Quantitative Modal Transition Systems

Kim Larsen, Axel Legay

Quantitative Modal Transition Systems (Invited Extended Abstract)

Kim G. Larsen[1] and Axel Legay[2]

[1] Aalborg University, Denmark, kgl@cs.aau.dk
[2] INRIA/IRISA, France, axel.legay@inria.fr

Abstract. This extended abstract offers a brief survey presentation of the specification formalism of modal transition systems and its recent extensions to the quantitative setting of timed as well as stochastic systems. Some applications will also be briefly mentioned.

1 Modal transition systems: the origines.

Modal transition systems [46] provids a behavioural compositional specification formalism for reactive systems. They grew out of the notion of relativized bisimulation, which allows for simple specifications of components by allowing the notion of bisimulation to take the restricted use that a given context may have in its.

A modal transition system is essentially a (labelled) transition system, but with two types of transitions: so-called *may* transitions, that any implementation may (or may not) have, and *must* transitions, that any implementation must have. In fact, ordinary labelled transition systems (or implementations) are modal transition systems where the set of may- and must-transitions coincide. Modal transition systems come equipped with a bisimulation-like notion of (modal) refinement, reflecting that the more must-transitions and the fewer may-transitions a modal specification has the more refined and closer to a final implementation it is.

Example 1. Consider the modal transition system shown in Figure 1 which models the requirements of a simple email system in which emails are first received and then delivered – must and may transitions are represented by solid and dashed arrows, respectively. Before delivering the email, the system may check or process the email, e.g. for en- or decryption, filtering of spam emails, or generating automatic answers using as an auto-reply feature. Any implementation of this email system specification must be able to receive and deliver email, and it may also be able to check arriving email before delivering it. No other behavior is allowed. Such an implementation is given in Figure 2.

Modal transition systems play a major role in various areas. However, the model is best known by its application in compositional reasoning, which has been recognized in the ARTIST Network of Excellence and several other related European projects. In fact, modal transition systems have all the ingredients of a complete compositinal specification theory allowing for logical compositions (e.g. conjunction) [42], structural compositions (e.g. parallel) [37] as well as quotienting permitting specifications

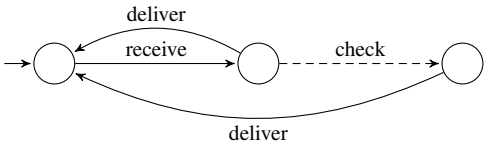

Fig. 1: Modal transition system modeling a simple email system, with an optional behavior: Once an email is received it may e.g. be scanned for containing viruses, or automatically decrypted, before it is delivered to the receiver.

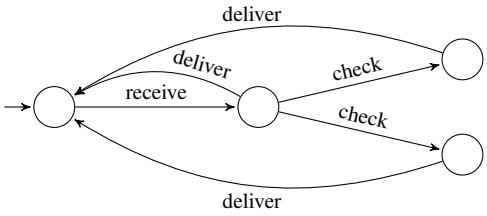

Fig. 2: An implementation of the simple email system in Figure 1 in which we explicitly model two distinct types of email pre-processing.

of composite systems to be transformed into necessary and sufficient specification of components [35]. Thus, modal transition systems have all the benefits of both logical and behavioural specification formalisms [18]. Though modal refinement – like bisimulation – is polynomial-time decidable for finite-state modal transition systems, it only provides a sound but not complete test for the true semantic refinement between modal specification, in terms of set inclusion between their implementation-sets (so-called thorough refinement). For several years, the complexity of thorough refinement – as well as the consistency – between modal specifications was an open problem, which after a series of attempts [44, 3] [2] was shown to be EXPTIME-complete [14].

In the rest of this overview, we will briefly introduce several quantitative extensions of modal transition systems that have been recently proposed in the literature. Sections 2 to 5 mainly focus on modal transition systems as a specification theory, while Sections 6 and 7 outline some other extensions.

2 Timed modal specifications

It is well acknowledged that real-time can be a crucial parameter in practice, for example in embedded systems. This motivates the study of extended modal transition systems to introduce real-time features.

