adaptable transition system*, our attempt to distill a formal model of adaptive systems. Adaptable transition systems are based on foundational models of component based systems (like I/O and interface automata). They can be used to specify properties related to the adaptive behaviour of a system. A central role is again played by control data, as well as by the interaction among components and with the environment.

3.3.1 Formal specification of White-Box Adaptivity

The aim of task 2.2 is to *develop robust mathematical foundations for interaction scenarios that are characterized by highly dynamic, autonomic components, that can update their behavior depending on the current environment,* [...] *react to events and compensate past activities.*

Last year we started addressing this challenging goal by answering easier questions that concern individual autonomic components rather than ensembles: *"When is a software system adaptive?"*, and *"how can we identify the adaptation logic in an adaptive system?"*. To this end, we developed a conceptual framework for white-box adaptation [BCG+12], proposing a simple structural criterion to characterize adaptivity: making explicit that the behavior of a component depends on well identified *control data*, and defining *white-box adaptation* as the *run-time modification of these data*.

White-box perspectives on adaptation thus allow one to inspect the internal structure of a system. They offer a clear separation of concerns to distinguish the cases where the changes of behaviour are part of the application logic from those where they realize the adaptation logic, calling adaptive only systems capable of the latter. A notable example is the Context Oriented Programming paradigm, where the contexts of execution and code variations are first-class citizens that can be used to structure the adaptation logic in a disciplined way [SGP11]. Nevertheless, it is not the programming language what makes a program adaptive or not: truly adaptive systems can be programmed in traditional languages, exactly like object-oriented systems can, with some effort, in traditional imperative languages.

Indeed any computational model or programming language can be used to implement an adaptive system, just by identifying the part of the data that governs the behavior. The nature of control data can thus vary considerably, in the range of all possible ways of encapsulating behavior: from simple configuration parameters to a complete representation of the program in execution that can be modified at run-time, typical of computational models that support meta-programming or reflective features.

This view is agnostic with respect to the form of interaction with the environment, the level of context-awareness, the use of re ection for self-awareness. In fact, our definition applies equally well to most of the existing approaches for designing adaptive systems. Overall, it provides a satisfactory answer to the question "what is adaptation conceptually?". But "what is adaptation formally?" and "which is the right way to reason about adaptation, formally?". Unfortunately, only few works (e.g. [MT02]) address the foundational aspects of adaptive systems, including their semantics and the use of formal reasoning methods, and very often only generic analysis techniques are applied.

Building on the intuitions briefly discussed above and on some foundational models of component based systems (like I/O automata [LT87] and interface automata [dAH01]), and some other developed for similar purposes (e.g. monitored and edit automata [LBW05]), we aimed at distilling a core, essential model of adaptive systems. We proposed a simple formal model based on a new class of transition systems, and we sketched how this definition can be used to specify properties related to the adaptive behaviour of a system. A central role is again played by control data, as well as by the interaction among components and with the enviroment (not addressed explicitly in [BCG$^+$12]).

Novelty and future works. The preliminary work on the novel notion of adatable transition systems, enriching classical I/O automata with an explicit notion of control data. The goal is to propose a formal definition of adaptive systems, as a basis for applying suitable verirfication techniques to components and finally to ensembles.

4 On Task 2.3. Modeling SCEs with collaborative and competitive behavior

The research on this task has the ambition of "develop[ing] a theory combining as much as possible the flexibility and compositionality of computer science semantic models with the expressiveness of models developed by economic theorists", with an in-depth analysis of the "adapt[ion] and re-use in this context many coalgebraic tools well-known for ordinary transition systems".

One side concerns the application of *Game paradigms for service composition*, adopting either non-cooperative job scheduling games, with an interest in abstractly capturing those scenarios where a Nash equilibrium is reached, and minority games for service composition, as needed by e.g. a peer-to-peer energy management scenario. On a more foundational issue, *Coalgebraic techniques for dynamical systems* pushed the coalgebraic view of weighted automata: non-deterministic automata where each transition has also a quantity expressing the cost of its execution.

4.1 Strand on "Game paradigms for service composition"

In [BMT10] a repeated non-cooperative job scheduling game, whose players are Grid sites and whose strategies are scheduling algorithms, have been formulated. We exploited there the concept of Nash equilibrium to express a situation in which no player can gain any profit by unilaterally changing its strategy. We have advanced on the results reported in the first year by giving an exhaustive treatment of the situation that may arise in different contexts. Full details on the final outcome of that research are available in [BMT12], the full version of [BMT10], and in Sonia Taneja's PhD Thesis [Tan12]. The second year advances are summarized in Section 4.1.1.

