INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team distribcom

Distributed Models and Algorithms for the Management of Telecommunication Systems

Rennes

THEME COM

Activity Report

2006

Table of contents

1. Team

Head of project–team

Albert Benveniste [DR INRIA, part–time, HdR]
Claude Jard [PROF. ENS CACHAN , HdR]

Administrative assistant

Myriam David [ending June 1st]
Diana Gaudin [ending July 1st]
Laurence Dinh [starting July 1st]
Sara Freoul [starting September 25th, ending October 25th]

Research scientist (inria)

Éric Fabre [CR]
Stefan Haar [CR]
Loïc Hélouët [CR]

Research scientist (cnrs)

Blaise Genest [CR]

Faculty member (ENS Cachan)

Anne Bouillard [teaching assistant, starting September 1st]

Technical staff (inria)

Guy-Bertrand Kamga [ingénieur expert, ending May 1st]
Franck Wielgus [ingénieur expert, ending March 15th]

Guests

Guy-Vincent Jourdan [Ottawa University, from May 10th to July 10th]

Post–doctoral fellow (inria)

Il-Gon Kim [ASAX Post-Doc, ending November 15th]
Agnes Madalinski [Post-Doc, ending December 31st]
Shaofa Yang [Starting september 10th]

Ph.D. student

Thomas Chatain [INRIA grant, ending August 31st]
Thomas Gazagnaire [Normalien]
Emmanuel Donin de Rosière [CIFRE grant, FT R&D, ending September 30th]
Debmalya Biswas [INRIA grant]
Hélia Poullyau [INRIA grant]
Sidney Rosario [INRIA grant, starting July 1st]
Didier Devaurs [MENRT fellowship, starting October 1st]

2. Overall Objectives

2.1. Overall Objectives

Keywords: *Orchestrations*, *Quality of Service*, *Web services*, *distributed algorithms*, *distributed testing*, *fault management*, *self-management*, *telecommunications*.

The DistribCom team addresses distributed algorithms for network and service management, and the management of Web services. More precisely, the overall focus of DistribCom is on *algorithms for distributed management.*

Today, research on network and service management as well as Web Services mainly focuses on issues of software architecture and infrastructure deployment. However, these areas also involve algorithmic problems such as fault diagnosis and alarm correlation, testing, QoS evaluation, and negotiation for QoS. DistribCom develops the foundations supporting such algorithms, namely: fundamentals of distributed observation and supervision of systems involving concurrency. Our algorithms are model-based. For obvious reasons of complexity, such models cannot be built by hand. Therefore we also address the novel topic of *self-modeling,* i.e., the automatic construction of models, both structural and behavioral.

Our research topics are currently structured as follows:

1. fundamentals of distributed observation and supervision of concurrent systems;
2. self-modeling;
3. algorithms for distributed management of telecommunications systems and services;
4. Web Services orchestrations, functional and QoS aspects;
5. Active XML peers for Web scale data and workflow management.

Our main industrial ties are with Alcatel and France-Telecom, on the topic of networks and service management.

Anne Bouillard was recruited at ENS Cachan this year, and joined our team only this fall. Therefore, although she is very welcome, her activities are not included here.

3. Scientific Foundations

3.1. Overview of the needed paradigms

Keywords: *asynchronous system*, *distributed system*, *monitoring*, *quality of service*, *testing*.

Management of telecommunications networks and services, and Web services, involves the following algorithmic tasks:

Observing, monitoring, and testing large distributed systems: Alarm or message correlation is one of the five basic tasks in network and service management. It consists in causally relating the various alarms collected throughout the considered infrastructure—be it a network or a service sitting on top of a transport infrastructure. Fault management requires in particular reconstructing the set of all state histories that can explain a given log of observations. Testing amounts to understanding and analyzing the responses of a network or service to a given set of stimuli; stimuli are generally selected according to given test purposes. All these are variants of the general problem of *observing* a network or service. Networks and services are large distributed systems, and we aim at observing them in a distributed way as well, namely: logs are collected in a distributed way and observation is performed by a distributed set of supervising peers.

Quality of Service (QoS) evaluation, negotiation, and monitoring: QoS issues are a well established topic for single domain networks or services, for various protocols — e.g., Diffserv for IP. Performance evaluation techniques are used that follow a “closed world” point of view: the modeling involves the overall traffic, and resource characteristics are assumed known. These approaches extend to some telecommunication services as well, e.g., when considering (G)MPLS over an IP network layer.

However, for higher level applications, including composite Web services (also called *orchestrations*, this approach to QoS is no longer valid. For instance, an orchestration using other Web services has no knowledge of how many users are calling the same Web services. In addition, it has no knowledge of the transport resources it is using. Therefore, the well developed “closed world” approach can no longer be used. *Contract* based approaches are considered instead, in which a given orchestration offers promises to its users on the basis of promises it has from its subcontracting services. In this context, contract composition becomes a central issue. Monitoring is needed to check for possible breaching of the contract. Coutermeasures would consist in reconfigurating the orchestration by replacing the failed subcontracted services by alternative ones.

The DistribCom team focuses on the algorithms supporting the above tasks. Therefore models providing an adequate framework are fundamental. We focus on models of discrete systems, not models of streams or fluid types of models. And we address the distributed and asynchronous nature of the underlying systems by using models involving only local, not global, states, and local, not global, time. These models are reviewed in section 3.2.

We use these mathematical models to support our algorithms and we use them also to study and develop formalisms of Web services orchestrations and workflow management in a more general setting.

3.2. Models of concurrency: nets, scenarios, event structures, graph grammars, and their variants

Keywords: *Models of concurrency*, *event structures*, *graph grammars*, *nets*, *scenarios*.

For Finite State Machines (FSM), a large body of theory has been developed to address problems such as: observation (the inference of hidden state trajectories from incomplete observations), control, diagnosis, and learning. These are difficult problems, even for simple models such as FSM's. One of the research tracks of DistribCom consists in extending such theories to distributed systems involving concurrency, i.e., systems in which both time and states are local, not global. For such systems, even very basic concepts such as "trajectories" or "executions" need to be deeply revisited. Computer scientists have for a long time recognized this topic of concurrent and distributed systems as a central one. In this section, we briefly introduce the reader to the models of scenarios, event structures, nets, languages of scenarios, graph grammars, and their variants.

3.2.1. Scenarios.

The simplest concept related to concurrency is that of a finite execution of a distributed machine. The *scenario* shown in Figure 1

is an example. The figure shows the life-time (from top to bottom) of four processes (or instances). The instance can exchange asynchronous messages. In this example, some local variables can be tested and assigned. In this model, events are totally ordered for each instance, but only partially ordered between different instances. Thus, time is local, not global. The natural concept of state is local too (i.e., attached to individual instances). Global states can be defined, they however require nontrivial algorithms for their distributed construction. Finite scenarios introduce the two key concepts of *causality* and *concurrency* [6]. The causality relation is a partial order, we denote it by $\preceq$. In Figure 1, the reception of AU_AIS is causally related to the sending of MS_AIS by the rs_TTP, while it is concurrent with the receipt of MS_AIS by the alarm manager.

Scenarios have been informally used by telecom engineers for a long time. Their formalization was introduced by the work done in the framework of ITU and OMG on High-level Message Sequence Charts and on UML Sequence Diagrams in the last ten years, see [57], [64]. This allowed in particular to formally define infinite scenarios, and to enhance them with variables, guards, etc [68], [66], [67]. Today, scenarios are routinely offered by UML and related systems and software modeling tools, Figure 1 showed such an example.

3.2.2. Event structures.

The next step is to model sets of finite executions of a distributed machine. *Event structures* were invented by Glynn Winskel and co-authors in 1980 [63], [73]. This data structure collects all the executions by superimposing shared prefixes. Figure 2 shows an example.

The top most diagram shows an HMSC. i.e., an automaton whose transitions are labeled by basic scenarios. Consider first the scenarios as abstract labels. The set of all executions of this automaton is then shown on the bottom left diagram, in the form of an execution tree. For sequential machines, executions trees collect all the executions by superimposing shared prefixes.