Timed extensions of modal transitions were introduced early on [23] as timed extension of the process algebra CCS. Unfortunately the supporting tool EPSILON was entirely relying on the so-called region-abstraction, making scalability extremely poor. Most recently, taking advantage of the powerfull game-theoretical engine of UPPAAL

Tiga [10, 21] a "modal-transition system"-like compositional specification theory based on Timed I/O Automata [41] has been proposed [24]. ECDAR [25] gives an efficient tool support for refinement, consistency and quotienting for this theory.

In [27], de Alfaro et al. suggested *timed interfaces*, a model that is similar to the one of TIOTSs. Our definition of composition builds on the one proposed there. However, the work in [27] is incomplete. Indeed, there was no notion of implementation and refinement. Moreover, conjunction and quotient were not studied. Finally, the theory has only been implemented in a prototype tool which does not handle continuous time, while our contribution takes advantage of the powerful game engine of UPPAAL Tiga.

The work of [16] suggests an alternative timed extension of modal transition systems (though still relying on regions for refinement algorithms). This work is less elaborated and implementation was not considered there.

3 Weighted modal specifications

The previous section was concerned with modal transition systems whose implementations are timed automata. There are various other extensions of automata, among which one finds weighted automata, that are classical automata whose transitions are equipped with integer weights. In [40], modal transition systems were extended with integer intervals in order to capture and abstract weighted automata. The works also proposes structural and logical composition as well as refinement for the extended model. Latter, in [7], the work was generalized to weighted modal transition systems, whose transition can be equipped with any type of quantity.

Albeit the extensions mentioned above allow for a quantitative treatment of automata behaviors, the operations on weighted modal transition systems remain qualitative. Especially, the refinement relation of modal transition systems is qualitative in nature, i.e. an implementation does, or does not, refine a specification. Such a view may be fragile in the sense that the inevitable approximation of systems by models, combined with the fundamental unpredictability of hardware platforms, make difficult to transfer conclusions about the behavior to the actual system. Hence, this approach is arguably unsuited for modern software systems. In [5], the first quantitative extension of modal automata was proposed. This model allows to capture quantitative aspects during the refinement and implementation process, thus leveraging the problems of the qualitative setting.

In [5], satisfaction and refinement are lifted from the well-known qualitative setting to the quantitative setting, by introducing a notion of distance between weighted modal transition systems. It is also shown that quantitative versions of parallel composition, as well as quotient (the dual operator to parallel composition), inherit the properties from the Boolean setting.

Example 2 (taken from [5]). Consider again the modal transition system of Figure 1, but this time with quantities, see Figure 3: Every transition label is extended by an integer intervals modeling upper and lower bounds on the time required for performing the corresponding actions. For instance, the reception of a new email (action *receive*) must take between one and three time units, the checking of the email (action *check*) is allowed to take up to five time units.

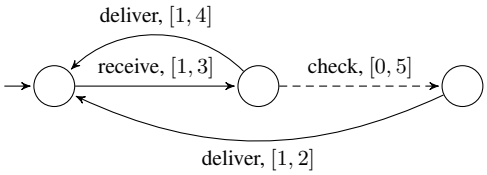

Fig. 3: Specification of a simple email system, similar to Figure 1, but extended by integer intervals modeling time units for performing the corresponding actions.

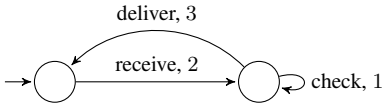

Fig. 4: Implementation a)

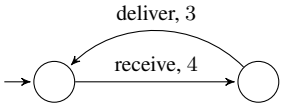

Fig. 5: Implementation b)

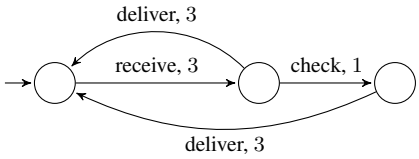

Fig. 6: Implementation c)

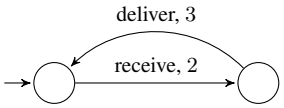

Fig. 7: Implementation d)

Four implementations of the simple email system in Figure 3.