A fruitful application of adaptation in service component is peer-to-peer energy management, displaying different trading strategies on the energy market. The domestic energy market is changing with the increasing availability of energy generating home-devices, such as solar panels, which provide energy for the house where they are installed as well as enabling the selling of surplus energy. In Sections 4.1.2 and 4.1.3 we report the advances in modelling a scenario where households have the chance to trade energy for purchasing to and for selling from a number of different actors, further including the ability to automatically retrieve data to be used in adaptive and collaborative aspects, such as those affecting the consumption of electricity.

4.1.1 A game-theoretic model of Grid systems

The work on the game-theoretic approach applied to the problem of on-line job scheduling in heterogeneous computational grids, discussed in Deliverable 2.1, has been published [BMT12], enhancing and refining on [BMT10]. The paper analyses potential scenarios where selfish or cooperative behaviours of organizations impact heavily on global Grid efficiency. In order to formalise this problem we formulated a non-cooperative Grid job scheduling game in which players are Grid sites and whose strategies are scheduling algorithms. We exploited the concept of Nash equilibrium to express a situation in which no Grid site can gain any profit by unilaterally changing its strategy. Specifically, we investigated different strategies and we have shown whether and under which circumstances each such strategy is a Nash equilibrium. In the negative case we provided a counter-example, in the positive case we either gave a formal proof or formulated a conjecture which was supported by experimental results obtained through simulations and exhaustive search.

An exhaustive presentation of this research problem and of the solution we proposed appears in Sonia Taneja's PhD thesis [Tan12]. In particular, the thesis gives a more detailed description of the Grid architecture, of the reference Grid system and of the algorithms used to obtain the experimental results. The thesis also includes insights on how to extend our game theoretical model to cloud computing. The results of this work fits well within the context of the Grid architecture proposed by Worldwide LHC Computing Grid (WLCG) which is one of the big global collaboration created for handling the immense amount of data being generated by LHC experiments at European Organization for Nuclear Research (CERN).

Novelty. In the second year we finalized the results developed during months 1-12 on the issue of on-line job scheduling in heterogeneous computational grids. Furthermore, a generalized presentation of this research problem and of the solution we proposed has been tackled.

4.1.2 Smart meter aware domestic energy trading agents

In the first year we designed and developed an agent-based system able to manage the interactions among *prosumers* (producers and consumers of energy, mainly by solar panels and wind turbines), consumers and *genco* (large-scale energy producers). The system relies on an approach coming from game theory; it was tested and simulated, providing good results.

During months 13-24 we proposed a web service integration, in order to enable agents contracting energy to automatically retrieve data to be used in adaptive and collaborative aspects [CAC12]. An example is represented by the retrieval of weather forecasting, which provides input on ongoing demand and data for the predicted availability (in case of photovoltaic or wind powered environments). The challenge lies in how to correctly and autonomously use data coming from different sources, since this information is crucial for user profiling and balancing in the short-term contracts in the Smart Grid.

Moreover, we refined the models evaluating the energy produced by the different kinds of producers, and the model defining the consumption of electricity, which is affected by different factors such as workability (calendar effect), climatic variables, seasonality and economic activity [CCPA12].

Novelty. In the second year we have carried on the work related to peer-to-peer energy management. With respect to the proposal developed during months 1-12, the overall quality of the results has been quite improved: the system relies on more precise models and has more information to adapt its behavior, thus increasing the awareness and the autonomy of the whole system.

4.1.3 Advanced metering infrastructure for energy consumption optimization

One of the challenges ahead the smart power grid is the development of models and systems for the operation of future energy market and for the decision processes of agents that can operate effectively in such markets. In addition, the contribution of the scientific communities linked to economics, computer science and engineering consists also of the systematic analysis of design factors for intelligent trading agents and systematic research, development and testing of human preference and prediction models for intelligent trading agents.

In the PhD thesis of Alain Tcheukam Siwe, under development at IMT Lucca, the main issue is, given a smart power market model, to exploit the different costs of the electricity in various situations by changing the consumption (or production) rate of end users. The contribution is to provide a smart meter (software agent or controller) that is able to interact dynamically with the grid and to minimize both consumption and production energy costs on behalf of prosumers connected to the power grid.