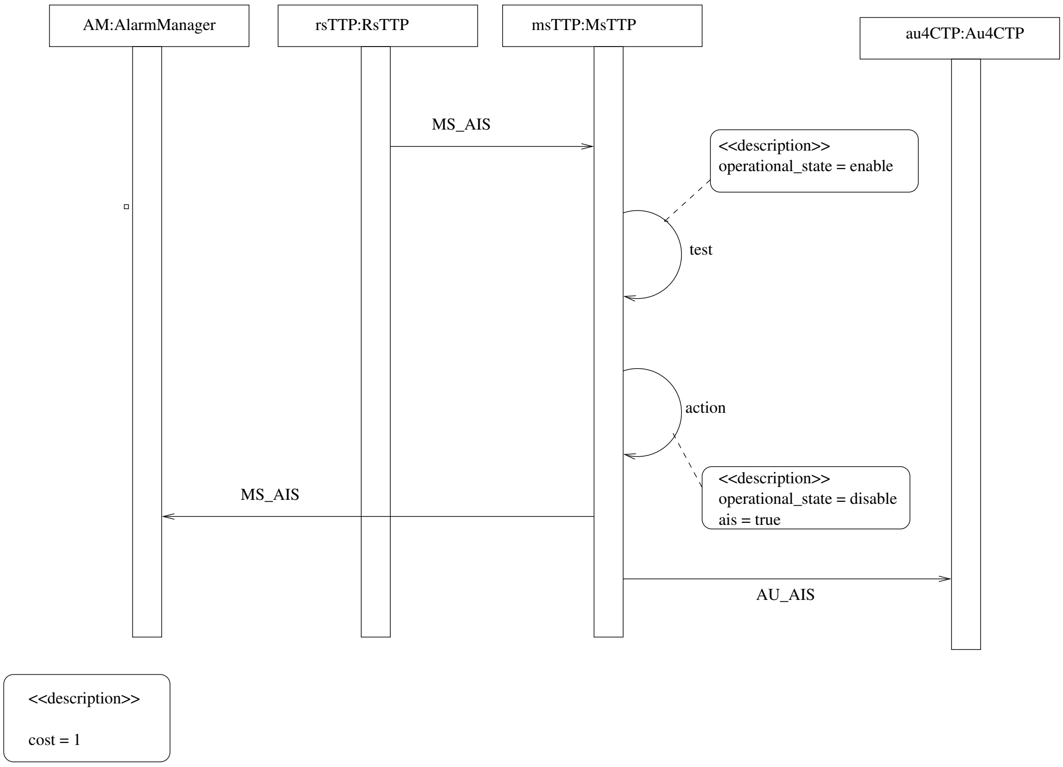

Figure 1. A basic scenario, drawn as a UML sequence diagram.

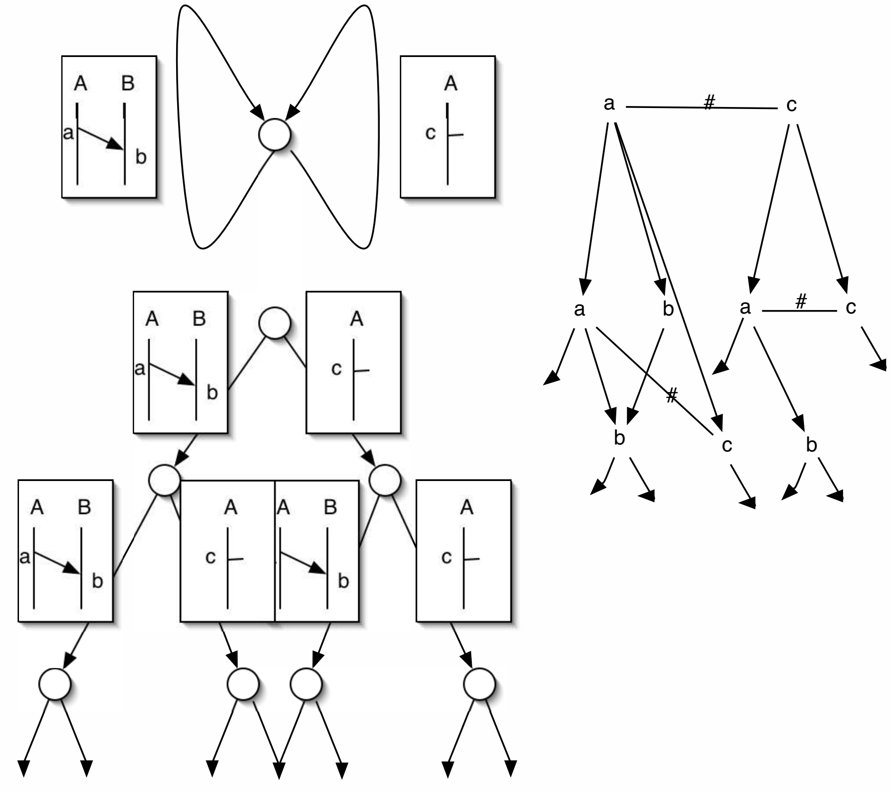

Figure 2. An HMSC (an automaton whose transitions are labeled by scenarios), its execution tree, and its event structure.

Now, the right diagram shows the "white box" version of the former, in which the concatenation of the successive basic scenarios has been performed by chaining them instance by instance. The result is a *event structure,* i.e., a branching structure consisting of events related by a *causality* relation (depicted by directed arrows) and a *conflict* relation (depicted by a non directed arc labeled by a $\#$ symbol). Events that are neither causally related nor in conflict are called *concurrent.* Concurrent processes model the "parallel progress" of components.

Categories of event structures have been defined, with associated morphisms, products, and co-products, see [74]. Products and co-products formalize the concepts of parallel composition and "union" of event structures, respectively. This provides the needed apparatus for composing and projecting (or abstracting) systems.

Event structures have been mostly used to give the semantics of various formalisms or languages, such as Petri nets, CCS, CSP, etc [63], [73]. We in DistribCom make a nonstandard use of these, e.g., we use them as a structure to compute and express the solutions of observation or diagnosis problems, for concurrent systems.

3.2.3. Nets and languages of scenarios.

The next step is to have finite representations of systems having possibly infinite executions. In DistribCom, we use two such formalisms: *Petri nets* [65], [48] and *languages of scenarios* such as High-level Message Sequence Charts (HMSC) [57], [67]. Petri nets are well known, at least in their basic form, we do not introduce them here. We use so-called *safe* Petri Nets, in which markings are boolean (tokens can be either 0 or 1); and we use also variants, see below. Languages of scenarios are simply obtained as illustrated in Figure 2: 1/ equip basic scenarios with a concatenation operation, and 2/ consider an automaton whose transitions are labeled with basic scenarios. Executions of Petri Nets and HMSC can be represented with concurrency in the form of event structures. We have shown this for HMSC's in Figure 2, and it is obtained in a similar way for Petri nets.

3.2.4. Extensions and variants.

Two extensions of the basic concepts of nets or scenario languages are useful for us:

- Nets or scenario languages enriched with variables, actions, and guards. This is useful to model general concurrent and distributed dynamical systems in which a certain discrete abstraction of the control is represented by means of a net or a scenario language. Manipulating such *symbolic nets* requires using abstraction techniques. Time Petri nets and network of timed automata are particular cases of symbolic nets.
- Probabilistic Nets or event structures. Whereas a huge literature exists on stochastic Petri nets or stochastic process algebras (in computer science), randomizing *concurrent models,* i.e., with ω's being concurrent trajectories, not sequential ones, has been addressed only since the 21st century. We have contributed to this new area of research.

3.2.5. Handling dynamic changes in the systems: graph grammars.

The last and perhaps most important issue, for our applications, is the handling of dynamic changes in the systems model. This is motivated by the constant use of dynamic reconfigurations in management systems. Extensions of net models have been proposed to capture this, for example the *dynamic nets* of Vladimiro Sassone [47]; for the moment, such models lack a suitable theory of unfoldings. A relevant alternative is the class of *graph grammars* [40], [46]. Graph grammars transform graphs by means of a finite set of rules.

Figures 3 shows how graph grammar rules can code Petri net behavior, and express modifications of the net structure.

Graph grammars have been equipped with a rich theory of unfoldings, and more generally received much attention from a theoretical viewpoint. While there are numerous modeling applications in Biology, Chemistry, computer science, etc, we at DistribCom test their use for distributed network management algorithms for systems subject to reconfiguration.

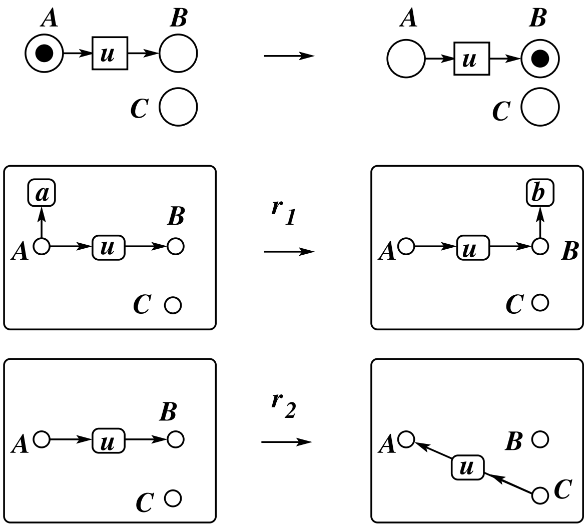

Figure 3. Top : a Petri net transition firing; center: a Graph grammar simulating that firing; bottom: change of net structure

4. Application Domains

4.1. Network and service management in telecommunications

Keywords: *QoS monitoring and negotiation*, *failure diagnosis*, *telecommunication transport network*.