In this quantitative setting, there is a problem with using a *Boolean* notion of refinement: If one only can decide *whether or not* an implementation refines a specification, then the quantitative aspects get lost in the refinement process. As an example, consider the email system implementations in Figures 4 to 7. Implementation (a) does not refine the specification, as there is an error in the discrete structure of actions: after receiving an email, the system can check it indefinitely without ever delivering it. Also, implementations (b) and (c) do not refine the specification: (b) takes too long to receive any email, while (c) does not deliver the email fast. Implementation (d), on the other hand, is a perfect refinement of the specification.

The work in [5] uses an accumulating distance, but the contribution was latter generalized to any type of distance in [6]. The extended model allowed, as an example, to define various notions of robustness for specification theories. Complexity of refinement and efficient implementations remain open problems.

4 Probabilistic modal specifications

In [39], modal transitions systems were extended into a specification formalism for Markov Chains by the introduction of so-called probabilistic specifications (now known

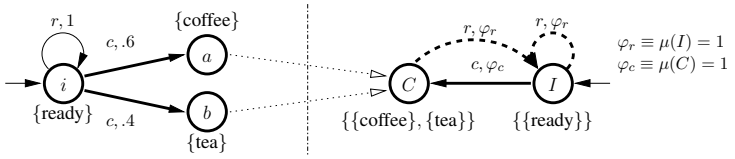

Fig. 8: Implementation PA and specification APA of a coffee machine.

as Interval Markov Chains), where concrete probabilities are replaced with intervals, and with refinement providing a conservative extension or probabilistic bisimulation [45]. However, Interval Markov Chains lack several of the properties required for a complete compositional specification theory; in particular, they are not closed neither under logical nor structural composition. Recently, the extended notion of Constraint Markov Chains [19] was introduced precisely with the purpose of providing these closure properties. A Constraint Markov Chain (CMC) is a Markov Chain (MC) equipped with a constraint on the next-state probabilities from any state. Roughly speaking, an implementation for a CMC is thus a MC, whose next-state probability distribution satisfies the constraint associated with each state. The power of constrains can be exploited to obtain closure under any logical/structural composition operation. The complexity of the refinement relation largely depends on the one to solve the constraints – it is at least quadratic (resp. exponential) for syntactic (resp. thorough) refinement. The reader interested in decision probblems for CMCs is redirected to [32, 31]

More recently, the concept of CMC was extended to offer abstractions for Probabilistic Automata (PA), i.e., structures that mix both stochastic and non-deterministic aspects. The work in [28] proposes Abstract Probabilistic Automata, that are a combination of modal transition systems and CMCs, modalities being used to capture the non-determinism in PAs. The model was implemented in the APAC toolset [30] and various decision problems, including stuttering and abstraction, were studied in [29, 54].

Example 3 (taken from [29]). Consider the implementation (left) and specification (right) of a coffee machine given in Figure 8. The specification indicates that there are two possible transitions from initial state I: a may transition labeled with action r (reset) and a must transition labeled with action c (coin). May transitions are represented with dashed arrows and must transitions are represented with plain arrows. The probability distributions associated with these actions are specified by the constraints φ_r and φ_c, respectively.

5 Beyond modalities

In a seminal paper [26], de Alfaro and Henzinger promoted another specification theory known under the name of *interface automata*. An interface is represented by an input/output automaton [47], i.e., an automaton whose transitions are labeled with *input* or *output* actions. The semantics of such an automaton is given by a two-players game: an *Input* player represents the environment, and an *Output* player represents the

component itself. Interface automata do not encompass any notion of model, because there is no way to distinguish between an interface and its implementations.

Refinement between interface automata corresponds to the alternating refinement relation between games [1], i.e., an interface refines another if its environment is more permissive, whereas its component is more restrictive. Contrary to most interfaces theories, the game-based interpretation offers an *optimistic* treatment of composition: two interfaces can be composed if there exists at least one environment (i.e., one strategy for the Input player) in which they can interact together in a safe way (i.e., whatever the strategy of the Output player is). This is referred to as the compatibility of interfaces. A quotient, which is the adjoint of the game-based composition, has been proposed in [17] for the deterministic case.

In [43, 52], modal transition systems and interface automata were combined to give rise to modal interfaces, a model that offers both the power of modalities and the optimistic composition approach of interface automata through labeling of may and must modalities with input/output features. Implementations of modal interfaces can be find in the Mika toolset [20] and the MIOs workbench [9].

