Rewriting Logic Revisited

José Meseguer

Computer Science Department
University of Illinois at Urbana-Champaign

Introduction

Rewriting logic is almost a teenager! Thanks to the efforts of many researchers, a lot has happened since the first papers were written in 1990.

This talk will give a personal view of some of the recent developments in theory, applications, and tools.

So much has happened that it is impossible to do justice to all the work that thas been done (over 350 papers). The intent will instead be to give an impressionistic picture of some conceptual developments and some promising research directions, among others.

(Planned) Contents of the Talk

- I. Theoretical Developments
- II. Rewriting Logic and Formal Methods
- III. Semantic Framework Applications
- IV. Logical Framework Applications (no time)
- V. Rewriting Logic Languages (no time)
- VI. Conclusions (no time)

I. Theoretical Developments

- Rewrite Theories in General
- Rewriting Logic in General
- Extensions of RWL and Related Formalisms
- Reflection
- Coherence

Rewrite Theories in General

From the beginning, it was clear that rewriting logic has the underlying equational logic as a parameter. More general equational logics give rise to correspondingly more general notions of (conditional) rewrite rule. For example, for membership equational logic, rules have the general form,

$$l : t \longrightarrow t' \;\Leftarrow\; (\bigwedge_i u_i = u'_i) \wedge (\bigwedge_j v_j : s_j) \wedge (\bigwedge_k w_k \longrightarrow w'_k).$$

It has also become increasingly clear that frozen operators, that restrict the rewrites allowed below them, are also very useful in practice.

Rewrite Theories in General (II)

We can illustrate frozen operators with the following nondeterministic choice example (in Maude syntax):

```
mod CHOICE is
  protecting INT .
  sorts Elt MSet .
  subsorts Elt < MSet .
  ops a b c d e f g : -> Elt .
  op __ : MSet MSet -> MSet [assoc comm] .
  op card : MSet -> Int [frozen] .
  eq card(X:Elt) = 1 .
  eq card(X:Elt M:MSet) = 1 + card(M:MSet) .
  rl [choice] : X:MSet Y:MSet => Y:MSet .
endm
```

Rewrite Theories in General (III)

It does not make much sense to rewrite below the cardinality function `card`, because then the multiset whose cadinality we wish to determine becomes a moving target.

If `card` had not been declared `frozen`, then the rewrites, `a b c` $\longrightarrow$ `b c` $\longrightarrow$ `c` would induce rewrites, $3 \longrightarrow 2 \longrightarrow 1$, which seems bizarre.

The point is that we think of the kind `[MSet]` as the state kind in this example, whereas `[Int]` is the data kind. By declaring `card` frozen, we restrict rewrites to the state kind, where they belong.

Rewrite Theories in General (IV)

This leads to the following general definition of a rewrite theory on membership equational logic:

A rewrite theory is a 4-tuple, $\mathcal{R} = (\Sigma, E, \phi, R)$, where:

- (Σ, E) is a membership equational theory, with, say, kinds K, sorts S, and operations Σ
- $\phi : \Sigma \longrightarrow \mathcal{P}_{fin}(\mathbb{N})$ is a $K^* \times K$-indexed family of functions assigning to each $f : k_1 \ldots k_n \longrightarrow k$ in Σ the finite set $\phi(f) \subseteq \{1, \ldots, n\}$ of its frozen argument positions
- R is a set of (universally quantified) labeled conditional rewrite rules of the form (with t, t' and the w_k, w'_k pairs of terms of same kind)

$$l : t \longrightarrow t' \;\Leftarrow\; (\bigwedge_i u_i = u'_i) \wedge (\bigwedge_j v_j : s_j) \wedge (\bigwedge_k w_k \longrightarrow w'_k).$$

Rewrite Theories in General (V)

Given a rewrite theory $\mathcal{R} = (\Sigma, E, \phi, R)$, and given a Σ-term $t \in T_\Sigma(X)$, we call a variable $x \in vars(t)$ frozen in t iff there is a nonvariable position $\alpha \in \mathbb{N}^*$ such that $t/\alpha = f(u_1, \ldots, u_i, \ldots, u_n)$, with $i \in \phi(f)$, and $x \in vars(u_i)$. Otherwise, we call $x \in X$ unfrozen.

Similarly, given Σ-terms $t, t' \in T_\Sigma(X)$, we call a variable $x \in X$ unfrozen in t and t' iff it is unfrozen in both t and t'.

Rewriting Logic in General

Given a rewrite theory $\mathcal{R} = (\Sigma, E, \phi, R)$, the sentences that it proves are universally quantified rewrites of the form, $(\forall X)\ t \longrightarrow t'$, with $t, t' \in T_{\Sigma,E}(X)_k$, for some kind k, which are obtained by finite application of the following rules of deduction:

- **Reflexivity**. For each $t \in T_{\Sigma}(X)$, $\dfrac{}{(\forall X)\ t \longrightarrow t}$

- **Equality**.
$$\frac{(\forall X)\ u \longrightarrow v \quad E \vdash (\forall X) u = u' \quad E \vdash (\forall X) v = v'}{(\forall X)\ u' \longrightarrow v'}$$

- **Congruence**. For each $f : k_1 \ldots k_n \longrightarrow k$ in Σ, with $\{1, \ldots, n\} - \phi(f) = \{j_1, \ldots, j_m\}$, with $t_i, t'_i \in T_\Sigma(X)_{k_i}$, $1 \leq i \leq n$, and with $t'_{j_l} \in T_\Sigma(X)_{k_{j_l}}$, $1 \leq l \leq m$,
$$\frac{(\forall X)\ t_{j_1} \longrightarrow t'_{j_1} \quad \ldots \quad (\forall X)\ t_{j_m} \longrightarrow t'_{j_m}}{(\forall X)\ f(t_1, \ldots, t_{j_1}, \ldots, t_{j_m}, \ldots, t_n) \longrightarrow f(t_1, \ldots, t'_{j_1}, \ldots, t'_{j_m}, \ldots, t_n)}$$

- **Replacement**. For each finite substitution $\theta : X \longrightarrow T_{\Sigma}(Y)$, with, say, $X = \{x_1, \ldots, x_n\}$, and $\theta(x_l) = p_l$, $1 \leq l \leq n$, and for each rule in R of the form,

$$l : (\forall X)\; t \longrightarrow t' \;\Leftarrow\; (\bigwedge_i u_i = u'_i) \wedge (\bigwedge_j v_j : s_j) \wedge (\bigwedge_k w_k \longrightarrow w'_k)$$

with $Z = \{x_{j_1}, \ldots, x_{j_m}\}$, the set of unfrozen variables in t and t', then,

$$\frac{\begin{array}{c}(\bigwedge_r (\forall Y)\; p_{j_r} \longrightarrow p'_{j_r}) \\ (\bigwedge_i (\forall Y)\; \theta(u_i) = \theta(u'_i)) \wedge (\bigwedge_j (\forall Y)\; \theta(v_j) : s_j) \wedge (\bigwedge_k (\forall Y)\; \theta(w_k) \longrightarrow \theta(w'_k))\end{array}}{(\forall Y)\; \theta(t) \longrightarrow \theta'(t')}$$

where for $x \in X - Z$, $\theta'(x) = \theta(x)$, and for $x_{j_r} \in Z$, $\theta'(x_{j_r}) = p'_{j_r}$, $1 \leq r \leq m$.