From a prosumer perspective, actual advanced metering infrastructure (shortly AMI) do not provide a way for prosumers to be proactive when interacting with a real time power market. The introduced advanced smart meter is an in-house technology that allowed to manage autonomously the energy production and consumption of a smart building. It represents the interface between the distributed power grid and the local aggregated power production or consumption. It operates by delaying or activating non emergency power production or consumption, given the feedback he has from the dynamic power market. The aim is to maximize profit or to minimize losses when selling or buying energy quantities in the power market.

Novelty and future works. The research on this topic started at UNIPI and IMT during the second year, and it is supposed to be further pushed by interacting with the UNIMORE work.

4.2 Strand on “Coalgebraic techniques for dynamical systems”

Autonomic, and more generally adaptable and reconfigurable systems, are complex objects where aspects like feedback and stability play a key role. Describing and analyzing those aspects calls for a systemic view of systems, where classical models of non-deterministic computation might be integrated with tools from control theory. In this context, *SCE*’s could be described and analyzed as dynamical systems. *Weighted automata*, a generalization of non-deterministic automata [KS86], seem a suitable tool. Recent investigations pointed out that weighted automata are, in a precise sense, a generalization of (discrete) time-invariant linear systems from control theory [Rut07, Bor09].

4.2.1 Advancements on coalgebraic views of Resource Aware operational models

In Deliverable D2.1 it has been argued that systems where feedback and stability play a key role, which include autonomic ones, might be conveniently analyzed by integrating classical models of non-deterministic computation with tools from control theory. *Weighted automata (WA)* [KS86] can play a key role towards achieving this integration. Recall that WA are a generalization of classical non-deterministic automata where each transition, in addition to an input letter, has also a quantity expressing the weight (e.g. cost or probability) of its execution, drawn from a semiring.

A thorough treatment of WA from a coalgebraic perspective is the subject of [BBB+12], which we overview in what follows. The theory of coalgebras offers a unifying mathematical framework for the study of many different types of state-based systems and infinite data structures. Given a functor $G : C \to C$ on a category C, a G-coalgebra is a pair consisting of an object X in C (representing the state space) and a morphism $f : X \to GX$ (determining the dynamics). The theory of coalgebras provides a general notion of behavioural equivalence ($\approx_G$) for G-coalgebras: two states are equivalent if and only if they are mapped to the same element of the, so-called, final coalgebra, when it exists. The first contribution of [BBB+12] is to recast weighted bisimilarity and language equivalence in the theory of coalgebras. WA for a field K and alphabet A are seen as coalgebras of the functor $W = K \times K^{-^A}$ on **Set**. A W-coalgebra consists of a set of states X and a function $(o, t) : X \to K \times K^{X^A}$ where, for each state $x \in X$, $o : X \to K$ assigns an output weight in K and $t : X \to K^{X^A}$ assigns a function in K^{X^A}. For each symbol $a \in A$ and state $x \in X$, $t(x)(a)(x')$ is a semiring element $k \in K$, representing the weight of a transition from x to x' with label a, in symbols $x \overset{a,k}{\rightarrowtail} x'$. If $t(x)(a)(x') = 0$, then there is no a-labelled transition from x to x' . One can similarly model *linear* WA (LWA) as coalgebras of the functor $L = K \times (-)^A$ on **Vect** (the category of vector spaces and linear maps). An LWA consists of a vector space V and a linear map $(o, t) : V \to K \times V^A$ where, as before, $o : V \to K$ defines the output function and $t : V \to V^A$ defines the transition structure. More precisely, for each vector $v \in V$ and $a \in A$, $t(v)(a) = v'$ means that there is a transition from v to v' with label a, in symbols $v \overset{a}{\rightarrowtail} v'$. Note that the transition structure is now deterministic, since for each vector v and input $a \in A$ there is only one vector $v' \in V$ such that $v \overset{a}{\rightarrowtail} v'$. When $V = K^X$, each vector $v \in V$ is a linear combination of states $x_1, \dots, x_n \in X$, i.e., $v = k_1x_1 + + k_nx_n$ for some coefficients $k_1, ..., k_n \in K$. Hence, the transitions $x \overset{a,k_1}{\rightarrowtail} x_1, \dots, x \overset{a,k_n}{\rightarrowtail} x_n$ of a WA correspond to a single transition $x \overset{a}{\rightarrowtail} k_1x_1 + \cdots + k_nx_n$ in a LWA.

We show that $\approx_W$ coincides with weighted bisimulation, and $\approx_L$ with weighted language equivalence. *Determinisation* is the techinque for moving from ordinary WA and weighted bisimilarity to LWA and weighted language equivalence. Unlike the powerset construction in a non-deterministic automaton, the resulting state space will not be finite, rather will form a vector space of finite dimension.