It is worth mentioning that the methodology used in [52] is rather generic and can be applied to other extensions of modal automata. As an example, the APA model was also extended to a modal APA model in [29].

6 Contract Theory

We already observed that modal transition systems act as a good model for a complete specification theory. Of course, there are other similar approaches. In fact, some of them advocate that specification theories should be equipped with additional structure that makes more explicit their possible connections. This is particularly the case of the contract theory approach, which is based on a assume-guarantee (AG) reasoning.

Concretely, contract theories differ from classical specification theories in that they strictly follow the principle of separation of concerns. They separate the specification of assumptions from the specification of guarantees, a choice largely inspired by early ideas on manual proof methods by Misra, Chandy [49] and Jones [38], along with the wide acceptance of the pre-/post-condition specification in programming [48, 55], and more in general semantical rules independent from language representation [22].

Recently, Benveniste et al [15] proposed a contract theory where assumptions and guarantees are represented by trace structures. While this work is of clear interest, it suffers from the absence of an effective representation for the embedded interface theory. Some extensions, such as the one proposed in [50, 35], leverage this problem but in a specific manner, i.e., just for the case of a single theory.

In [4], it was shown how a theory of contracts can be built on top of a given abstract specification theory. Contracts are just pairs (A, G) of an assumption and a guarantee specification. Particularly, it was shown how the contract theory can be instantiated by using modal transition systems.

7 Others modal extensions and applications

There are many other extensions of modal transition systems, which include those that encompass unbounded data or costs and parameters [8, 12, 13], and those that offer a more elaborated treatment of modalities [51, 11].

In addition to their contribution to specification theories, modal transition systems have also played a major role in abstraction-based model checking [33, 34], software differences [53], and in the design of efficient approaches for software product lines verification [36].

References

1. Rajeev Alur, Thomas A. Henzinger, Orna Kupferman, and Moshe Y. Vardi. Alternating refinement relations. In *CONCUR*, volume 1466 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 1998.
2. Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. 20 years of modal and mixed specifications. *Bulletin of the EATCS*, 95:94–129, 2008.
3. Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. Modal and mixed specifications: key decision problems and their complexities. *Mathematical Structures in Computer Science*, 20(1):75–103, 2010.
4. Sebastian S. Bauer, Alexandre David, Rolf Hennicker, Kim Guldstrand Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Moving from specifications to contracts in component-based design. In *FASE*, pages 43–58, 2012.
5. Sebastian S. Bauer, Uli Fahrenberg, Line Juhl, Kim G. Larsen, Axel Legay, and Claus R. Thrane. Quantitative refinement for weighted modal transition systems. In *MFCS*, volume 6907 of *Lecture Notes in Computer Science*, pages 60–71. Springer, 2011.
6. Sebastian S. Bauer, Uli Fahrenberg, Axel Legay, and Claus R. Thrane. General quantitative specification theories with modalities. In *CSR*, volume 7353 of *Lecture Notes in Computer Science*, pages 18–30. Springer, 2012.
7. Sebastian S. Bauer, Line Juhl, Kim G. Larsen, Axel Legay, and Jirí Srba. Extending modal transition systems with structured labels. *Mathematical Structures in Computer Science*, 22(4):581–617, 2012.
8. Sebastian S. Bauer, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. A modal specification theory for components with data. In *FACS*, Lecture Notes in Computer Science. Springer, 2011. to appear in post proceedings.
9. Sebastian S. Bauer, Philip Mayer, and Axel Legay. Mio workbench: A tool for compositional design with modal input/output interfaces. In *ATVA*, volume 6996 of *Lecture Notes in Computer Science*, pages 418–421. Springer, 2011.
10. Gerd Behrmann, Agnès Cougnard, Alexandre David, Emmanuel Fleury, Kim Guldstrand Larsen, and Didier Lime. Uppaal-tiga: Time for playing games! In Werner Damm and Holger Hermanns, editors, *CAV*, volume 4590 of *Lecture Notes in Computer Science*, pages 121–125. Springer, 2007.
11. Nikola Benes and Jan Kretínský. Modal process rewrite systems. In *ICTAC*, volume 7521 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2012.
12. Nikola Benes, Jan Kretínský, Kim G. Larsen, Mikael H. Møller, and Jirí Srba. Parametric modal transition systems. In *ATVA*, volume 6996 of *Lecture Notes in Computer Science*, pages 275–289. Springer, 2011.