- **Transitivity**

$$\frac{(\forall X)\ t_1 \longrightarrow t_2 \qquad (\forall X)\ t_2 \longrightarrow t_3}{(\forall X)\ t_1 \longrightarrow t_3}$$

Rewriting Logic in General (II)

Note that the above notion of rewrite theory and the proof rules exactly generalize the simpler unsorted and unfrozen case, since this is a special case of the general notion of theory and of the general proof rules.

Of course, the categorical model theory must also be generalized. This is current joint work with Roberto Bruni.

One advantage over the 1992 TCS paper is that we now have membership equational logic (MEL) available (including its partial version); therefore the model constructions can be carried out in the MEL framework.

Extensions of RWL and Related Formalisms

To deal with other aspects besides concurrency, rewriting logic has been extended in several directions:

- Real-Time rewrite theories [ÖM01]
- Probabilistic rewrite theories (ongoing work by Kumar, Shen, Agha, and Meseguer at UIUC)
- Behavioral Extensions by Diaconescu [Dia96] and the recent work on behavioral MEL and BMaude by Meseguer and Roşu [MR02]

Extensions of RWL and Related Formalisms (II)

Furthermore, two related rewriting formalisms have been proposed, to deal, respectively, with synchronization, and with a λ-calculus-like universal rewriting model:

- Tile Logic by Gaducci, Montanari, and Bruni [GM00, Bru99]
- The ρ-calculus by Cirstea and Kirchner [CK99]

Reflection

Important progress has also happened in the area of reflection, showing that there are universal theories simulating all other (finitary) theories for:

- unsorted unconditional rewriting logic [Cla00]
- many-sorted conditional rewriting logic [CM01]
- membership conditional rewriting logic [CMP02]
- many-sorted equational and Horn logic with equality, and membership equational logic [CMP02]

Reflection (II)

Yet another area in which reflection has been substantially advanced is in the work by Basin Clavel and Meseguer on reflecting induction principles [BCM00].

This work makes possible reflective inductive proofs happening not in a sigle theory, but in a parameterized family of theories, and has many meta-reasoning applications.

Coherence

Thanks to the work of Viry, the important topic of coherence —a confluence-like property between rules and equations crucial for practical executability of rewrite theories— has been further advanced, particularly in the latest paper [Vir01] where,

- coherence modulo different combinations of A, C, and AC is treated, including completion algorithms

- the case of conditional rules is treated, and

- several important applications are discussed.

II. Rewriting Logic and Formal Methods

- A Wide Spectrum of Formal Methods
- System Specification vs. Property Specification
- Modal and Temporal Logics for Rewriting Logic
- Formal Tools

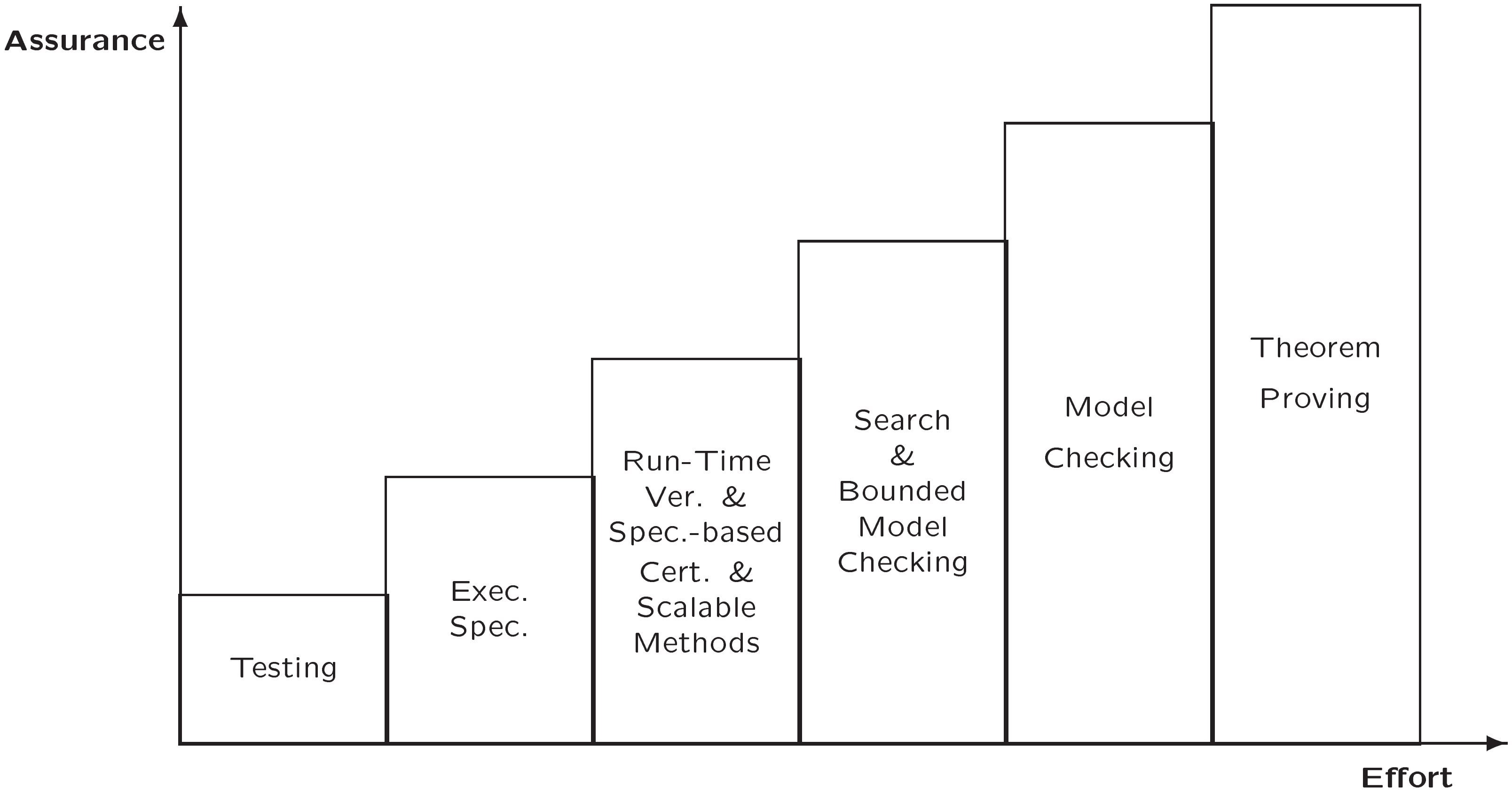
A Wide Spectrum of Formal Methods

Rewriting logic and its formal tools support a wide spectrum of formal methods. This allows great flexibilty in negotiating cost vs. assurance tradeoffs, which vary from application to application.

From less costly methods such as executable specification, through scalable methods such as runtime verification and specification-based certification, to increasingly stronger methods such as model checking and theorem proving, all methods can be supported.

All this can be done for rewriting logic specifications and programs, and for conventional code to which a rewriting logic semantics has been given.