Once we have fixed the mathematical framework, we investigate three different types of algorithms for computing weighted equivalence $\approx_L$ under the assumption that the underlying vector space V has a finite dimension. The first one is a forward algorithm that generalises the usual partition-refinement algorithm for ordinary automata. The second algorithm proceeds in a similar way, but relies on a backward procedure based on the concept of dual space (a concrete version of this algorithm had been introduced in [Bor09].) The advantage of this algorithm over the previous one is that the size of the intermediate relations is typically much smaller. Finally, the third algorithm relies on manipulating syntactical (rational) expressions representing the weighted language recognized by the automaton.

Novelty and future works. In the second year we finalized and generalized the results outlined during the first one on the treatment of WA from a coalgebraic perspective. The ultimate goal is still the development of a comprehensive framework integrating classical models of computation and control theory. Coalgebras and WA have proven useful towards achieving this integration.

A key concept left out of the present analysis is *feedback*. Conceptually, it is not difficult to see systems with feedback as transformers, manipulating streams subjected to certain constraints. Indeed, [Rut07] presents a coalgebraic treatment of discrete linear time invariant systems with explicit input and output streams. The final coalgebra is the set of *causal* transfer functions between streams. A description of feedback is however missing and might well be the next step of our investigation.

5 Concluding remarks and deliverable cross influence

The deliverable reports the advances of the second year on Work Package 2. Its foundational nature resulted in a large amount of contributions in different areas, which were reported adopting a taxonomy based on “strands”, further split in single topics. The advances of each strand have been summarised as the final paragraph of the sections detailing the topics, including some details on future activities that specifically concern each contribution. We try here instead to sketch a tentative workplan for the year to come, as well as to relate the work in each strand with the advances in other work packages.

Foundations of resource-aware connectors The work on connectors is pivotal in order to investigate the features of BIP: its expressiveness and its ability to offer a flexible and comprehensive framework for supporting rigorous system design. This is bound to have consequences on Work Package 5, as it adopts verification techniques for SCEs using BIP (as reported e.g. in Deliverable D5.2). Indeed, BIP lies at the core of the integrated approach for modelling, validating and verifying ensembles proposed in Joint Deliverable JD2.2.

Foundations of resource-aware languages and models The work on resource aware models is still developing. On the one side, the functorial semantics proved in recent years a key innovation, and it has been now put at work also for bigraphical models. As far as the project is concerned, it should be tested with a suitable fragment of SCEL, possibly integrated with the network primitives from NCpi. Part of the work on stochastic calculi should be merged instead with the strand on “Coalgebraic techniques for dynamical systems”, tackling coalgebraic models on weighted automata proposed in Section 2.3, possibly resulting in a satisfactory integrated proposal. As far as other work packages are concerned, stochastic calculi are pivotal in the analysis of case studies, as put e.g. to work on the analysis of swarm robotics systems in Deliverable D7.2 and Joint Deliverable JD2.2.

Negotiate, Commit, Execute mechanisms The current results on the semantics foundation of Sagas as a prototyping language for ensembles are going to be further pursued, with the idea of transferring some of the concepts related to compensation policies in the design of SCEL, along the lines of the extensions proposed in Deliverable D1.2.

Enhancements of conceptual models for autonomicity This topic deals with the extension of the soft constraint approach, in order to make it more flexible and expressive. The specific case of lexicographic ordering has been tackled, but a more general model might be sought, covering more instances, and additional operators should be looked after. The relevance of this endeavour is witnessed by the final paragraph of the section which is rounded up with a general overview of the uses of the soft constraint paradigm in different work packages. More precisely, they are reported on Deliverables D1.2 (about ccSCEL), D3.2 (about SCKL, integrating soft constraints in KnowLang as a KR technique), and D7.2. (about the use of soft CLP for the e-mobility optimization problem), and on Joint Deliverables JD2.1 (ccSCEL and SCKL) and JD2.2 (SCLP).

Conceptual models for autonomicity The white-box approach presented in Deliverable D1.2 has been fostering a tight connection with Work Package 4. As a start, it helped focusing the complementary black-box approach, thus helping with the integration of SOTA and GEM (now reported in Join Deliverable JD2.1). Most importantly, an on-going research is focusing on the formalisation of an architectural model for self-adaptive patterns. The current work on transition systems should be strengthened in order to prove its effectiveness as a specification formalism and its suitability for verification, as well as to suggest suitable extensions for the operational models of SCEL.