Telecommunications have grown up from a basic technology of networks and transport to a much more complex jungle of networks, services, and applications. This motivates a strong research effort towards “autonomic communications”: one would like to program networks at the service level (sometimes called the business level, since it directly involves contracts with customers), and let the (possibly cross-domain) infrastructure adapt itself in order to ensure a given QoS, isolate and repair failures, etc. This tendency appears for example in the policy-based management, or in the quest for “self-XX” functionalities (self-configuration, self-monitoring, self-healing, etc.). One of our objectives is to address these subjects.

These problems have several common features. First of all, they involve concurrent systems, *i.e.* systems where several things can happen independently at the same time. Secondly, they are built in a modular way, by combining elementary components into large connected structures. Third common point: these systems exhibit dynamicity. Reconfigurations, connections/deconnections of new components or clients are part of the normal activity, and should not require that monitoring algorithms be reset or modified each time the system is changed. And finally, the size and heterogeneity of these systems prevents from using a centralized monitoring architecture. This motivates developing distributed and modular approaches.

As an example of distributed monitoring, our first application was related to diagnosis issues in transport networks, i.e. the low-level layers of networks (physical, transport and network layers). We have focused on circuit oriented networks, such as SDH/WDM protocols or GMPLS protocols. These systems assemble hundreds of functions of components, and the failure of one of them generally induces side-effects in many others. This phenomenon is known as “fault propagation;” it results in hundreds of alarms produced by the various components and collected at different locations in the network. Identifying origins of faults from these alarms has now reached a level of complexity that prevents the traditional human analysis of alarms. Due to the size of systems, the automatic diagnosis task cannot be done in a centralized manner, and must be solved by a network of local supervisors that coordinate their work to provide a coherent view of what happened in the network.

Currently we are addressing issues concerning dynamic services in heterogeneous networks and services. The emphazis is on guaranteeing desired levels of QoS in situations where SLAs have to be negotiated, instantiated, and monitored in a non-local fashion: the immediate peer-to-peer contact allowing for negotiation, monitoring and penalties, is between a client and a provider, or two neighboring providers only.

4.2. Web services and workflow management

Keywords: *Active XML*, *QoS*, *Web services*, *choreographies*, *orchestrations*.

In higher layers, at the level of services, the same search for flexibility motivates the development of tools to rapidly assemble Web-services into larger services, called *orchestrations* or *choreographies*. Recently, standard languages for service workflow have even been proposed such as IBM's Web Services Flow Language [41] or Microsoft's XLang [42], which converged to the BPEL4WS proposal [39] and subsequently WSCDL proposal [58] for choreographies. Tools for BPEL are, among others, commercially available from Telelogic[1] and Oracle[2].

The implementation of orchestration and choreography description languages raises a number of difficulties related to efficiency, clean semantics, and reproducibility of executions, that are impairing their industrial acceptance. In addition, issues of composite QoS associated with orchestrations are not a mature area [75]. We develop studies in these areas.

A serious shortcoming of approaches to Web Service orchestration and choreography is that they mostly abstract data away. Symmetrically, modern approaches to Web data management [55], [76] typically based on XML and Xqueries rely on too simplistic forms of control. We believe that time has come for a convergence of sophistication in terms of control and richness in data, for workflow and data management over the Web. We believe that *active Peer-to-Peer XML-based documents*, as proposed by S. Abiteboul under the name of AXML [3] provide the basis for an adequate infrastructure for this. We cooperate with S. Abiteboul and A. Muscholl at lifting AXML to handle workflow management as well, thus providing a document oriented alternative to Web services orchestrations and choreographies.

5. Software

5.1. SOFAT : a scenario toolbox

Keywords: *partial orders*, *scenarios*.

Participant: Loïc Hélouët.

The SOFAT toolbox is a scenario manipulation toolbox. Its aim is to implement all known formal manipulations on scenarios. The toolbox implements several formal models such as partial orders, graph grammars, graphs, and algorithm dedicated to these models (Tarjan, cycle detection for graphs, Caucal's normalization for graph grammars, etc.). The SOFAT toolbox is permanently updated to integrate new algorithms. It is currently used for a research contract with France Telecom, and is freely available form INRIA's website. The last update of SOFAT includes the fibered product operation described in [59]. This year SOFAT was used in the CO2 project. A connection to a performance evaluation toolchain in the RNRT project PERSIFORM is also under study.

[1] http://www.telelogic.com/standards/bpel.cfm
[2] http://www.oracle.com/appserver/bpel_home.html
[3] http://activexml.net

6. New Results

6.1. User oriented modeling

Keywords: *Automatic abstraction*, *Self-modeling*, *learning*, *scenario weaving*, *testing*.

Participants: Claude Jard, Eric Fabre, Loïc Hélouët, Blaise Genest, Thomas Gazagnaire, Franck Wielgus, Guy-Bertrand Kamga.

The emerging topic of *self-modeling* addresses the automatic construction of sophisticated behavioral models. This problem is a real challenge for large systems, and an unavoidable and delicate task that directly affects the performances of model-based monitoring tools. We address self-modeling issues in two different ways: by assembling generic model blocs, and by learning methods.

The first approach has been developed in previous RNRT projects, dedicated to alarm correlation techniques in telecommunication networks, and is now part of the team background. The principle is as follows: small generic network components are designed by hand, using information from failure propagation scenarios (expert knowledge) and information from technology standards. These network components include connectivity capabilities, as detailed in information model standards, and generic behaviors. As an interesting feature, standards define network components in a hierarchical manner, progressively refining their definition from general functions down to a specific technology and finally to equipment implementation. This hierarchy is reflected in our models, so only a limited part of component models has to be adjusted by hand. In a second time, the supervised network is scanned to discover which components are present, and how they are connected. This "network discovery" phase builds the model of the network by connecting the corresponding component models. It results in a possibly large model of the system that is then used as the basis for alarm correlation algorithms. In 2005-2006, this approach has been experimented in a real situation under a contract with the Alcatel Research and Innovation, in cooperation with the Optical Network business Division, see 7.4. Tools have been developed, both to easily model components in UML (with the above mentioned inheritage features), and to perform failure diagnosis in submarine line terminal equipment.

These last two years, effort has be shifted toward the second approach, by automatically inferring or refining (part of) the model with tests and learning algorithms.

Efficient learning algorithms [45], [71], [50] were known for a long time in the sequential case with centralized observations. Interestingly enough, learning a distributed system is not as easy as learning a sequential one, and in many cases, Vasilevskii-Chow's algorithm is not efficient anymore. In [19], we provide a new algorithm for learning a distributed system, together with the proof that it is optimal, and experimental results to compare it with generic algorithms.

Last year, we proposed an online compression algorithm for distributed executions [54] that could be used to infer a scenario model from observations of a distributed system. We have also investigated extensions of scenario models [56] to gain expressive power and allow users to model typical behaviors of distributed systems containing sliding-windows executions. In complement, we had studied how to obtain HMSCs by abstraction of communicating automata [49]. This year, we have investigated several approaches to provide users with languages and tools to build expressive models more easily.

The first work in this direction [27] consists in weaving behavioral aspects on scenario model. Aspects are a new approach to separate different concerns of software, and, more recently, models. We have proposed a weaving mechanism for Message Sequence Charts. In the MSC world, an aspect consists in a simple finite scenario called the *Pointcut*, and of another scenario called the *Advice*. The principle of scenario weaving is as follows: the pointcut scenario shall be replaced by the advice wherever it is detected in the partial orders generated by a HMSC. This scenario weaving can be interpreted as a rewriting technique for scenarios. However, the rewritten partial order family can not always be expressed by a HMSC. We have proposed an algorithm that perform this kind of rewriting when the result remains expressible as a HMSC.

Another research direction consists in helping a user collecting and gathering observations of a system to build a coherent model. In 2004, we have proposed an operator to compose redundant scenarios [59], called amalgamated sum. The main drawback of this sum is that the common parts of two behaviors must be given explicitely before composing two scenarios. This year, we have provided an algorithm to detect automatically redundancies in scenarios [25]. As the number of possible solutions that have to be studied is exponential in the size of the smallest scenario, we have implemented an heuristic method, and proved its convergence towards the best solution (i.e., the greatest common subsets in the two scenarios).

6.2. Fundamentals of modeling

Keywords: *category theory*, *concurrent systems*, *distributed algorithms*, *event structures*, *probabilistic event structures*, *unfoldings*.

Participants: Albert Benveniste, Claude Jard, Eric Fabre, Stefan Haar, Loïc Hélouët, Blaise Genest, Thomas Chatain, Agnes Madalinski.

In this section we collect our fundamental results regarding the models we use for distributed systems.