Formal Methods: their Cost and Assurance



System Specification vs. Property Specification

Rewriting logic has good properties as a logical and semantic framework to specify systems. They may be formal systems, such as a logic or a theorem prover's inference system, or computational systems such as a programming language, a middleware system or a ceryptographic protocol: both can be specified very naturaly.

To ensure existence of initial models and executability, both rewriting logic and its underlying equational logic are lean logics, with no extra fat. While they have high expressive power to specify systems, they have more limited expressiveness to specify system properties.

System Specification vs. Property Specification (II)

The obvious idea is that we should use the most suitable tool for each purpose, rather than trying to use the same tool for everything. The system vs. property specification distinction helps us do that.

For property specification we do not need to be executable in any reasonably efficient sense (although the logic can be mechanizable), nor do we need theories with initial models. We can therefore choose more expressive logic extensions, such as full fist-order logic extensions of equational and rewriting logic. Or we can use semantically related logics, using appropriate mappings.

System Specification vs. Property Specification (III)

Some very natural choices, which have already been used as property specification logics for rewriting logic include:

- Full First-Order Logic, enriched with induction principles for the equational and rewriting specifications
- Modal and Temporal Logics
- Higher-Order Logics (e.g., OCC)

I discuss modal and temporal logics in more detail.

Modal and Temporal Logics for Rewriting Logic

The following modal and temporal logics have been proposed to reason about rewriting logic specifications:

- different modal logics by Fiadeiro et al. [FMMO^{+}00], and by Pattinson [Pat00]
- an object-oriented modal μ-calculus proposed by Lechner [Lec97]
- Denker's object-oriented distributed temporal logic [Den98]
- Ölveczky's timed linear time temporal logic used in Real-Time Maude [Ölv00]
- the logic $LTL(\mathcal{R})$ used in Maude's Model Checker [EMS02]

Formal Tools

The first reasoning tools to be mentioned are the CafeOBJ, ELAN, and Maude rewriting logic languages (to be further discussed in Part V). Indeed, besides supporting executable specification, these tools can be directly used to perform quite sophisticated forms of formal reasoning, including:

- behavioral rewriting and OTS verification (CafeOBJ)
- search-based reasoning (ELAN and Maude)
- efficient LTL model checking (Maude)

Formal Tools (II)

Next come tools belonging to formal environments supporting different kinds of formal reasoning for rewriting logic languages. Formal tools in this category include:

- CafeOBJ's Pig Nose resolution prover
- Maude's ITP [Cla00, CDEM00], and Church-Rosser Checker, KB completion, and Termination tools [CDEM00, Dur00]
- Real-Time Maude [Ölv00]
- the Open Calculus of Constructions (OCC) [Ste02]

Formal Tools (III)

A third category of tools comprises rewriting-logic-based formal tools used to support inference, reason about, or interoperate, different formal and computational sytems.

Both ELAN and Maude have been used to develop a wide variety of formal tools and automated deduction algorithms in this third category.

Some of them will be discussed as part of RWL's semantic framework (Part III) and logical framework (Part IV) applications.

III. Semantic Framework Applications

- Models of Computation
- Programming Languages and Hardware
- Software Architectures, Components, and Middleware
- Communication Protocols and Networks
- Bioinformatics

Models of Computation

The following models of computation have been naturally expressed in rewriting logic:

- equational programming, lambda calculi, and combinatory reduction systems [Mes92, LM92, LM96, SM99, Ste00]
- labeled transition systems, grammars and string-rewriting systems [Mes92]
- Petri nets, including place/transition nets, contextual nets, algebraic nets, colored nets, and timed Petri nets [Mes92, Mes96b, Ste01b, SMÖ01, ÖM01, Ste01a]

- Gamma and the Chemical Abstract Machine [Mes92]
- CCS, LOTOS, and E-LOTOS [MFW92, MOM93, Vir94, CDG98, DGP00, VMO00b, VMO, LN00]
- the π calculus [Vir96, Ste00]
- concurrent objects and actors [Mes92, Mes93, Tal96, Tal97b, Tal01]
- the UNITY language [Mes92]
- concurrent graph rewriting [Mes96b]
- dataflow [Mes96b]
- neural networks [Mes96b]

- real-time systems, including timed automata, timed transition systems, hybrid automata, and timed Petri nets [ÖM01, Ölv00]; and

- the tile logic [GM96, GM00, FM00] model of synchronized concurrent computation [MM98, BMM98, Bru99, GM01].

Operational vs. Mathematical Semantics

Since the above specifications of models of computation as rewrite theories are typically executable, this suggests that rewriting logic is a very flexible operational semantic framework to specify the semantics of such models.

What is not immediately apparent from the above list is that rewriting logic is also a flexible mathematical semantic framework for concurrency models.

Quite well-known mathematical models of concurrency happen to be isomorphic to the initial model $\mathcal{T}_{\mathcal{R}}$ of the rewrite theory $\mathcal{R}$ axiomatizing the particular models:

- Laneve and Montanari [LM96] show that for rewrite theories of the form $\mathcal{R} = (\Sigma, \emptyset, L, R)$, with the rules R left-linear, $\mathcal{T}_{\mathcal{R}}$ is isomorphic to a model based on residuals and permutation equivalence proposed by Boudol

- in the same paper they also show that for $\mathcal{R} = (\Sigma, E, L, R)$ a rewrite theory axiomatizing an orthogonal combinatory reduction system, including the λ-calculus, a quotient of $\mathcal{T}_{\mathcal{R}}$ by a few additional equations is isomorphic to a well-known model of parallel reductions based on residuals and permutation equivalence

- Stehr, Meseguer, and Ölveczky [SMÖ01] show in detail that for $\mathcal{R} = (\Sigma, E, L, R)$ a rewrite theory axiomatizing a place/transition net, $\mathcal{T}_{\mathcal{R}}$ is naturally isomorphic to the Best-Devillers net process model and then generalize this natural isomorphism to one between $\mathcal{T}_{\mathcal{R}}$ and a Best-Devillers-like model for $\mathcal{R}$ the axiomatization of an algebraic net

- work by Carabetta, Degano, Gadducci, and Priami [CDG98, DGP00] shows that for $\mathcal{R} = (\Sigma, E, L, R)$ a rewrite theory axiomatizing CCS, a truly concurrent semantics causal model based on proved transition systems is isomorphic to a quotient of $\mathcal{T}_{\mathcal{R}}$ by a few additional axioms

- Meseguer and Talcott [MT99] show that for $\mathcal{R} = (\Sigma, E, L, R)$ a rewrite theory axiomatizing a concurrent object-oriented system satisfying reasonable requirements, a subcategory of $\mathcal{T}_{\mathcal{R}}$ is isomorphic to a partial order of events model which, for asynchronous object systems corresponding to actors, coincides with the finitary part of the Hewitt-Baker partial order of events model.

Programming Languages and Hardware

Rewriting logic is a promising semantic framework for formally specifying programming languages as rewrite theories. Since those specifications usually can be executed in a rewriting logic language, they in fact become interpreters for the languages in question.

In addition, such language specifications can be subjected to a variety of formal analyses. These analyses can be carried out either through features and tools of rewriting logic languages, or through rewriting-logic-based program analysis tools.