References

[AB09] Andrea Asperti and Nadia Busi. Mobile petri nets. *Mathematical Structures in Computer Science*, 19(6):1265–1278, 2009.

[ABC$^+$09] Farhad Arbab, Roberto Bruni, Dave Clarke, Ivan Lanese, and Ugo Montanari. Tiles for Reo. In Andrea Corradini and Ugo Montanari, editors, *WADT*, volume 5486 of *Lect. Notes Comput. Sci.*, pages 37–55. Springer, 2009.

[Bag89] R. Bagrodia. Process synchronization: Design and performance evaluation of distributed algorithms. *IEEE Transactions on Software Engineering (TSE)*, 15(9):1053–1065, 1989.

[BBB$^+$12] Filippo Bonchi, Marcello Bonsangue, Michele Boreale, Jan Rutten, and Alexandra Silva. A coalgebraic perspective on linear weighted automata. *Information and Computation*, 211:77–105, 2012.

[BBBS08] A. Basu, P. Bidinger, M. Bozga, and J. Sifakis. Distributed semantics and implementation for systems with interaction and priority. In *Formal Techniques for Networked and Distributed Systems (FORTE)*, pages 116–133, 2008.

[BBF$^+$05] Roberto Bruni, Michael J. Butler, Carla Ferreira, C. A. R. Hoare, Hernán C. Melgratti, and Ugo Montanari. Comparing two approaches to compensable flow composition. In Martín Abadi and Luca de Alfaro, editors, *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 383–397. Springer, 2005.

[BBJ$^+$10] B. Bonakdarpour, M. Bozga, M. Jaber, J. Quilbeuf, and J. Sifakis. From high-level component-based models to distributed implementations. In *EMSOFT*, pages 209–218, 2010.

[BBJ$^+$12] B. Bonakdarpour, M. Bozga, M. Jaber, J. Quilbeuf, and J. Sifakis. A framework for automated distributed implementation of component-based models. *Distributed Computing*, 2012. to appear.

[BBPS11] Ananda Basu, Saddek Bensalem, Doron Peled, and Joseph Sifakis. Priority scheduling of distributed systems based on model checking. *Formal Methods in System Design*, 39(3):229–245, 2011.

[BBQS12] S. Bensalem, M. Bozga, J. Quilbeuf, and J. Sifakis. Knowledge-based distributed conflict resolution for multiparty interactions and priorities. In *FMOODS/FORTE*, pages 118–134, 2012.

[BBS06] Ananda Basu, Marius Bozga, and Joseph Sifakis. Modeling heterogeneous real-time components in BIP. In *SEFM*, pages 3–12. IEEE Computer Society, 2006.

[BCG$^+$12] Roberto Bruni, Andrea Corradini, Fabio Gadducci, Alberto Lluch Lafuente, and Andrea Vandin. A conceptual framework for adaptation. In J. de Lara and A. Zisman, editors, *FASE 2012*, volume 7212 of *LNCS*, pages 240–254. Springer, 2012.

[BDL10] M. Bernardo, R. De Nicola, and M. Loreti. Uniform Labeled Transition Systems for Nondeterministic, Probabilistic, and Stochastic Processes. In Wirsing, M. and Hofmann, M. and Rauschmayer, A. , editor, *Trustworthy Gobal Computing - TGC 2010*, page 3556, 2010. LNCS 6084.

[BFK12] Roberto Bruni, Carla Ferreira, and Anne Kersten Kauer. First-order dynamic logic for compensable processes. In Marjan Sirjani, editor, *COORDINATION*, volume 7274 of *Lecture Notes in Computer Science*, pages 104–121. Springer, 2012.

[BG06a] S. Bistarelli and F. Gadducci. Enhancing constraints manipulation in semiring-based formalisms. In *ECAI 2006*, volume 141 of *FAIA*, pages 63–67. IOS, 2006.

[BG06b] M. Boreale and F. Gadducci. Processes as formal power series: A coinductive approach to denotational semantics. *Theor. Comput. Sci.*, 360(1-3):440–458, 2006.

[BK12] Roberto Bruni and Anne Kersten Kauer. Lts semantics for compensation-based processes. In *TGC 2012*, 2012. To appear.

[BKLS10] Roberto Bruni, Anne Kersten, Ivan Lanese, and Giorgio Spagnolo. A new strategy for distributed compensations with interruption in long-running transactions. In Till Mossakowski and Hans-Jörg Kreowski, editors, *WADT*, volume 7137 of *Lecture Notes in Computer Science*, pages 42–60. Springer, 2010.