6.2.1. Event structures and unfoldings for distributed systems.

The monitoring algorithms developed in Distribcom heavily rely on an efficient representation of trajectory sets for large concurrent systems. Unfoldings, event structures, or the more recently proposed time-unfoldings are natural candidates for that. A key feature to derive efficient distributed algorithms is the factorization property of these objects: when a system can be expressed as a combination of components, its (time-)unfolding can be expressed as well as a combination of (time-)unfoldings of its components. The "combination" can be a standard product of components, or more interestingly can be a pullback. The latter expresses that components interact via an interface. We have shown that all pullbacks exist in the category of safe Petri nets [22], and so, by category theory arguments, these operations are preserved by all the "unfolding" operations mentioned above.

Once one knows that the unfolding of a compound system admits itself a factorized form, a problem consists in computing its minimal factors, *i.e.* the trajectories of each component that contribute to a least one trajectory of the global system. In other words, one would like to determine projections of the global unfolding on each component, without computing the (huge) global unfolding, of course. Several solutions have been proposed, that take the form of local computations combining products (or pullbacks) and projections. For example in [52] computations were based on event structures, and in [53] they were based on augmented branching processes. In [14], Paolo Baldan (Venice), Stefan Haar and B. Koenig (Duisburg) address the problem in yet another manner: they consider a restricted set of compositions by pullback, and propose to base computations on *interleaving Structures* or **ILS**. The advantage is that projections are (much) easier in the category of **ILS** ; in particular, [14] shows that **ILS** possess a useful (embedding,projection)-factorization property.

Distributed computations based on event structures, on branching processes (prefixes of unfoldings) or on trellis processes (prefixes of time-unfoldings) are quite technical. This is essentially due to the specific features of the true concurrency semantics. We have done the exercise of re-expressing the theory in a simpler framework, where runs are ordinary sequences of events rather than partial orders. All results can be rederived nicely: products and pullbacks exist, unfoldings and time-unfoldings can also be defined and enjoy the same factorization properties. And projections are much easier to define. Alltogether, one can design distributed monitoring algorithms for networks of automata in very much the same way as for combinations of Petri nets [21]. Several surprises came up however. In particular, trellis processes enjoy the necessary properties only if they are defined with respect to a local notion of time (time elapses differently in each component), instead of a global notion of time. This strongly suggests that one should avoid using a global clock if distributed computations are desired. And once local clocks are necessary, one is not far from true concurrency semantics [20], [32].

As a last topic related to distributed computations based on various representations of trajectory sets, we have explored the possibility of using symbolic unfoldings (see next section). The latter encode in an even more compact manner the unfolding of systems where transitions are “programs” changing the value of a limited set of variables. The idea is to avoid representing all possible changes on these variables (as in the classical notion of unfolding), but rather to simply indicate that these variables have been modified by the transition. We have proposed a category theory description of these objects, and proved that the symbolic unfolding functor preserves product. The last step to distributed computations will be the derivation of a projection operation. Paper in preparation.

6.2.2. Probabilistic event structures and Markov nets.

Our work on *true concurrency probabilistic models* is joint work with our former PhD student Samy Abbes, this year post-doc at LIAFA, Paris. The work of this year has consisted in finalizing papers. We review our progresses and refer the reader to the 2004 activity report for the motivations of this study.

In year 2000, we launched a research programme on probabilistic models of concurrent systems. This is different from stochastic Petri nets in all existing variants, since the latter are ultimately interpreted as Markov chains, a model in which both state and time are global. This is also different from probabilistic automata and process algebras, which are in fact tightly related to Markov Decision Processes. In our case, trajectories ω are partial orders of events with local states, and the Ω space consists of the set of maximal configurations of the unfolding of the considered net. This problem was also recently and independently considered by Hagen Völzer [72], Daniele Varacca and Glynn Winskel [70].

S. Abbes has constructed *probabilistic event structures* [1]. His work encompasses event structures with confusion (corresponding to Petri nets that may not be free choice). The models he has developed exhibit the nice property that concurrent processes are probabilistically independent, conditionally on their common past. In words, “concurrency matches probabilistic independence”. When specialized to event structures arising from safe Petri nets, this model specializes to that of *Markov nets*. Markov nets satisfy a strong Markov property and are such that concurrency matches probabilistic independence. Global and local renewal theory for Markov nets have been developed by S. Abbes [43]. This year has been devoted to the finalization of an important paper [10] on Markov nets and their law of large numbers.

6.2.3. Scenarios.

Research on scenarios has followed two main directions. The first direction is the extension of the expressivity of scenario languages. The second direction is the study of composition mechanisms for scenarios.

An important aspect of our work is to compare different modeling formalisms to know what can and cannot be described with a particular formalism. This year, we have obtained an important result regarding communicating systems, and their relationship with scenarios. We prove in [11] that the CHMSC scenario language, the communicating finite state machines, the regular languages of interleavings and the monadic second order logic on MSCs have the same expressivity under realistic assumptions, de facto extending Kleene, Buchi and Zielonka Theorems. The restriction we consider is existential boundednesses, meaning that there exists a channel bound such that every scenario can be executed using that bound, even if *some* interleaving of a scenario may overcome the bound. If a scenario is not existentially-bounded, then the system can make some choices leading to a deadlock, whatever scheduler and channel bounds are used. Not only did we prove that the expressive power is the same, but we gave a constructive algorithm to transform one formalism into another one, allowing to verify such systems. A perspective is to use such results in control, and more particularly in quasi static scheduling.

The research on scenarios has followed two other directions. The first direction is the extension of the expressivity of scenario languages. The second direction is the study of composition mechanisms for scenarios.

A major challenge with scenarios is to be able to model distributed sytems while preserving the decidability of some problems. In their simplest version, scenarios can be seen as automata labeled by partial orders. These order automata generate a family of partial orders, that aim at describing non-interleaved execution traces of distributed systems. Several problems such as vacuity of families intersection, equality, and so on are reputedly undecidable [62]. However, some interesting problems, such as diagnosis remain decidable [24].

Scenarios embed the expressive power of Mazurkiewicz traces (which results in most of undecidability results associated to scenarios), but are not able to model typical behaviors of so-called sliding windows. When a protocol implements a sliding window, distributed executions may have the shape of infinite braids. Partial order automata only allow for the description of recognizable MSC languages [61], i.e., families of partial orders that can be build by concatenation of a finite number orders from a finite order alphabet.

To extend the expressivity of order automata, several solutions have been proposed. E. Gunter proposed an extension called compositional MSC, that embeds into order automata the expressive power of communicating automata. Unsurprisingly, several decidable problems that were decidable for order automata (and among them diagnosis) become undecidable for compositional MSCs. This year, we have focused on an extension of order automata called causal MSCs that allows some commutation among composed orders. This extension seems to have nice properties : causal MSCs embed the expressive power of order automata, we can identify a regular subset of the language, and furthermore, diagnosis remains decidable on this model.

The second research direction concerning scenarios is composition. Previous work on composition [59] relied on a fibered product of order automata. The main disadvantage of this approach is that it forces an arbitrary synchronization among orders. This year, we have proposed a new approach to compose several scenarios, using a mixed product [31]. The mixed product allows an interleaving of two partial order automata. Some control on the interleaving is performed using control events that are necessarily located on a common process in the system. This composition mechanism opens new perspectives for a modular design of services specifications.

6.3. Algorithms: symbolic unfoldings and applications to diagnosis

Keywords: *High-level Petri net*, *Network of Timed Automata*, *Symbolic diagnosis*, *Symbolic unfolding*, *Time Petri net*.

Participants: Claude Jard, Thomas Chatain.

Since three years, we have developped in the context of Thomas Chatain's thesis, a supervision method based on the unfolding of high-level models of concurrency. By high-level, we mean models with variables, which demand a symbolic approach to build partial order trajectories from distributed observations.

In 2006, we focussed our research on time aspects. Time Petri nets have proved their interest in modeling real-time concurrent systems. Their usual semantics is defined in term of firing sequences, which can be coded in a (symbolic and global) state graph, computable from a bounded net. An alternative is to consider a partial order semantics given in term of processes, which keep explicit the notions of causality and concurrency without computing arbitrary interleavings. In ordinary place/transition bounded nets, it has been shown for many years that the whole set of processes can be finitely represented by a prefix of what is called the unfolding. We have defined [17], for the first time, such a prefix for safe time Petri nets. It is based on a symbolic unfolding of the net, using a notion of partial state.

It is also known that there are a lot of similarities between time Petri nets and networks of timed automata. Timed automata are also a well studied class of time models. Surprisingly, the concurrency aspects of this model are ignored, and the first reflex when dealing with networks od timed automata is to consider the computation of a single equivalent timed automaton (containing all the interleavings, but having destroyed the concurrency information). With the collaboration of F. Cassez from IRCCyN, Nantes, we have given in [16] a symbolic concurrent semantics for network of timed automata (NTA) in terms of extended symbolic nets. Symbolic nets are standard occurrence nets extended with read arcs and symbolic constraints on places and transitions. We prove that there is a complete finite prefix for any NTA that contains at least the information of the simulation graph of the NTA but keep explicit the notions of concurrency and causality of the network.