13. Nikola Benes, Jan Kretínský, Kim Guldstrand Larsen, Mikael H. Møller, and Jirí Srba. Dual-priced modal transition systems with time durations. In *LPAR*, volume 7180 of *Lecture Notes in Computer Science*, pages 122–137. Springer, 2012.
14. Nikola Benes, Jan Kretínský, Kim Guldstrand Larsen, and Jirí Srba. Checking thorough refinement on modal transition systems is exptime-complete. In Martin Leucker and Carroll Morgan, editors, *ICTAC*, volume 5684 of *Lecture Notes in Computer Science*, pages 112–126. Springer, 2009.
15. Albert Benveniste, Benoît Caillaud, Alberto Ferrari, Leonardo Mangeruca, Roberto Passerone, and Christos Sofronis. Multiple viewpoint contract-based specification and design. In *FMCO*, volume 5382 of *LNCS*, pages 200–225. Springer, 2007.
16. Nathalie Bertrand, Axel Legay, Sophie Pinchinat, and Jean-Baptiste Raclet. A compositional approach on modal specifications for timed systems. In Karin Breitman and Ana Cavalcanti, editors, *ICFEM*, volume 5885 of *Lecture Notes in Computer Science*, pages 679–697. Springer, 2009.
17. P. Bhaduri. Synthesis of interface automata. In *xATVA*, volume 3707 of *Lecture Notes in Computer Science*, pages 338–353. Springer, 2005.
18. Gérard Boudol and Kim Guldstrand Larsen. Graphical versus logical specifications. In André Arnold, editor, *CAAP*, volume 431 of *Lecture Notes in Computer Science*, pages 57–71. Springer, 1990.
19. Benoît Caillaud, Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Constraint markov chains. *Theor. Comput. Sci.*, 412(34):4373–4404, 2011.
20. Benoît Caillaud and Jean-Baptiste Raclet. Ensuring reachability by design. In *ICTAC*, volume 7521 of *Lecture Notes in Computer Science*, pages 213–227. Springer, 2012.
21. Franck Cassez, Alexandre David, Emmanuel Fleury, Kim Guldstrand Larsen, and Didier Lime. Efficient on-the-fly algorithms for the analysis of timed games. In Martín Abadi and Luca de Alfaro, editors, *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2005.
22. Antonio Cau and Pierre Collette. Parallel composition of assumption-commitment specifications: A unifying approach for shared variable and distributed message passing concurrency. *Acta Inf.*, 33(2):153–176, 1996.
23. Karlis Cerans, Jens Chr. Godskesen, and Kim Guldstrand Larsen. Timed modal specification - theory and tools. In Costas Courcoubetis, editor, *CAV*, volume 697 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 1993.
24. Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Timed i/o automata: a complete specification theory for real-time systems. In Karl Henrik Johansson and Wang Yi, editors, *HSCC*, pages 91–100. ACM ACM, 2010.
25. Alexandre David, Kim Guldstrand Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Ecdar: An environment for compositional design and analysis of real time systems. In Ahmed Bouajjani and Wei-Ngan Chin, editors, *ATVA*, volume 6252 of *Lecture Notes in Computer Science*, pages 365–370. Springer, 2010.
26. Luca de Alfaro and Thomas A. Henzinger. Interface theories for component-based design. In *EMSOFT*, volume 2211 of *Lecture Notes in Computer Science*, pages 148–165. Springer, 2001.
27. Luca de Alfaro, Thomas A. Henzinger, and Mariëlle Stoelinga. Timed interfaces. In *EMSOFT*, volume 2491 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2002.
28. Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract probabilistic automata. In Ranjit Jhala and David A. Schmidt, editors, *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 324–339. Springer, 2011.

29. Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. New results on abstract probabilistic automata. In *ACSD*, pages 118–127. IEEE, 2011.
30. Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Apac: A tool for reasoning about abstract probabilistic automata. In *QEST*, pages 151–152. IEEE Computer Society, 2011.
31. Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Decision problems for interval markov chains. In Adrian Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, *LATA*, volume 6638 of *Lecture Notes in Computer Science*, pages 274–285. Springer, 2011.
32. Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Consistency and refinement for interval markov chains. *J. Log. Algebr. Program.*, 81(3):209–226, 2012.
33. Patrice Godefroid. Reasoning about abstract open systems with generalized module checking. In *EMSOFT*, volume 2855 of *Lecture Notes in Computer Science*, pages 223–240. Springer, 2003.
34. Patrice Godefroid. Generalized model checking. In *TIME*, page 3. IEEE Computer Society, 2005.
35. Gregor Goessler and Jean-Baptiste Raclet. Modal contracts for component-based design. In *SEFM*, pages 295–303. IEEE Computer Society, 2009.
36. Alexander Gruler, Martin Leucker, and Kathrin D. Scheidemann. Modeling and model checking software product lines. In *FMOODS*, volume 5051 of *Lecture Notes in Computer Science*, pages 113–131. Springer, 2008.
37. Hans Hüttel and Kim Guldstrand Larsen. The use of static constructs in a modal process logic. In Albert R. Meyer and Michael A. Taitslin, editors, *Logic at Botik*, volume 363 of *Lecture Notes in Computer Science*, pages 163–180. Springer, 1989.
38. Cliff B. Jones. *Development methods for computer programs including a notion of interference*. PhD thesis, Oxford University Computing Laboratory, 1981.
39. Bengt Jonsson and Kim Guldstrand Larsen. Specification and refinement of probabilistic processes. In *LICS*, pages 266–277, 1991.
40. Line Juhl, Kim G. Larsen, and Jirí Srba. Modal transition systems with weight intervals. *J. Log. Algebr. Program.*, 81(4):408–421, 2012.
41. Dilsun Kirli Kaynar, Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. *The Theory of Timed I/O Automata, Second Edition*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2010.
42. Kim Guldstrand Larsen. Modal specifications. In Joseph Sifakis, editor, *Automatic Verification Methods for Finite State Systems*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1989.
43. Kim Guldstrand Larsen, Ulrik Nyman, and Andrzej Wasowski. Modal I/O automata for interface and product line theories. In *ESOP*, volume 4421 of *Lecture Notes in Computer Science*, pages 64–79. Springer, 2007.
44. Kim Guldstrand Larsen, Ulrik Nyman, and Andrzej Wasowski. On modal refinement and consistency. In Luís Caires and Vasco Thudichum Vasconcelos, editors, *CONCUR*, volume 4703 of *Lecture Notes in Computer Science*, pages 105–119. Springer, 2007.
45. Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991.
46. Kim Guldstrand Larsen and Bent Thomsen. A modal process logic. In *LICS*, pages 203–210, 1988.
47. Nancy Lynch and Mark R. Tuttle. An introduction to Input/Output automata. *CWI-quarterly*, 2(3), 1989.

48. Bertrand Meyer. Applying "design by contract". *IEEE Computer*, 25(10):40–51, 1992.
49. Jayadev Misra and K. Mani Chandy. Proofs of networks of processes. *IEEE Trans. Software Eng.*, 7(4):417–426, 1981.
50. Sophie Quinton and Susanne Graf. Contract-based verification of hierarchical systems of components. In *SEFM*, pages 377–381. IEEE Computer Society, 2008.
51. Jean-Baptiste Raclet. Residual for component specifications. *Electr. Notes Theor. Comput. Sci.*, 215:93–110, 2008.
52. Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. A modal interface theory for component-based design. *Fundam. Inform.*, 108(1-2):119–149, 2011.
53. Mathieu Sassolas, Marsha Chechik, and Sebastián Uchitel. Exploring inconsistencies between modal transition systems. *Software and System Modeling*, 10(1):117–142, 2011.
54. Falak Sher and Joost-Pieter Katoen. Compositional abstraction techniques for probabilistic automata. In *IFIP TCS*, volume 7604 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2012.
55. Qiwen Xu, Antonio Cau, and Pierre Collette. On unifying assumption-commitment style proof rules for concurrency. In *CONCUR*, pages 267–282, 1994.