[BKM06] Filippo Bonchi, Barbara Konig, and Ugo Montanari. Saturated semantics for reactive systems. In *Proceedings of the 21st Annual IEEE Symposium on Logic in Computer Science*, pages 69–80. IEEE Computer Society, 2006.

[BLM06] Roberto Bruni, Ivan Lanese, and Ugo Montanari. A basic algebra of stateless connectors. *Theor. Comput. Sci.*, 366(1-2):98–120, 2006.

[BM99] Roberto Bruni and Ugo Montanari. Cartesian closed double categories, their lambda-notation, and the pi-calculus. In *LICS 1999*, pages 246–265. IEEE Computer Society, 1999.

[BM07] Maria Grazia Buscemi and Ugo Montanari. Cc-pi: A constraint-based language for specifying service level agreements. In *ESOP 2007*, volume 4421 of *LNCS*, pages 18–32. Springer, 2007.

[BMM02] Roberto Bruni, José Meseguer, and Ugo Montanari. Symmetric monoidal and cartesian double categories as a semantic framework for tile logic. *Mathematical Structures in Computer Science*, 12(1):53–90, 2002.

[BMM05] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Theoretical foundations for compensations in flow composition languages. In Jens Palsberg and Martín Abadi, editors, *POPL*, pages 209–220. ACM, 2005.

[BMM11a] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. A connector algebra for p/t nets interactions. In Joost-Pieter Katoen and Barbara König, editors, *CONCUR 2011*, volume 6901 of *Lecture Notes in Computer Science*, pages 312–326. Springer, 2011.

[BMM11b] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Connector algebras, petri nets, and bip. In Edmund M. Clarke, Irina Virbitskaite, and Andrei Voronkov, editors, *Ershov Memorial Conference 2011*, volume 7162 of *Lecture Notes in Computer Science*, pages 19–38. Springer, 2011.

[BMR97] S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint solving and optimization. *Journal of ACM*, 44(2):201–236, 1997.

[BMR01a] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Semiring-based contstraint logic programming: syntax and semantics. *ACM Trans. Program. Lang. Syst.*, 23(1):1–29, 2001.

[BMR01b] Roberto Bruni, Ugo Montanari, and Francesca Rossi. An interactive semantics of logic programming. *CoRR*, cs.LO/0107022, 2001.

[BMR06] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Soft concurrent constraint programming. *ACM Trans. Comput. Log.*, 7(3):563–589, 2006.

[BMT10] Maria Grazia Buscemi, Ugo Montanari, and Sonia Taneja. Toward a game-theoretic model of grid systems. In Martin Wirsing, Martin Hofmann, and Axel Rauschmayer, editors, *TGC*, volume 6084 of *Lect. Notes Comput. Sci.*, pages 57–72. Springer, 2010.

[BMT12] Maria Grazia Buscemi, Ugo Montanari, and Sonia Taneja. Analysis of grid job scheduling. *Journal of Grid Computing*, 10:501–519, 2012.

[Bor09] Michele Boreale. Weighted bisimulation in linear algebraic form. In *CONCUR*, volume 5710 of *Lecture Notes in Computer Science*, pages 163–177. Springer, 2009.

[BPRV11] Stefano Bistarelli, Maria Silvia Pini, Francesca Rossi, and Kristen Brent Venable. Uncertainty in bipolar preference problems. *J. Exp. Theor. Artif. Intell.*, 23(4):545–575, 2011.

[BPS10] Saddek Bensalem, Doron Peled, and Joseph Sifakis. Knowledge based scheduling of distributed systems. In *Essays in Memory of Amir Pnueli*, pages 26–41, 2010.

[Bru99] Roberto Bruni. *Tile Logic for Synchronized Rewriting of Concurrent Systems*. PhD thesis, Computer Science Department, University of Pisa, 1999.

[BS01] Maria Grazia Buscemi and Vladimiro Sassone. High-level petri nets as type theories in the join calculus. In Furio Honsell and Marino Miculan, editors, *FoSSaCS*, volume 2030 of *Lecture Notes in Computer Science*, pages 104–120. Springer, 2001.

[Buc94] P. Buchholz. Exact and ordinary lumpability in finite Markov chains. *J. Applied Probability*, 31:59–74, 1994.

[CAC12] Nicola Capodieci, Emanuel Alsina, and Giacomo Cabri. A context-aware agent-based approach for deregulated energy market. In *Proceedings of the 21st IEEE WETICE conference Toulouse, France, June 2012*, 2012.