The last work considering time and concurrency was done in collaboration with the France Telecom research center in Lannion during the developpement of a new testing language called Late, in the context of the Emmanuel Donin's thesis. [18] presents a case study which is the test of a voice XML service. To develop this application, we proposed a new kind of non-deterministic testing. One testing scenario can describe several different executions and the interpreter tries to find the execution that well fit with the real behavior of the System Under Testing.

6.4. Algorithms: scenario based unfoldings

Keywords: *Mazurkiewics Traces*, *Scenario*, *Symbolic Unfoldings*, *diagnosis*.

Participants: Loïc Hélouët, Blaise Genest, Thomas Gazagnaire.

Using scenarios for diagnosis seems natural since an observation of a behavior can be described by means of a set of events and dependencies and independencies between these events. That is, an observation can be modeled easily as a scenario. Moreover, scenario languages enjoy a visual and appealing formalism. On the other hand, dealing with scenarios in algorithms is often non trivial [5]: since a scenario can be decomposed in several ways, finding a particular occurrence of a scenario in a set of behavior is not easy.

Our first result of the year in the topic is an algorithm for finding every behavior of an HMSC explaining a particular scenario observation scenario. The task is not easy since the behaviors can be arbitrarily long in case where there are many events that cannot be observed. In [24], we developed a symbolic algorithm to represent an infinite number of dependencies in a finite way. This gives us an algorithm that we proved to be optimal, without requiring any restriction. A prototype has been implemented to generate this set of explanations given an observation. We are currently researching a way to diagnose a scenario on-line and in a distributed way in order to get a monitoring tool.

In order to help distributing the diagnosis algorithm, we independently study the best way to pass information between peers to get a global knowledge of the behavior of the system. A fundamental procedure to share a regular and commutation closed knowledge in Mazurkiewicz traces is known as Zielonka's algorithm. However, the construction given in the original paper synthesizes exponential size information to pass with every communication, which is infeasible. In [23], we improve the fundamental Zielonka's algorithm with a symbolic data structure called tiles, which allow sharing only a polynomial size information. Furthermore, this information can be computed in quadratic time, while it was non tractable in the original construction. In order to achieve our goal for a monitoring tool, we need to deal with scenarios in Zielonka's algorithm instead of Mazurkiewicz traces, and to take care of non observable events.

Our last work in the topic of algorithms for Scenarios was to consider problem more complicated than diagnosis. Indeed, diagnosis tries to give an explanation to a single scenarios, whereas usually, we are interested in dealing directly with a set of scenarios. However, in general, problems on sets of scenarios like HMSCs are undecidable, due to the confusion between different decompositions of the same scenario. In [12], the intersection problem was considered for restriction of HMSCs: in case the HMSCs are local-choice, then the problem becomes tractable in quadratic time. However, the restriction is quite strict forbidding excessive parallelism, and many high level abstracted models may not fulfill it. Then the globally-cooperative restriction was defined, under which the intersection problem is decidable, although the complexity is not polynomial anymore. Interestingly enough, the main ingredient used in this construction is an encoding of scenarios into traces using atoms of MSCs. Bridging this gap between scenarios and traces should make the use of Zielonka's algorithm easier to obtain a monitoring tool.

6.5. Specific studies: cross-domain services in heterogeneous networks

Keywords: *heterogeneous networks*, *negotiation*, *telecommunications services*.

Participants: Stefan Haar, Eric Fabre, Hélia Pouyllau.

Consider the life-cycle of a cross-domain video-conference on demand. Suppose End-user A requests, with his host domain, a video-conference connection with user B, such that A and B do not have direct access to a common domain; the instances of the domains concerned (called service providers or SPs for short) thus have to set up a chain of inter-domain connections until the SP giving access to B is reached. Clearly, local domain services will not suffice for negotiating and managing this chain. With the Madynes team at LORIA and Alcatel, we are working on the algorithms for cross-domain QoS contract negotiation and monitoring; Hélia Pouyllau has implemented a prototype negotiation module using web services for the peer-to-peer negotiation of QoS budget along a fixed chain of service providers, by nested contractualization in which only neighboring SP's interact, and using dynamic programming (DP) techniques. The description of the algorithm is given in the publications [28], [13] and in SWAN deliverables. The module has been integrated into the management platform of the project SWAN. Current work focusses on extending the optimized negotiation for single requests into global multi-request optimal negotiation protocols, and the use of optimization principles other than DP.

6.6. Specific studies: Web services orchestrations

Keywords: *BPEL*, *Orchestrations*, *P2P Systems*, *QoS*, *Web Services*.

Participants: Stefan Haar, Albert Benveniste, Claude Jard, Sydney Rosario [intern ending Aug. 31, PhD starting Sept. 1st], Aakansha Chowdhery [intern], Rahul Jha [intern].

Regarding Web services *orchestrations* and *choreographies,* several standardization efforts are underway. The most mature effort is around Business Process Execution Languages (BPEL) [44]. WS-Choreography Definition Language (WS-CDL) [58] complements BPEL by paying attention to so-called choreographies, i.e., peers of interacting business processes. As these formalisms result from standardization discussions, they are quite complex, offer a number of detail features, and address technical difficulties such as the so-called problem of “correlations” with lengthly and informal explanations, which makes their modeling a cumbersome task — see, however, the work of [69] modeling of BPEL by means of Petri net systems of workflow type. This is why we decided to base this study on a simpler and much cleaner formalism for WS orchestrations, namely the ORC formalism proposed by Jayadev Misra and co-workers [60].

Most important is the study of Quality of Service (QoS) composition when composing Web services under orchestrations or choreographies. Here, the challenge is: 1/ To establish a relation between the QoS of queried Web services and that of the orchestration; 2/ To negotiate and tune the QoS parameters of the orchestration, in an efficient way; and 3/ To detect or predict the breaching of a QoS contract, leading to 4/ a reconfiguration of the orchestration.

All these tasks require having adequate models supporting QoS aspects. Regarding the functional aspect, Sydney Rosario had proposed last year a translation of ORC into the formalism of Petri net systems, *i.e.,* systems of equations involving Petri nets [51], see [29]. This behavioural model could then be enhanced with QoS parameters, but the resulting model was quite heavy .

This year, Claude Jard has developed a small tool written in Prolog to implement the sequential semantics of ORC as defined by the authors of the language. In addition, this tool can deliver a partial order form of the corresponding sequential executions, thus dramatically reducing the size of the executions for storing. Also, jointly with the inventors of ORC, namely William Cook and Jayadev Misra, from Austing University, we started developing direct translations of ORC into partial order models (more precisely, into so-called Asymmetric Event Structures). Based on preliminary versions of this translation, a tool chain has been developed by Sidney Rosario and our two indian interns, with the following features:

- A module performing the simulation of ORC programs in the form of partial orders;
- A module to generate samples of QoS parameters for the Web services called by the orchestration; these can be either given by measurements on actual Web services, or by Monte-Carlo generation from a Gamma or other distribution random simulator, or by bootstrapping a set of available measurements;

- A module to enhance the partial order executions with corresponding QoS parameters, based on MaxPlus and related algebras;
- A module with statistical tools to compute the composed QoS of the orchestration, from the QoS of the constituent Web services. This functionality allows to design and fine-tune global QoS contracts by selecting adequate quantiles over the sample data.

One objective of the tool is to demonstrate the possibility of performing overbooking, thus improving efficiency of the orchestration in terms of QoS.

6.7. Specific studies: Active XML documents

Keywords: *Active XML*, *P2P Systems*, *XML*.

Participants: Stefan Haar, Albert Benveniste, Eric Fabre, Loïc Hélouët, Blaise Genest, Debmalya Biswas [PhD], Il-Gon Kim [Post-Doc].

The language *Active XML* or *AXML* is an extension of XML which allows to enrich documents with *service calls* or sc's for short. These sc's point to web services that, when triggered, access other documents; this materialization of sc's produces in turn AXML code that is included in the calling document. One therefore speaks of dynamic or intentional documents; note in particular that materialization can be *total* (inserting data in XML format) or *partial* (inserting AXML code containing further sc's). AXML has been developed by the GEMO team at INRIA Futurs, headed by Serge Abiteboul; it allows to set up P2P systems around repositories of AXML documents (one repository per peer).

We are currently cooperating with the GEMO team (Serge Abiteboul) and the LIAFA laboratory in Paris (Anca Muscholl) to explore the behavioral semantics of AXML in the framework of the ASAX project, see 8.2 below. Our objective is to be able to ensure confluence despite distribution and asynchrony, even for documents not belonging to the so-called "positive" class [2], where confluence is ensured thanks to the absence of revision of facts.

Obviously, if one bounds the documents to some size, then the usual tools and techniques can be applied to analyze AXML documents. One of the work done this year was to show how to translate AXML documents into CSP under this restrictive hypothesis of boundedness [26].