Rewritig Logic vs. SOS

In particular, there is a close connection between SOS language specifications and their rewriting logic specification counterparts. One can:

- view SOS definitions directly as conditional rewrite theories [MOM93], or
- define a mapping $MSOS \longrightarrow RWL$ from Mosses' modular SOS [Mos98, Mos99], and prove its correctness, as done by Braga [Bra01], or
- use tile logic rewrite rules as an alternative formalism [GM96, GM00, GM01], which can then be mapped into rewriting logic for execution purposes [MM98, BMM98, Bru99].

Language and Hardware Specification Case Studies

A number of encouraging case studies, giving rewriting logic definitions of programming languages and harware, have been carried out by different authors:

- rewriting logic definitions of the lambda calculus and (mini-) ML, CCS, the π-calculus, and sketches of UNITY and Gamma (references given earlier)

- the usefulness of rewriting strategies to specify program evaluations has been clearly demonstrated in ELAN specifications of Prolog and of Babel [Vit94], and also in Maude specifications of CCS by Bruni and Clavel [Cla98, Bru99], and by Verdejo and Martí-Oliet [VMO00b, VMO00a]

- the fact that rewriting logic naturally supports concurrent objects has proved very useful in formally specifying a number of novel concurrent and mobile languages, including:
  - Ishikawa et al.'s formal specification of the GAEA language [IMW+97, IFW]
  - Mason and Talcott's rewriting logic semantic definitions of actor languages [MT97]
  - the use of Maude by Van Baalen, Caldwell, and Mishra, to give a formal semantics to the DaAgent mobile agent language [BCM]

- ◦ Stehr and Talcott's formal executable specification in Maude of UPenn's PLAN active network programming language [**?**].
- ◦ the semantic definition in Maude of its Mobile Maude extension [DELM00]

- finally, Harman has used Maude to specify and verify microprocessors [Har00a, Har00b].

RWL-Based Program Specification & Analysis Tools

The following tools use Maude for program specification and analysis:

- Braga et al.'s Maude Action Tool (for Action Semantics) [BHMM00]

- Braga's tool for MSOS specification and execution [Bra01]

- Fischer and Roșu's abstract interpretation specification, execution, and verification tool [FR01]

- Havelund and Roșu's LTL runtime verification module of Java Path Explorer [HR01]

- Roșu et al.'s domain-specific program certifiers for: (1) units of measurment, and (2) state estimation.

- Verdejo and Martí-Oliet's CCS specification and verification tool [VMO00b]

Software Architectures, Components, and Middleware

It is very important to detect errors and inconsistencies as early as possible in the software design cycle. Formal approaches can increase the analytic power of architectural notations such as architectural description languages (ADLs) and object-oriented design notations like UML.

A related concern is the formal specification and analysis of distributed component architectures, including middleware systems.

Rewriting logic applications in these areas include:

- Nodelman and Talcott's work representing the Wright ADL and its CSP semantics in Maude
- Durán, Meseguer, and Talcott's work on semantic

interoperation of heterogeneous software architectures [MT, CDE+99]

- Wirsing and Knapp's work on transforming UML diagrams into Maude specifications [WK96, Kna00, Kna01, WK01];
- Fernández-Alemán and Toval's formalization in Maude of the UML metamodel and its evolution, for formal analysis and prototyping [TF00, FT01, FT00, TF01]
- Nakajima and Futatsugi's work on the transformation of GILO-2 O-O design diagrams for execution and formal analysis [NF97];
- Talcott's work on actor-based distributed components [Tal96, Tal97a, Tal97b, Tal00, Tal01]

- Denker, Meseguer, and Talcott's work on a composable distributed communication services [DMT00b];
- Najm and Stefani's work on the Reference Model for Open Distributed Processing (RM-ODP) [NS94, NS95, NS97] (see also [DV01])
- work by Nakajima on the architecture of WEB-NMS, a Java/ORB implementation of a network management system [Nak]
- work by Albarrán, Durán, and Vallecillo on interoperating Maude executable specifications with distributed component platforms such as CORBA and SOAP [ADV01a, ADV01b, ADV01c], and
- Meseguer and Talcott's work on semantic models of reflective distributed systems [MT02].

Communication Protocols and Networks

Because of its flexibility to model distributed objects with different modes of communication and interaction, rewriting logic is very well suited to specify and analyze communication protocols, including cryptographic protocols, and, more generally, network software such as active network programming languages, active network algorithms, and network management systems.

Applications of this kind include:

- work by researchers at Stanford, SRI, and at the Computer Communications Research Group at University of California Santa Cruz using Maude to analyze the early design of a new reliable broadcast protocol for active networks [DGLAM+99a, DGLAM+99b]

- work of Denker, Meseguer, and Talcott on the specification and analysis of cryptographic protocols using Maude [DMT98, DMT00a]

- work by Rodriguez on analysis of security prptocols [Rod00]

- work of Basin and Denker on an experimental comparison of the advantages and disadvantages of using Maude versus using Haskell to analyze security protocols [BD00]

- work of Millen and Denker at SRI using Maude to give a formal semantics to CAPSL, and to endow CAPSL with an execution and formal analysis environment [DM99a, DM99b, DM00a, DM00b];

- work of Wang, Gunter, and Meseguer using Maude to formally specify and analyze a PLAN active network algorithm [WMG00]

- work by Ölvecky et al. using Real-Time Maude to specify and analyze the AER/NCA suite of active network protocol components for reliable multicast [ÖKM+01]

- work of Verdejo, Pita, and Martí-Oliet on the Maude specification and verification of the FireWire leader election protocol [VPMO00];

- work of Mason and Talcott on modeling, simulation and analysis of network architectures and communication protocols [MT00]; and

- work of Pita and Martí-Oliet using the reflective features of Maude to specify some management processes of broadband telecommunication networks [PMO96, PMO99, PMO01].

Bioinformatics

Biology lacks at present adequate mathematical models that can provide something analogous to the analytic and predicitive power that mathematical models provide for, say, Physics.

Of course, the mathematical models of Chemistry describing, say, molecular structures are still applicable. The problem is that they do not scale up to something like a cell.

What are needed, and what rewriting logic contributes, are new computable mathematical models of cell biology that are at a high enough level of abstraction so that:

Bioinformatics (II)

- they fit biologist's intuitions and informal notations
- they make those intuitions mathematically precise
- they provide biologists with the predictive power of mathematical models, so that the consequences of their hypotheses and theories can be analyzed, and can then suggest laboratory experiments to prove them or disprove them.

Specifically, the approach of S. Eker, M. Knapp, K. Laderoute, P.Lincoln, J. Meseguer, K. Sonmez, and C. Talcott [pat02] develops computable models based on rewriting logic that fit very well the biologists' natural level of abstraction, and have good predictive and analytic capabilities.

References

[ADV01a] Antonio Albarrán, Francisco Durán, and Antonio Vallecillo. From Maude specifications to SOAP distributed implementations: A smooth transition. Manuscript, Universidad de Málaga, Spain, submitted for publication, 2001.

[ADV01b] Antonio Albarrán, Francisco Durán, and Antonio Vallecillo. Maude meets CORBA. In *Proceedings 2nd Argentine Symposium on Software Engineering*, Buenos Aires, Argentina, September 10–11, 2001.