[CCPA12] Nicola Capodieci, Giacomo Cabri, Andrea Pagani, and Marco Aiello. An agent-based application to enable deregulated energy markets. In *Proceedings of the 36th IEEE Computer Software and Applications Conference (COMPSAC 2012), Izmir Turkey, July 2012*, 2012.

[CFV09] L. Caires, C. Ferreira, and H. Vieira. A process calculus analysis of compensations. In *TGC'08*, volume 5474 of *LNCS*, pages 87–103. Springer, 2009.

[CM84] K. M. Chandy and J. Misra. The drinking philosophers problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 6(4):632–646, 1984.

[CM88] K. M. Chandy and J. Misra. *Parallel program design: a foundation*. Addison-Wesley Longman Publishing Co., Inc., 1988.

[Coo03] M.C. Cooper. Reduction operations in fuzzy or valued constraint satisfaction. *Fuzzy Sets and Systems*, 134(3):311–342, 2003.

[dAH01] L. de Alfaro and T.A. Henzinger. Interface automata. In *ESEC/SIGSOFT FSE 2001*, volume 26(5) of *ACM SIGSOFT Software Engineering Notes*, pages 109–120. ACM, 2001.

[FM00] Gian Luigi Ferrari and Ugo Montanari. Tile formats for located and mobile systems. *Inf. Comput.*, 156(1-2):173–235, 2000.

[FT01] Marcelo P. Fiore and Daniele Turi. Semantics of name and value passing. In *LICS*, pages 93–104, 2001.

[GCS$^+$08] A. Gutiérrez, A. Campo, F.C. Santos, C. Pinciroli, and M. Dorigo. Social odometry in populations of autonomous robots. In *ANTS 2008*, volume xxx of *LNCS*, pages 371–378. Springer, 2008.

[GM00] Fabio Gadducci and Ugo Montanari. The tile model. In Gordon D. Plotkin, Colin Stirling, and Mads Tofte, editors, *Proof, Language, and Interaction*, pages 133–166. The MIT Press, 2000.

[GM02] Fabio Gadducci and Ugo Montanari. Comparing logics for rewriting: rewriting logic, action calculi and tile logic. *Theor. Comput. Sci.*, 285(2):319–358, 2002.

[Her02] H. Hermanns. *Interactive Markov Chains*. Springer, Berlin/New York, 2002. LNCS 2428.

[HHM98] H. Hermanns, U. Herzog, and V. Mertsiotakis. Stochastic process algebras — between LOTOS and Markov chains. *Computer Networks and ISDN Systems*, 30(9–10):901 – 924, 1998.

[Hil96a] J. Hillston. A compositional approach to performance modelling, 1996. Distinguished Dissertation in Computer Science. Cambridge University Press.

[Hil96b] J. Hillston. *A compositional approach to performance modelling*. Cambridge University Press, New York, NY, USA, 1996.

[Hil05] J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of Quantitative Evaluation of Systems*, pages 33–43. IEEE Computer Society Press, 2005.

[HJS07] I. Hasuo, B. Jacobs, and A. Sokolova. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3(4), 2007.

[HMW09] M. Hoelzl, M. Meier, and M. Wirsing. Which soft constraints do you prefer. In *WRLA 2008*, volume 283(2) of *ENTCS*, pages 189–205. Elsevier, 2009.

[JL87] Joxan Jaffar and Jean-Louis Lassez. Constraint logic programming. In *POPL 1987*, pages 111–119. ACM Press, 1987.

[KM01] Barbara König and Ugo Montanari. Observational equivalence for synchronized graph rewriting with mobility. In Naoki Kobayashi and Benjamin C. Pierce, editors, *TACS 2001*, volume 2215 of *Lecture Notes in Computer Science*, pages 145–164. Springer, 2001.

[KS86] Werner Kuich and Arto Salomaa. *Semirings, Automata, Languages*, volume 5 of *Monographs in Theoretical Computer Science. An EATCS Series*. Springer, 1986.

[KZBS12] N. Koch, K. Zemmer, B.Werther, and R.Y. Siegwarty. Electric vehicle travel optimization-customer satisfaction despite resource constraints. In *IVS 2012*. IEEE, 2012.

[LBW05] J. Ligatti, L. Bauer, and D. Walker. Edit automata: enforcement mechanisms for runtime security policies. *Intl. Journal of Information Security*, 4(1-2):2–16, 2005.