One challenge is to model the dynamicity of the document that can grow arbitrarily large, and to define a way to unfold such structures. Another challenge is to define restrictions that are not too intrusive in order to model real systems, and not permissive enough to fall in the undecidability of every non trivial problem.

7. Contracts and Grants with Industry

7.1. RNRT project swan

Participants: Stefan Haar, Albert Benveniste, Eric Fabre, Claude Jard, Hélia Pouyllau, Sidney Rosario.

Contract INRIA *2 04 A 0082 MC 01 1 — December 2003/June 2006*

The project *"SWAN: Self-Aware Management"* is being funded by the French national network RNRT, Ministry of Research. I started in December 2003 and is scheduled to last 30 months. The DistribCom team cooperates in SWAN with

- the *MADYNES* team of INRIA Lorraine and Paris-Nord University (the latter replaced by *LABRI* Bordeaux in 2005),
- industrial partners Alcatel, France Telecom, and QoSmetrics.

SWAN aims at empowering local autonomous diagnosis and administration functions in networks and services. Compared to the preceding projects *MAGDA* and *MAGDA2*, where *asynchronicity* and *distribution* were already at the heart, the new additional challenge in SWAN is *dynamicity*, namely non-static topologies of interaction. Networks expand or shrink as peers and connections are added or withdrawn at runtime, with the necessary adaptations and negotiations managed locally in the domain directly concerned. Web Services show by nature this dynamical behavior. Both applications present thus a fundamental challenge to all model-based approaches to diagnosis and supervision more generally: find models that allow for self-modification - compare the discussion in the section on models of concurrency. DistribCom is leader of the SWAN project; the main scientific contributions are the formal investigation and simulations for Orc, described in 6.6, and the multi-domain QoS negotiation detailed in 6.5

7.2. RNRT project Persiform

Participants: Loïc Hélouët, Claude Jard.

RNRT November 2004 - November 2006

Very often, software and systems functionalities and performance models are developed by different kind of specialists. The goal of Persiform is to provide a complete methodology and toolbox to allow performance evaluation from functional models. A first prototype of a toolchain that translates functional models (namely sequence diagrams and activity diagrams) into a performance model(namely SES Workbench models). Functional languages are first translated into a common language, a variant of stochastic colored Petri nets. These nets are then transleted into a queueing network model. The partners for this project are: France Telecom R&D, Verimag, INRIA Rennes, INT, and a software company, Orpheus.

7.3. CO2 : Composition and coherence of scenarios

Participants: Loïc Hélouët, Blaise Genest.

External research project with France Telecom

Software development often starts with requirement capture, i.e. collecting a set of representative behaviors of a system. Scenarios collected can be considered as partial views of a system, but may however involve some incoherences. The objective of CO2 was to provide formal definitions of scenario compositions, define notions of coherence for a set of views defined as a collection of scenarios, and provide decision algorithms indicating whether there exists an implementation realizing them. CO2 ended in september. The outcome of the project is a prototype to compose scenarios, 8 deliverables, and a publication in a conference. Furthermore, the work accomplished during CO2 on the mixed product of scenarios opened new perspectives on modular composition of services.

7.4. Alcatel contract

Participants: Eric Fabre, Guy-Bertrand Kamga, Franck Wielgus.

December 2004 - May 2006

The general objective of this contract with Alcatel Research & Innovation is to perform exploratory developments in relation with two Alcatel business divisions, namely: Optical Networks Division and Mobile Radio Division.

In general, telecommunication systems are composed of many interconnected functions, softwares, protocol layers, etc. These elements are designed to monitor their internal state and their ability to fulfill the desired function. In case of abnormal behavior, malfunction or failure, they raise alarms that are collected by a supervisor. The interdependence of components, and the general failure propagation phenomenon, introduces a strong correlation between alarms. So one generally observes bursts of correlated information, that must be analyzed and interpreted to locate possible origin(s) of the failures (up to now, this work is performed by a human operator).

The objective of this contract is to develop diagnosis methods for such systems, made of many interconnected functions. We make use of models that capture the concurrency of behaviors, and describe runs of such systems by event structures. Two application domains have been identified : Submarine Line Terminal Equipment, for high rate optical intercontinental transmissions, and the radio access layer, for GSM/GPRS networks.

8. Other Grants and Activities

8.1. Politiques de sécurité: TEST et Analyse par le Test de systèmes en réseau ouvert

Participant: Loïc Hélouët.

ACI Sécurité — september 2004 - september 2007

The purpose of the Potestat ACI is to study security policies in networks, and to analyze the security of such networks with test techniques. The partners involved in this ACI are : LSR/IMAG - INPG (Vasco team), VERIMAG (DCS team), INRIA Rennes (Vertecs, Lande, and DistribCom teams)

8.2. Arc (Action de Recherche Coopérative) ASAX

Participants: Albert Benveniste, Eric Fabre, Stefan Haar, Loïc Hélouët, Blaise Genest, Hélia Pouyllau, Il-Gon Kim.

February 2005 - January 2007

ASAX (http://gemowiki.futurs.inria.fr/twiki/bin/view/Gemo/AsaxWeb) is a cooperative research action headed by DistribCom, in cooperation with INRIA's GEMO team, the LIAFA/Paris, and Tel-Aviv university. It started in January 2005 and is scheduled to end in December 2006. ASAX's purpose is the analysis of Active XML systems, see URL http://activexml.net on Active XML and Web services. Currently, only a fragment of AXML, called "positive AXML", such as systems having monotonic answers to queries, can be given a deterministic behavioral semantics [2]; the goal of ASAX is to break this limitation, and provide a formal semantics and analysis algorithms for AXML systems.

8.3. Control, Analysis and Synthesis of Distributed Systems

Participants: Loïc Hélouët, Blaise Genest, Thomas Gazagnaire, Shaofa Yang.

Associated Team INRIA-NUS — 2006

This associated team is a collaboration with the National University of Singapore. The main research theme is the control and diagnosis of distributed communicating systems. Two application areas are targeted: Real-time embedded systems and telecommunications systems and services. Although very different in nature, both areas make fundamental use of models of concurrency. Several types of formal models are considered: scenario languages, communicating automata and Petri-nets. More specifically, we work together on the following problems:

- An extension of scenario models for distributed systems diagnosis.
- Distributed control synthesis, with applications to the quasi-static scheduling problem.

As the cooperation just begun works are still ongoing. Thomas Gazagnaire spent three months in NUS between May and July, working on sliding scenarios. Blaise Genest spent two weeks in NUS in May working on quasi-static scheduling. Loïc Hélouët plan to spend two weeks in NUS in the end of 2006 to work on a new formalism of specification.

9. Dissemination

9.1. Scientific animation

A. Benveniste is associated editor at large (AEAL) for the journal *IEEE Trans. on Automatic Control* and member of the editorial board of the journal and «Proceedings of the IEEE». He has been in 2006 member of the Program Committee of the following conferences: WODES, EMSOFT. He has been plenary speaker at WODES'2006. He is member of the Strategic Advisory Council of the Institute for Systems Research, Univ. of Maryland, College Park, USA. He is in charge of managing the INRIA side of the Alcatel external Research Programme (ARP).

E. Fabre is co-organizing with Victor Khomenko (Newcastle) the UFO workshop, a satellite event of ATPN'07. He has been invited in the Program Committe of DX'07.

C. Jard has been in 2006 member of the Program Committee of the following conferences: FORTE, NOTERE, TESTCOM, MOVEP, AFADL, and has been invited for 2007 to the Program Committee of ICAPTN, UFO and NOTERE. He has served as an expert in several programs of the French ministry of research (particularly in the RNRT programme in telecommunications). He is also member of the editorial board of the *Annales des Télécommunications* and the steering committee of MSR series of conferences. C. Jard is a member of the administration council of the ENS Cachan. He has been president of the Atlanstic research program (at Nantes). He participated to the scientific evaluation comittee of french labs (IRCCyN, Nantes, as president and LIFC, Besançon) and of the research center in computer science in Montreal. In 2006, C. Jard was member of the PhD Committees of O. Constant (University of Pau) and L. Huo (University of MacGill, Montreal) as rapporteur, of T. Chatain and E. Donin at Rennes (as supervisor).He also participated to the Habilitation Committee of V. Rusu at Rennes (as president).

Stefan Haar is member of the working group for evaluation of international activities with the COST committee of INRIA; he also served on the IFSIC's "commission de spécialistes” section 27 until summer of 2006. He is an associate editor of *IEEE Transactions on Automatic Control.*

Loïc Hélouët was invited to become co-rapporteur at ITU for the question 17 on MSC language. This nomination should become official in december. He was also invited to participate in the program committee of SDL 2007.

9.2. Teaching

É. Fabre teaches information theory and communication theory at Ecole Normale Supérieure de Cachan, Ker Lann campus, in the computer science and telecommunications magistère program.