[ADV01c] Antonio Albarrán, Francisco Durán, and Antonio Vallecillo. On the smooth implementation of component-based system specifications. In *Proceedings 6th ECOOP International Workshop on Component-Oriented Programming, WCOP'01*, Budapest, Hungary, June 2001.

[BCM] Jeffrey Van Baalen, James L. Caldwell, and Shivakant Mishra. Specifying and checking fault-tolerant agent-based protocols using Maude. In *First Goddard Workshop on Formal Approaches to Agent-Based Systems, Greenbelt, MD, USA, April 5–7, 2000, Proceedings*, Lecture Notes in Computer Science. To appear.

[BCM00] David Basin, Manuel Clavel, and José Meseguer. Rewriting logic as a metalogical framework. In S. Kapoor and S. Prasad, editors, *Twentieth Conference on the Foundations of Software Technology and Theoretical Computer Science, New Delhi, India, December 13–15, 2000, Proceedings*, volume 1974 of *Lecture Notes in Computer Science*, pages 55–80. Springer-Verlag, 2000.

[BD00] David Basin and Grit Denker. Maude versus Haskell: An experimental comparison in security protocol analysis. In Futatsugi [Fut00], pages 235–256. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[BHMM00] Christiano Braga, Hermann Haeusler, José Meseguer, and Peter Mosses. Maude Action Tool: Using reflection to map action semantics to rewriting logic. In T. Rus, editor, *Algebraic Methodology and Software Technology, 8th International Conference, AMAST 2000, Iowa City, Iowa, USA, May 20–27, 2000, Proceedings*, volume 1816 of *Lecture Notes in Computer Science*, pages 407–421. Springer-Verlag, 2000.

[BMM98] Roberto Bruni, José Meseguer, and Ugo Montanari. Process and term tile logic. Technical Report SRI-CSL-98-06, Computer Science Laboratory, SRI International, July 1998. Also TR-98-09, Dipartimento di Informatica, Università di Pisa. ftp://ftp.di.unipi.it/pub/techreports/TR-98-09.ps.Z.

[Bra01] C.O. Braga. *Rewriting Logic as a Semantic Framework for Modular Structural Operational Semantics*. PhD thesis, Departamento de Informática, Pontificia Universidade Católica de Rio de Janeiro, Brasil, 2001.

[Bru99] Roberto Bruni. *Tile Logic for Synchronized Rewriting of Concurrent Systems*. PhD thesis, Dipartimento di Informatica, Università di Pisa, 1999. Technical Report TD-1/99. http://www.di.unipi.it/phd/tesi/tesi_1999/TD-1-99.ps.gz.

[CDE+99] Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and José F. Quesada. Maude: Specification and programming in rewriting logic. Manual distributed as documentation of the Maude system, Computer Science Laboratory, SRI International. `http://maude.csl.sri.com/manual`, January 1999.

[CDEM00] Manuel Clavel, Francisco Durán, Steven Eker, and José Meseguer. Building equational proving tools by reflection in rewriting logic. In Futatsugi et al. [FNT00], pages 1–31. `http://maude.csl.sri.com/papers`.

[CDG98] Georgia Carabetta, Pierpaolo Degano, and Fabio Gadducci. CCS semantics via proved transition systems and rewriting logic. In Kirchner and Kirchner [KK98], pages 253–272. `http://www.elsevier.nl/locate/entcs/volume15.html`.

[CK99] Horatiu Cirstea and Claude Kirchner. Combining higher-order and first-order computations using ρ-calculus: Towards a semantics of ELAN. In D. Gabbay and M. de Rijke, editors, *Frontiers of*

Combining Systems 2, pages 95–121. Research Studies Press/Wiley, 1999.

[Cla98] Manuel Clavel. *Reflection in General Logics and in Rewriting Logic, with Applications to the Maude Language*. PhD thesis, Universidad de Navarra, Spain, February 1998.

[Cla00] Manuel Clavel. *Reflection in Rewriting Logic: Metalogical Foundations and Metaprogramming Applications*. CSLI Publications, 2000.

[CM01] Manuel Clavel and José Meseguer. Reflection in conditional rewriting logic. *Theoretical Computer Science*, 2001. This volume.

[CMP02] Manuel Clavel, José Meseguer, and Miguel Palomino. Reflection in membership equational logic, many-sorted equational logic, horn logic with equality, and rewriting logic. ENTCS, Elsevier, 2002. Proc. 4th. Intl. Workshop on Rewriting Logic and its Applications.

[DELM00] Francisco Durán, Steven Eker, Patrick Lincoln, and José Meseguer. Principles of Mobile Maude. In D. Kotz and F. Mattern, editors, *Agent Systems, Mobile Agents, and Applications, Second International Symposium on Agent Systems and Applications and Fourth International Symposium on Mobile Agents, ASA/MA 2000, Zurich, Switzerland, September 13–15, 2000, Proceedings*, volume 1882 of *Lecture Notes in Computer Science*, pages 73–85. Springer-Verlag, 2000.

[Den98] Grit Denker. From rewrite theories to temporal logic theories. In Kirchner and Kirchner [KK98], pages 273–294. `http://www.elsevier.nl/locate/entcs/volume15.html`.

[DGLAM+99a] Grit Denker, José J. García-Luna-Aceves, José Meseguer, Peter Csaba Ölveczky, Jyoti Raju, Brad Smith, and Carolyn L. Talcott. Specification and analysis of a reliable broadcasting protocol in Maude. In B. Hajek and R. S. Sreenivas, editors, *Proceedings 37th Allerton Conference on Communication, Control and Computation*, pages 738–747. University of Illinois, 1999. `http://maude.csl.sri.com/casestudies/rbp`.

[DGLAM+99b] Grit Denker, José J. García-Luna-Aceves, José Meseguer, Peter Csaba Ölveczky, Jyoti Raju, Brad Smith, and Carolyn L. Talcott. Specifying a reliable broadcasting protocol in Maude. Technical report, Computer Science Laboratory, SRI International, 1999. `http://www.csl.sri.com/casestudies/rbp`.

[DGP00] Pierpaolo Degano, Fabio Gadducci, and Corrado Priami. A causal semantics for CCS via rewriting logic. Manuscript, Dipartimento di Informatica, Università di Pisa, submitted for publication, 2000.

[Dia96] Răzvan Diaconescu. Foundations of behavioural specification in rewriting logic. In Meseguer [Mes96a], pages 225–244. http://www.elsevier.nl/locate/entcs/volume4.html.

[DM99a] Grit Denker and Jon Millen. CAPSL and CIL language design: A common authentication protocol specification language and its intermediate language. Technical Report SRI-CSL-99-02, Computer Science Laboratory, SRI International, 1999. http://www.csl.sri.com/~denker/pub_99.html.

[DM99b] Grit Denker and Jon Millen. CAPSL intermediate language. In N. Heintze and E. Clarke, editors, *Proceedings of Workshop on Formal Methods* *and Security Protocols, FMSP'99, July 1999, Trento, Italy*, 1999. http://www.cs.bell-labs.com/who/nch/fmsp99/program.html.

[DM00a] Grit Denker and Jon Millen. CAPSL integrated protocol environment. In D. Maughan, G. Koob, and S. Saydjari, editors, *Proceedings DARPA Information Survivability Conference and Exposition, DISCEX 2000, Hilton Head Island, South Carolina, January 25-27, 2000*, pages 207–222. IEEE Computer Society Press, 2000. http://schafercorp-ballston.com/discex/.