[LMdV12] D. Latella, M. Massink, and E.P. de Vink. Bisimulation of labeled state-to-function transition systems of stochastic process languages. In Ulrike Golas and Thomas Soboll, editors, Proceedings Seventh ACCAT Workshop on *Applied and Computational Category Theory* , Tallinn, Estonia, 01/04/2012, volume 93 of *Electronic Proceedings in Theoretical Computer Science*, pages 23–43. Open Publishing Association, 2012.

[LS03] Javier Larrosa and Thomas Schiex. In the quest of the best form of local consistency for Weighted CSP. In Georg Gottlob and Toby Walsh, editors, *IJCAI 2003*, pages 239–244. Morgan Kaufmann, 2003.

[LT87] N.A. Lynch and M.R. Tuttle. Hierarchical correctness proofs for distributed algorithms. In *PODC 1987*, pages 137–151. ACM, 1987.

[MM12] G. V. Monreale and U. Montanari. Soft constraint logic programming for electric vehicle travel optimization. In *WLP 2012*, 2012.

[MP05] Ugo Montanari and Marco Pistore. Structured coalgebras and minimal hd-automata for the *pi*-calculus. *Theor. Comput. Sci.*, 340(3):539–576, 2005.

[MR99] Ugo Montanari and Francesca Rossi. Graph rewriting, constraint solving and tiles for coordinating distributed systems. *Applied Categorical Structures*, 7(4):333–370, 1999.

[MS12a] Ugo Montanari and Matteo Sammartino. Network conscious pi-calculus. Technical Report TR-12-01, Computer Science Department, University of Pisa, February 2012.

[MS12b] Ugo Montanari and Matteo Sammartino. Network conscious pi-calculus: a concurrent semantics. In *Proc. MFPS 2012*, 2012. to appear, available at `http://www.di.unipi.it/~sammarti/papers/mfps28.pdf`.

[MT02] José Meseguer and Carolyn L. Talcott. Semantic models for distributed object reflection. In Boris Magnusson, editor, *ECOOP*, volume 2374 of *Lect. Notes Comput. Sci.*, pages 1–36. Springer, 2002.

[MV12] Ugo Montanari and Emil Vassev. Soft constraints for knowlang. In *FMSAS 2012*. ACM Press, 2012.

[PCT04] J. A. Pérez, R. Corchuelo, and M. Toro. An order-based algorithm for implementing multiparty synchronization. *Concurrency and Computation: Practice and Experience*, 16(12):1173–1206, 2004.

[Pin98] J.-E. Pin. Tropical semirings. In *Idempotency*, pages 50–69. Cambridge University Press, 1998.

[PS92] J. Parrow and P. Sjödin. Multiway synchronizaton verified with coupled simulation. In *International Conference on Concurrency Theory (CONCUR)*, pages 518–533, 1992.

[PW92] Dewayne E. Perry and Alexander L. Wolf. Foundations for the study of software architecture. *ACM SIGSOFT Softw. Eng. Notes*, 17:40–52, 1992.

[Rut07] Jan J. M. M. Rutten. Coalgebraic foundations of linear systems. In *CALCO*, volume 4624 of *Lecture Notes in Computer Science*, pages 425–446. Springer, 2007.

[Sar93] Vijay A. Saraswat. *Concurrent constraint programming*. ACM Doctoral dissertation awards. MIT Press, 1993.

[SFV95] T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems: Hard and easy problems. In *IJCAI 1995*, pages 631–637. Morgan Kaufman, 1995.

[SGP11] Guido Salvaneschi, Carlo Ghezzi, and Matteo Pradella. Context-oriented programming: A programming paradigm for autonomic systems. *CoRR*, abs/1105.0069, 2011.

[Sob10] Pawel Sobociński. Representations of Petri net interactions. In Paul Gastin and François Laroussinie, editors, *CONCUR*, volume 6269 of *Lect. Notes Comput. Sci.*, pages 554–568. Springer, 2010.

[Tan12] Sonia Taneja. *A game-theoretical analysis of Grid job scheduling*. PhD thesis, IMT Lucca, 2012. Available at http://e-theses.imtlucca.it/84/.

[TT12] M. Tschaikowski and M. Tribastone. Exact fluid lumpability for Markovian process algebra. In *23rd International Conference on Concurrency Theory*, September 2012. To appear.

[VF10] C. Vaz and C. Ferreira. Towards compensation correctness in interactive systems. In *WS-FM'09*, volume 6194 of *LNCS*, pages 161–177. Springer, 2010.