L. Hélouët teaches the UML notation to the mastere classes at ENST Bretagne. He also participates in module MAS (with C.Jard and S.Haar) of master M2RI, that is dedicated to models and algorithms for large systems supervision.

C. Jard is a full-time professor at the ENS Cachan and teaches mainly at the Master level, in Computer Science and Telecom, and in Maths. He manages the Info-Telecom track of the Master-Recherche-STS of the Rennes 1 university. It is to be noted that one course in this track is on the research subject of DistribCom. He is also in charge of the competitive examination for the entry of new students in computer science in the French ENS schools.

9.3. Participation in workshops, seminars, lectures, etc.

A. Benveniste gave one of the three plenary lectures at the conferences WODES'2006, July 2006, Ann Arbor. The talk was co-authored with Eric Fabre and title was: “Partial order techniques for distributed discrete event systems: why you can't avoid using them”.

S. Haar presented, on invitation by Reiko Heckel, a seminar on aspects of unfolding and diagnosis at the CS department of the university of Leicester/UK in June 2006.

L. Hélouët has been invited in march 2006 at the LABRI (Laboratoire Bordelais de Recherche en Informatique), to give a talk on applications of game theory to covert channels detection.

B. Genest gave a talk in March 2006 at the LIAFA, Paris 7 on learning algorithms for distributed systems. During his stay at NUS, Singapore, he presented ongoing work on distributed dynamic monitoring. He also presented the improvement of Zielonka's algorithm at LABRI in October 2006.

9.4. Visits and invitations

Guy-Vincent Jourdan is professor at the university of Ottawa/CDN. He has visited Distribcom during two months (may-june 2006) as invited professor at the ENS Cachan and INRIA. He worked with C. Jard and S. Haar on concurrent machine identification, using a new model of IO-partial order automaton. This was the starting point for a project of formal collaboration. S. Haar has visited U. of Ottawa for several months, and requested an INRIA sabbatical leave for 2007 to continue working in Ottawa.

Christoforos Hadjicostis visited Distribcom for two weeks in November 2006. This cooperation aims at building modular diagnosers for distributed systems.

10. Bibliography

Major publications by the team in recent years

[1] S. ABBES, A. BENVENISTE. *Probabilistic true-concurrency models: branching cells and distributed probabilities*, in "Information and Computation", february 2006, p. 231-274.

[2] S. ABITEBOUL, Z. ABRAMS, S. HAAR, T. MILO. *Diagnosis of asynchronous discrete event systems: datalog to the rescue!*, in "PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA", ACM Press, 2005, p. 358–367.

[3] A. BENVENISTE, E. FABRE, C. JARD, S. HAAR. *Diagnosis of asynchronous discrete event systems, a net unfolding approach*, in "IEEE Transactions on Automatic Control", vol. 48, n^o 5, May 2003, p. 714–727.

[4] E. FABRE, A. BENVENISTE, S. HAAR, C. JARD. *Distributed Monitoring of Concurrent and Asynchronous Systems*, in "Journal of Discrete Event Systems, special issue", 5 2005, p. 33–84.

[5] B. GENEST, A. MUSCHOLL, D. PELED. *Message Sequence Charts.*, in "Lectures on Concurrency and Petri Nets", vol. LNCS 3098, 2003, p. 537-558.

[6] L. HÉLOUËT, C. JARD, B. CAILLAUD. *An Event Structure Semantics for Message Sequence Charts*, in "Mathematical Structures in Computer Science (MSCS) journal", vol. 12, n^o 4, 2002, p. 377–403.

Year Publications

Books and Monographs

[7] C. JARD. *Méthodes formelles pour l'ingénierie des protocoles*, Vuibert, 2006.

Doctoral dissertations and Habilitation theses

[8] T. CHATAIN. *Symbolic Unfoldings of High-Level Petri Nets and Application to Supervision of Distributed Systems*, Ph. D. Thesis, Université de Rennes 1, 2006.

Articles in refereed journals and book chapters

[9] S. ABBES, A. BENVENISTE. *Probabilistic true-concurrency models: branching cells and distributed probabilities*, in "Information and Computation", february 2006, p. 231-274.

[10] S. ABBES, A. BENVENISTE. *Probabilistic true-concurrency models: Markov nets and a law of large numbers*, in "TCS special issue: FOSSACS 2005", to appear, 2006.

[11] B. GENEST, D. KUSKE, A. MUSCHOLL. *A Kleene Theorem and Model Checking for a Class of Communicating Automata.*, in "Information and Computation.", vol. 204, n^{o} 6, 2006, p. 920-956.

[12] B. GENEST, A. MUSCHOLL, H. SEIDL, M. ZEITOUN. *Infinite-State High-Level MSCs: Model-Checking and Realizability.*, in "J. Computer and System Science JCSS.", vol. 72, n^{o} 4, 2006, p. 617-647.

Publications in Conferences and Workshops

[13] A. AGHASARYAN, S. PIEKAREC, H. POUYLLAU, S. HAAR, E. FABRE, L. CIARLETTA, N. MBAREK, E. MOREAU. *Multi-Domain Self Aware Management: Negotiation and Monitoring*, in "13th Int. Conf. on Telecommunications, Funchal, Madeira", May 2006.

[14] P. BALDAN, S. HAAR, B. KOENIG. *Distributed Unfolding of Petri Nets*, in "FOSSACS 2006", LNCS, n^{o} 3921, Springer, 2006, p. 126-141.

[15] D. BISWAS, K. VIDYASANKAR. *Modeling Visibility in Hierarchical Systems.*, in "ER", vol. LNCS 4215, 2006, p. 155-167.

[16] F. CASSEZ, T. CHATAIN, C. JARD. *Symbolic unfoldings for networks of timed automata*, in "Proc. of ATVA'06", 2006.

[17] T. CHATAIN, C. JARD. *Complete finite prefixes of symbolic unfoldings of safe time Petri nets*, in "Proc. of ICATPN", LNCS, n^{o} 4024, June 2006, p. 125–145.

[18] E. DONIN DE ROSIÈRE, C. JARD, B. PARREAUX. *LaTe, A Non-fully Deterministic Testing Language*, in "18th IFIP International Conference on Testing Communicating Systems -TestCom'2006", LNCS, n^{o} 3964, May 2006.

[19] E. ELKIND, B. GENEST, D. PELED, H. QU. *Grey Box Checking.*, in "FORTE", vol. LNCS 4229, 2006, p. 420-435.

[20] E. FABRE, A. BENVENISTE. *Partial Order Techniques for Distributed Discrete Event Systems: why you can't avoid using them*, in "plenary address, 8th Int. Workshop on Discrete Events Systems, WODES, Ann Arbor", July 2006.

[21] E. FABRE. *A Trellis Notion for Distributed System Diagnosis with Sequential Semantics*, in "8th Int. Workshop on Discrete Events Systems, WODES, Ann Arbor", July 2006.

[22] E. FABRE. *On the Construction of Pullbacks for Safe Petri Nets*, in "Applications and Theory of Petri Nets and other Models of Concurrency, ATPN'06, Turku, Finland,", June 2006.

[23] B. GENEST, A. MUSCHOLL. *Constructing Exponential-size Deterministic Zielonka Automata.*, in "ICALP", vol. LNCS 4052, 2006, p. 565-576.

[24] L. HÉLOUËT, T. GAZAGNAIRE, B. GENEST. *Diagnosis from Scenarios*, in "proc. of WODES'06", 2006, p. 307-312.

[25] L. HÉLOUËT, T. HÉNIN, C. CHEVRIER. *Automating scenario merging*, in "Proc. of SAM' 2006", LNCS, Springer, 2006, ??.

[26] I.-G. KIM, D. BISWAS. *Application of Model Checking to AXML System's Security: A Case Study.*, in "WS-FM", vol. LNCS 4184, 2006, p. 242-256.

[27] J. KLEIN, L. HÉLOUËT, J. JÉZÉQUEL. *Semantic-based Weaving of Scenarios*, in "Proc. of the 5th International Conference on Aspect-Oriented Software Development (AOSD' 06), Bonn, Germany", March 2006.

[28] H. POUYLLAU, L. CIARLETTA, A. AGHASARYAN, S. HAAR. *X-domain QoS budget negotiation using Dynamic Programming*, in "IEEE Advanced International Conference on Telecommunications (AICT)", 2006.

[29] S. ROSARIO, A. BENVENISTE, S. HAAR, C. JARD. *Foundations for Web services Orchestrations: functional and QoS aspects*, in "ISOLA 2006, 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation", 2006.

Internal Reports

[30] M. BOZGA, S. GRAF, L. HÉLOUËT. *Modèle intermédiaire pour la transformation des formalismes utilisateurs vers les langages orientés performance (langage de la couche LU')*, Livrable 2.1, Technical report, Projet Persiform, 2006.

[31] P. DARONDEAU, B. GENEST, L. HÉLOUËT. *L2.2 R : description d'un modèle de tuiles, propriétés fondamentales*, Technical report, Projet CO2, 2006.