[DM00b] Grit Denker and Jon Millen. The CAPSL integrated protocol environment. Technical Report SRI-CSL-2000-02, Computer Science Laboratory, SRI International, 2000. http://www.csl.sri.com/~denker/pub_99.html.

[DMT98] Grit Denker, José Meseguer, and Carolyn L. Talcott. Protocol specification and analysis in Maude. In N. Heintze and J. Wing, editors, *Proceedings of Workshop on Formal Methods and Security Protocols, June 25, 1998, Indianapolis, Indiana*, 1998. `http://www.cs.bell-labs.com/who/nch/fmsp/index.html`.

[DMT00a] Grit Denker, José Meseguer, and Carolyn L. Talcott. Formal specification and analysis of active networks and communication protocols: The Maude experience. In D. Maughan, G. Koob, and S. Saydjari, editors, *Proceedings DARPA Information Survivability Conference and Exposition, DISCEX 2000, Hilton Head Island, South Carolina, January 25–27, 2000*, pages 251–265. IEEE Computer Society Press, 2000. `http://schafercorp-ballston.com/discex/`.

[DMT00b] Grit Denker, José Meseguer, and Carolyn L. Talcott. Rewriting semantics of meta-objects and composable distributed services. In Futatsugi [Fut00], pages 407–427. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[Dur00] Francisco Durán. Termination checker and Knuth-Bendix completion tools for Maude equational specifications. Manuscript, Computer Science Laboratory, SRI International, `http://maude.csl.sri.com/papers`, 2000.

[DV01] Francisco Durán and Antonio Vallecillo. Writing ODP enterprise specifications in Maude. In *Proceedings Workshop On Open Distributed Processing: Enterprise, Computation, Knowledge, Engineering and Realisation, WOODPECKER 2001*, Setúbal, Portugal, July 2001. `http://www.lcc.uma.es/~av/Publicaciones/01/ITI-2001-8.pdf`.

[EMS02] Steven Eker, José Meseguer, and Ambarish Sridharanarayanan. The Maude LTL model checker. ENTCS, Elsevier, 2002. Proc. 4th. Intl. Workshop on Rewriting Logic and its Applications.

[FM00] Gianluigi Ferrari and Ugo Montanari. Tile formats for located and mobile systems. *Information and Computation*, 156(1/2):173–235, 2000.

[FMMO+00] José Luiz Fiadeiro, Tom Maibaum, Narciso Martí-Oliet, José Meseguer, and Isabel Pita. Towards a verification logic for rewriting logic. In D. Bert, C. Choppy, and P. Mosses, editors, *Recent Trends in Algebraic Development Techniques, 14th International Workshop, WADT'99, Chateau de Bonas, France, September 15–18, 1999, Selected Papers*, volume 1827 of *Lecture Notes in Computer Science*, pages 438–458. Springer-Verlag, 2000.

[FNT00] Kokichi Futatsugi, Ataru T. Nakagawa, and Tetsuo Tamai, editors. *Cafe: An Industrial-Strength Algebraic Formal Method*. Elsevier, 2000.

[FR01] Bernd Fischer and Grigore Roșu. Interpreting abstract interpretations in membership equational logic. Technical Report RIACS 01.16, Research Institute for Advanced Computer Science, May 2001.

[FT00] José Luis Fernández and Ambrosio Toval. Can intuition become rigorous? Foundations for UML model verification tools. In F. M. Titsworth, editor, *International Symposium on Software Reliability Engineering*, pages 344–355, San Jose, California, October 8–11, 2000. IEEE Press.

[FT01] José Luis Fernández and Ambrosio Toval. Seamless formalizing the UML semantics through metamodels. In K. Siau and T. Halpin, editors, *Unified Modeling Language: Systems Analysis, Design, and Development Issues*, pages 224–248. Idea Group Publishing, 2001.

[Fut00] Kokichi Futatsugi, editor. *Proceedings Third International Workshop on Rewriting Logic and its Applications, WRLA 2000, Kanazawa, Japan, September 18–20, 2000*, volume 36 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2000. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[GM96] Fabio Gadducci and Ugo Montanari. Tiles, rewriting rules, and CCS. In Meseguer [Mes96a], pages 1–19. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[GM00] Fabio Gadducci and Ugo Montanari. The tile model. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays* *in Honour of Robin Milner*. The MIT Press, 2000. `http://www.di.unipi.it/~ugo/festschrift.ps`.

[GM01] Fabio Gadducci and Ugo Montanari. Comparing logics for rewriting: Rewriting logic, action calculi and tile logic. *Theoretical Computer Science*, 2001. This volume.

[Har00a] Neal A. Harman. Correctness and verification of hardware systems using Maude. Technical Report 3-2000, Department of Computer Science, University of Wales Swansea, 2000. `http://www-compsci.swan.ac.uk/reports/yr2000/CSR3-2000.pdf`.

[Har00b] Neal A. Harman. Verifying a simple pipelined microprocessor using Maude. Technical Report 4-2000, Department of Computer Science, University of Wales Swansea, 2000.

`http://www-compsci.swan.ac.uk/reports/yr2000/CSR4-2000.pdf`.

[HR01] Klaus Havelund and Grigore Roşu. Java PathExplorer — A runtime verification tool. In *Proceedings 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space, ISAIRAS'01*, Montreal, Canada, June 18–22, 2001.

[IFW] Hiroshi Ishikawa, Kokichi Futatsugi, and Takuo Watanabe. An operational semantics of GAEA in CafeOBJ. pages 213–227.

[IMW+97] Hiroshi Ishikawa, José Meseguer, Takuo Watanabe, Kokichi Futatsugi, and Hideyuki Nakashima. On the semantics of GAEA — An object-oriented specification of a concurrent reflective language in rewriting logic. In *Pro-*

ceedings IMSA'97, pages 70–109. Information-Technology Promotion Agency, Japan, 1997.

[KK98] Claude Kirchner and Hélène Kirchner, editors. *Proceedings Second International Workshop on Rewriting Logic and its Applications, WRLA'98, Pont-à-Mousson, France, September 1–4, 1998*, volume 15 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 1998. `http://www.elsevier.nl/locate/entcs/volume15.html`.

[Kna00] Alexander Knapp. Generating rewrite theories from UML collaborations. In Futatsugi et al. [FNT00], pages 97–120.

[Kna01] Alexander Knapp. *A Formal Approach to Object-Oriented Software Engineering*. Shaker Verlag, Aachen, Germany, 2001. PhD thesis, Institut für Informatik, Universität München, 2000.

[Lec97] Ulrike Lechner. *Object-Oriented Specification of Distributed Systems*. PhD thesis, Fakultät für Mathematik und Informatik, Universität Passau, June 1997.

[LM92] Cosimo Laneve and Ugo Montanari. Axiomatizing permutation equivalence in the λ-calculus. In H. Kirchner and G. Levi, editors, *Algebraic and Logic Programming, Third International Conference, Volterra, Italy, September 2–4, 1992, Proceedings*, volume 632 of *Lecture Notes in Computer Science*, pages 350–363. Springer-Verlag, 1992.

[LM96] Cosimo Laneve and Ugo Montanari. Axiomatizing permutation equivalence. *Mathematical Structures in Computer Science*, 6(3):219–249, June 1996.

[LN00] Martin Leucker and Thomas Noll. Rewriting logic as a framework for generic verification tools. In Futatsugi [Fut00], pages 117–133. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[Mes92] José Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.

[Mes93] José Meseguer. A logical theory of concurrent objects and its realization in the Maude language. In G. Agha, P. Wegner, and A. Yonezawa, editors, *Research Directions in Concurrent Object-Oriented Programming*, pages 314–390. The MIT Press, 1993.

[Mes96a] José Meseguer, editor. *Proceedings First International Workshop on Rewriting Logic and its Applications, WRLA'96, Asilomar, California,* *September 3–6, 1996*, volume 4 of *Electronic Notes in Theoretical Computer Science*. Elsevier, September 1996. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[Mes96b] José Meseguer. Rewriting logic as a semantic framework for concurrency: A progress report. In U. Montanari and V. Sassone, editors, *CONCUR'96: Concurrency Theory, 7th International Conference, Pisa, Italy, August 26–29, 1996, Proceedings*, volume 1119 of *Lecture Notes in Computer Science*, pages 331–372. Springer-Verlag, 1996.

[MFW92] José Meseguer, Kokichi Futatsugi, and Timothy Winkler. Using rewriting logic to specify, program, integrate, and reuse open concurrent systems of cooperating agents. In *Proceedings of the 1992 International Symposium on New Models for Software Architecture*, pages 61–106, Tokyo, Japan, November 1992. Research Institute of Software Engineering.

[MM98] José Meseguer and Ugo Montanari. Mapping tile logic into rewriting logic. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT'97, Tarquinia, Italy, June 3–7, 1997, Selected Papers*, volume 1376 of *Lecture Notes in Computer Science*, pages 62–91. Springer-Verlag, 1998.

[MOM93] Narciso Martí-Oliet and José Meseguer. Rewriting logic as a logical and semantic framework. Technical Report SRI-CSL-93-05, SRI International, Computer Science Laboratory, August 1993. To appear in D. Gabbay, editor, *Handbook of Philosophical Logic, Second Edition, Volume 6*, Kluwer Academic Publishers, 2001. `http://maude.csl.sri.com/papers`.

[Mos98] Peter D. Mosses. Semantics, modularity, and rewriting logic. In Kirchner and Kirchner [KK98]. `http://www.elsevier.nl/locate/entcs/volume15.html`.

[Mos99] Peter D. Mosses. Logical specification of operational semantics. In J. Flum and M. Rodríguez-Artalejo, editors, *Computer Science Logic, 13th International Workshop, CSL'99, 8th Annual Conference of the EACSL, Madrid, Spain, September 20–25, 1999, Proceedings*, volume 1683 of *Lecture Notes in Computer Science*, pages 32–49. Springer-Verlag, 1999.

[MR02] J. Meseguer and G. Roşu. Towards behavioral Maude: Behavioral membership equational logic. In *Proc. CMCS'02*. Elsevier ENTCS, 2002.

[MT] José Meseguer and Carolyn L. Talcott. Using rewriting logic to interoperate architectural description languages (I and II). Lectures at the Santa Fe and Seattle *DARPA-EDCS Workshops*, March and July 1997. http://www-formal.stanford.edu/clt/ArpaNsf/adl-interop.html.

[MT97] Ian A. Mason and Carolyn L. Talcott. A semantics preserving actor translation. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Automata, Languages and Programming, 24th International Colloquium, ICALP'97, Bologna, Italy, July 1997, Proceedings*, volume 1256 of *Lecture Notes in Computer Science*, pages 369–378. Springer-Verlag, 1997.

[MT99] José Meseguer and Carolyn L. Talcott. A partial order event model for concurrent objects. In J. C. M. Baeten and S. Mauw, editors, *CONCUR'99, Concurrency Theory, 10th International Conference Eindhoven, The Netherlands, August 24–27, 1999, Proceedings*, volume 1664 of *Lec-*

ture Notes in Computer Science, pages 415–430. Springer-Verlag, 1999.

[MT00] Ian A. Mason and Carolyn L. Talcott. Simple network protocol simulation within Maude. In Futatsugi [Fut00], pages 277–294. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[MT02] J. Meseguer and C. Talcott. Semantic models for distributed object reflection. In *Proceedings of ECOOP'02, Málaga, Spain, June 2002*. Springer LNCS 2374, 2002.

[Nak] Shin Nakajima. Encoding mobility in CafeOBJ: An exercise of describing mobile code-based software architecture.

[NF97] Shin Nakajima and Kokichi Futatsugi. An object-oriented modeling method for algebraic specifications in CafeOBJ. In *Proceedings, 19th International Conference on Software Engineering*, pages 34–44, Boston, Massachussets, May 1997. IEEE Computer Society Press.

[NS94] Elie Najm and Jean-Bernard Stefani. A formal operational semantics for the ODP computational model with signals, explicit binding, and reactive objects. Manuscript, ENST, Paris, France, 1994.

[NS95] Elie Najm and Jean-Bernard Stefani. A formal semantics for the ODP computational model. *Computer Networks and ISDN Systems*, 27:1305–1329, 1995.

[NS97] Elie Najm and Jean-Bernard Stefani. Computational models for open distributed systems. In H. Bowman and J. Derrick, editors, *Proceedings Second IFIP Conference on Formal Methods for Open Object-Based Distributed Systems, FMOODS'97, July 21–23, 1997, Canterbury,* *Kent, UK*, pages 157–176. Chapman & Hall, 1997.

[ÖKM+01] Peter Csaba Ölveczky, Mark Keaton, José Meseguer, Carolyn L. Talcott, and Steve Zabele. Specification and analysis of the AER/NCA active network protocol suite in Real-Time Maude. In H. Hussmann, editor, *Fundamental Approaches to Software Engineering, 4th International Conference, FASE 2001, Held as Part of ETAPS 2001, Genova, Italy, April 2001, Proceedings*, volume 2029 of *Lecture Notes in Computer Science*, pages 333–347. Springer-Verlag, 2001. `http://maude.csl.sri.com/papers`.

[Ölv00] Peter Csaba Ölveczky. *Specification and Analysis of Real-Time and Hybrid Systems in Rewriting Logic*. PhD thesis, University of Bergen, Norway, 2000. `http://maude.csl.sri.com/papers`.

[ÖM01] Peter Csaba Ölveczky and José Meseguer. Specification of real-time and hybrid systems in rewriting logic. *Theoretical Computer Science*, 2001. This volume.

[Pat00] Dirk Pattinson. Modal logic for rewriting theories. In Futatsugi [Fut00], pages 173–191. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[pat02] *Pathway Logic: Symbolic Analysis of Biological Signaling*, January 2002.

[PMO96] Isabel Pita and Narciso Martí-Oliet. A Maude specification of an object oriented database model for telecommunication networks. In Meseguer [Mes96a], pages 404–422. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[PMO99] Isabel Pita and Narciso Martí-Oliet. Using reflection to specify transaction sequences in rewriting logic. In J. L. Fiadeiro, editor, *Recent Trends in Algebraic Development Techniques, 13th International Workshop, WADT'98, Lisbon, Portugal, April 2–4, 1998, Selected Papers*, volume 1589 of *Lecture Notes in Computer Science*, pages 261–276. Springer-Verlag, 1999.