[32] E. FABRE, A. BENVENISTE. *Partial Order Techniques for Distributed Discrete Event Systems: why you can't avoid using them*, plenary address at WODES'2006, Ann Arbor, MI, USA, Technical report, n^{o} 1800, IRISA, may 2006.

[33] L. HÉLOUËT, O. CONSTANT, C. JARD. *Traduction des Diagrammes d' activités et des Message Sequence Charts vers le formalisme intermédiaire des Réseaux de Petri Colorés Stochastiques*, Livrable 3.1, Technical report, Projet Persiform, 2006.

[34] L. HÉLOUËT. *l1.2.2 R : Description de l'intégration d'un algorithme de calcul d'interface dans SOFAT*, Technical report, Projet CO2, 2006.

[35] L. HÉLOUËT. *L3.1 R: Critères de cohérence entre vues*, Technical report, Projet CO2, 2006.

[36] L. HÉLOUËT. *L3.1 R: Critères de réalisabilité d'un ensemble de vues*, Technical report, Projet CO2, 2006.

[37] L. Hélouët, T. Hénin, C. Chevrier. *Livrable L1.2.1 R: Rapport technique décrivant l'algorithme de calcul d'interface*, Technical report, Projet CO2, 2006.

[38] S. Rosario, A. Benveniste, S. Haar, C. Jard. *Net systems semantics of Web Services Orchestrations modeled in ORC.*, Technical report, n^{o} 1780, IRISA, jan. 2006.

References in notes

[39] *Business Process Execution Language for Web Services Version 1.1*, 2005, http://www.ibm.com/developerworks/library/ws-bpel/.

[40] H. Ehrig, H.-J. Kreowski, U. Monanari, G. Rozenberg (editors). *Handbook of graph grammars and computing by graph transformations: vol. 3, concurrency, parallelism, and distribution*, World Scientific, 1999.

[41] *Web Services Flow Language (WSFL 1.0)*, 2001, http://www.ibm.com/.

[42] *XLANG, Web Services for Business Process Design*, 2001, http://www.gotdotnet.com/team/xml_wsspecs/xlang-c.

[43] S. Abbes. *The (True) Concurrent Markov Property and Some Applications to Markov Nets.*, in "ICATPN", G. Ciardo, P. Darondeau (editors). , LNCS, vol. 3536, 2005, p. 70-89.

[44] T. Andrew, F. Curbera, H. Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, S. Weerawarana. *Business Process Execution Language for Web Services. [BPEL4WS.], version 1.1.*, May 2003, http://xml.coverpages.org/BPELv11-May052003Final.pdf.

[45] D. Angluin. *Learning Regular Sets From Queries and Counterexamples*, in "Information and Computation", 1987, p. 87-106.

[46] P. Baldan, A. Corradini, U. Montanari. *Unfolding and event structure semantics for graph grammars*, in "Proc. of FOSSACS 1999", LNCS, vol. 1578, Springer Verlag, 1999, p. 73–89.

[47] M. Buscemi, V. Sassone. *High-Level Petri Nets as Type Theories in the Join Calculus*, in "FOSSACS'2001", 2001.

[48] C. Cassandras, S. Lafortune. *Introduction to discrete event systems*, Kluwer Academic Publishers, 1999.

[49] T. Chatain, L. Hélouët, C. Jard. *From Automata Networks to HMSCs: a Reverse Model Engineering Perspective*, in "25th IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2005)", LNCS, n^{o} 3731, Springer Verlag, October 2005, p. 489–502.

[50] T. Chow. *Testing software design modeled by finite-states machines.*, in "IEEE transactions on software engineering", vol. SE-4, 1978, p. 178-187.

[51] R. R. Devillers, H. Klaudel, M. Koutny. *Petri Net Semantics of the Finite pi-Calculus.*, in "FORTE", 2004, p. 309-325.

[52] E. FABRE, A. BENVENISTE, S. HAAR, C. JARD. *Distributed Monitoring of Concurrent and Asynchronous Systems*, in "Journal of Discrete Event Systems, special issue", 2004.

[53] E. FABRE. *Factorization of Unfoldings for Distributed Tile Systems, Part 2 : General Case*, Publication Interne, n^{o} 1606, IRISA, may 2004, ftp://ftp.irisa.fr/techreports/2004/PI-1606.ps.gz.

[54] T. GAZAGNAIRE, L. HÉLOUËT, C. JARD. *Online abstraction of distributed executions*, to appear, Technical report, INRIA, 2005.

[55] A. GUPTA. *Integration of Information Systems: Bridging Heterogeneous Databases*, IEEE Press, 1989.

[56] L. HÉLOUËT. *Weakening Weak Sequential Composition in Scenarios*, Research report, n^{o} 5512, INRIA, March 2005.

[57] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*, ITU-TS, Geneva, September 1999.

[58] N. KAVANTZAS, D. BURDETT, G. RITZINGER, T. FLETCHER, Y. LAFON. *Web Services Choreography Description Language – WS-CDL, version 1.0.*, 2004, http://www.w3.org/2002/ws/chor/edcopies/cdl/cdl.html.

[59] J. KLEIN, B. CAILLAUD, L. HÉLOUËT. *Merging Scenarios*, in "Proc. of FMICS'04 (Formal Methods in Industrial and Critical Systems", Sep. 2004.

[60] J. MISRA. *A Programming Model for the Orchestration of Web Services.*, in "SEFM", 2004, p. 2-11.

[61] R. MORIN. *Recognizable sets of Message Sequence Charts*, in "Proc. of STACS 2002, 19th Annual Symposium on Theoretical Aspects of Computer Science", H. ALT, A. FERREIRA (editors). , LNCS, Springer Verlag, march 2002, p. 523-534.

[62] A. MUSCHOLL, D. PELED, Z. SU. *Deciding Properties for Message Sequence Charts*, in "Lecture Notes in Computer Science", vol. 1378, 1998, p. 226-242, http://citeseer.ist.psu.edu/muscholl98deciding.html.

[63] M. NIELSEN, G. PLOTKIN, G. WINSKEL. *Petri nets, event structures and domains, part 1*, in "T.C.S.", 1981.

[64] OBJECT MANAGEMENT GROUP. *Unified Modeling Language Specification version 2.0: Superstructure*, Technical report, n^{o} pct/03-08-02, OMG, 2003.

[65] W. REISIG. *Petri nets*, Springer Verlag, 1985.

[66] M. RENIERS, S. MAUW. *High-level Message Sequence Charts*, in "SDL97: Time for Testing - SDL, MSC and Trends, Evry, France", A. CAVALLI, A. SARMA (editors). , Proceedings of the Eighth SDL Forum, September 1997, p. 291-306.

[67] M. RENIERS. *Message Sequence Charts: Syntax and Semantics*, Ph. D. Thesis, Eindhoven University of Technology, 1998.

[68] E. Rudolph, P. Graubman, J. Grabowski. *Tutorial On Message Sequence Charts*, in "Computer Networks and ISDN Systems", vol. 28, 1996, p. 1629-1641.

[69] B.-H. Schlingloff, A. Martens, K. Schmidt. *Modeling and Model Checking Web Services*, in "Electronic Notes in Theoretical Computer Science: Issue on Logic and Communication in Multi-Agent Systems", vol. 126, mar 2005, p. 3-26, http://www.sciencedirect.com/science?_ob=MImg&_imagekey=B75H1-4FKXPY9-F-1&_cdi=13109&_user=10&_orig=browse&_coverDate=03/08/2005&_sk=998739999&view=c&wchp=dGLzVzz-zSkWW&md5=da83be1d134c9d04ded5ee2029ac50bf&ie=/sdarticle.pdf.

[70] D. Varacca, H. Völzer, G. Winskel. *Probabilistic Event Structures and Domains.*, in "Proc. of CONCUR'04", LNCS, vol. 3170, 2004, p. 481-496.

[71] M. Vasilevskii.. *Failure diagnosis of automata.*, in "Kibertetika.", vol. 4, 1973, p. 98-108.

[72] H. Völzer. *Randomized non-sequential processes*, in "CONCUR", K. G. Larsen, M. Nielsen (editors). , LNCS, vol. 2154, 2001, p. 184–201.

[73] G. Winskel. *Event structures semantics in CCS and related languages*, in "LNCS", vol. 140, Springer Verlag, 1982.

[74] G. Winskel. *Categories of models for concurrency*, in "Seminar on Concurrency, Carnegie-Mellon Univ. (July 1984), LNCS 197", 1985, p. 246-267.

[75] R. D. van der Mei, H. B. Meeuwissen. *Modelling end-to-end Quality-of-Service for Transaction-Based Services in Multi-Domain Environments*, in "IEEE International Conference on Web Services, ICWS", september 2006, p. 453-462.

[76] T. Özsu, P. Valduriez. *Principles of Distributed Database Systems, 2nd Edition*, Prentice-Hall, 1999.