[PMO01] Isabel Pita and Narciso Martí-Oliet. A Maude specification of an object-oriented model for telecommunication networks. *Theoretical Computer Science*, 2001. This volume.

[Rod00] Dilia E. Rodríguez. Case studies in the specification and analysis of protocols in Maude. In Futatsugi [Fut00], pages 257–275. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[SM99] Mark-Oliver Stehr and José Meseguer. Pure type systems in rewriting logic. In *Proceedings of LFM'99: Workshop on Logical Frameworks* *and Meta-languages*, Paris, France, September 28, 1999. http://www.cs.bell-labs.com/~felty/LFM99/.

[SMÖ01] Mark-Oliver Stehr, José Meseguer, and Peter Csaba Ölveczky. Rewriting logic as a unifying framework for Petri nets. In H. Ehrig, G. Juhas, J. Padberg, and G. Rozenberg, editors, *Unifying Petri Nets*, Lecture Notes in Computer Science. Springer-Verlag, 2001. To appear.

[Ste00] Mark-Oliver Stehr. CINNI — A generic calculus of explicit substitutions and its application to λ-, ς- and π-calculi. In Futatsugi [Fut00], pages 71–92. http://www.elsevier.nl/locate/entcs/volume36.html.

[Ste01a] L. Jason Steggles. Rewriting logic and Elan: Prototyping tools for Petri nets with time. In J.-M. Colom and M. Koutny, editors, *Applica-*

tions and Theory of Petri Nets 2001, 22nd International Conference, ICATPN 2001, Newcastle upon Tyne, UK, June 25–29, 2001, Proceedings, volume 2075 of *Lecture Notes in Computer Science*, pages 363–381. Springer-Verlag, 2001.

[Ste01b] Mark-Oliver Stehr. A rewriting semantics for algebraic nets. In C. Girault and R. Valk, editors, *Petri Nets for System Engineering — A Guide to Modeling, Verification, and Applications*. Springer-Verlag, 2001.

[Ste02] Mark-Oliver Stehr. *Rewriting Logic and Type Theory — From Applications to Unification*. PhD thesis, Computer Science Department, University of Hamburg, Germany, 2002.

[Tal96] Carolyn L. Talcott. An actor rewriting theory. In Meseguer [Mes96a], pages 360–383. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[Tal97a] Carolyn L. Talcott. Composable semantic models for actor theories. In M. Abadi and T. Ito, editors, *Theoretical Aspects of Computer Science, Third International Symposium, TACS'97, Sendai, Japan, September 23–26, 1997, Proceedings*, volume 1281 of *Lecture Notes in Computer Science*, pages 321–364. Springer-Verlag, 1997.

[Tal97b] Carolyn L. Talcott. Interaction semantics for components of distributed systems. In E. Najm and J.-B. Stefani, editors, *Proceedings IFIP Conference on Formal Methods for Open Object-Based Distributed Systems, FMOODS'96*, pages 154–169. Chapman & Hall, 1997.

[Tal00] Carolyn L. Talcott. Towards a toolkit for actor system specification. In T. Rus, editor, *Algebraic Methodology and Software Technology, 8th International Conference, AMAST 2000, Iowa*

City, Iowa, USA, May 20–27, 2000, Proceedings, volume 1816 of *Lecture Notes in Computer Science*, pages 391–406. Springer-Verlag, 2000.

[Tal01] Carolyn L. Talcott. Actor theories in rewriting logic. *Theoretical Computer Science*, 2001. This volume.

[TF00] Ambrosio Toval and José Luis Fernández. Formally modeling UML and its evolution: A holistic approach. In S. F. Smith and C. L. Talcott, editors, *Proceedings IFIP Conference on Formal Methods for Open Object-Based Distributed Systems IV, FMOODS 2000, September 6–8, 2000, Stanford, California, USA*, pages 183–206. Kluwer Academic Publishers, 2000.

[TF01] Ambrosio Toval and José Luis Fernández. Improving system reliability via rigorous software modeling: The UML case. In *Proceedings IEEE*

Aerospace Conference, Volume 6, pages 6–17, Big Sky, MT, USA, March 10–17, 2001. IEEE Press.

[Vir94] Patrick Viry. Rewriting: An effective model of concurrency. In C. Halatsis, D. Maritsas, G. Philokyprou, and S. Theodoridis, editors, *PARLE'94 Parallel Architectures and Languages Europe, 6th International PARLE Conference, Athens, Greece, July 4–8, 1994, Proceedings*, volume 817 of *Lecture Notes in Computer Science*, pages 648–660. Springer-Verlag, 1994.

[Vir96] Patrick Viry. Input/output for ELAN. In Meseguer [Mes96a], pages 51–64. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[Vir01] Patrick Viry. Equational rules for rewriting logic. *Theoretical Computer Science*, 2001. This volume.

[Vit94] Marian Vittek. *ELAN: Un Cadre Logique pour le Prototypage de Langages de Programmation avec Contraintes*. PhD thesis, Université Henri Poincaré – Nancy I, November 1994.

[VMO] Alberto Verdejo and Narciso Martí-Oliet. Executing E-LOTOS processes in Maude. In H. Ehrig, M. Grosse-Rhode, and F. Orejas, editors, *INT 2000, Integration of Specification Techniques with Applications in Engineering, Extended Abstracts*, pages 49–53. Technical report 2000/04, Technische Universitat Berlin, March 2000.

[VMO00a] Alberto Verdejo and Narciso Martí-Oliet. Executing and verifying CCS in Maude. Technical Report 99-00, Departamento de Sistemas Informáticos y Programación, Universidad Complutense de Madrid, February 2000. `http://maude.csl.sri.com/casestudies/ccs`.

[VMO00b] Alberto Verdejo and Narciso Martí-Oliet. Implementing CCS in Maude. In T. Bolognesi and D. Latella, editors, *Formal Methods For Distributed System Development. FORTE/PSTV 2000 IFIP TC6 WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE XIII) and Protocol Specification, Testing and Verification (PSTV XX) October 10–13, 2000, Pisa, Italy*, volume 183 of *International Federation for Information Processing*, pages 351–366. Kluwer Academic Publishers, 2000.

[VPMO00] Alberto Verdejo, Isabel Pita, and Narciso Martí-Oliet. The leader election protocol of IEEE 1394 in Maude. In Futatsugi [Fut00], pages 385–406. `http://www.elsevier.nl/locate/entcs/volume36.html`.

[WK96] Martin Wirsing and Alexander Knapp. A formal approach to object-oriented software engineering. In Meseguer [Mes96a], pages 321–359. `http://www.elsevier.nl/locate/entcs/volume4.html`.

[WK01] Martin Wirsing and Alexander Knapp. A formal approach to object-oriented software engineering. *Theoretical Computer Science*, 2001. This volume.

[WMG00] Bow-Yaw Wang, José Meseguer, and Carl A. Gunter. Specification and formal analysis of a PLAN algorithm in Maude. In P.-A. Hsiung, editor, *Proceedings International Workshop on Distributed System Validation and Verification, Taipei, Taiwan*, pages 49–56, April 2000.