CADP'97 – Status, Applications and Perspectives

H. Garavel, M. Jorgensen, R. Mateescu, Ch. Pecheur, M. Sighireanu, B. Vivien
INRIA Rhône-Alpes and DYADE / VASY group
655, avenue de l'Europe
F-38330 Montbonnot Saint Martin
France
hubert.garavel@inria.fr, vasy@inrialpes.fr

Abstract

This article gives an overview of the most recent features implemented in CADP *(*CÆSAR/ALDÉBARAN *Development Package), a toolbox dedicated to the design and verification of communication protocols and distributed systems. Besides the description of the new features, this paper also lists the latest applications of* CADP *to industrial case-studies and mentions the current research directions for improving* CADP.

1 Introduction

CADP (CÆSAR/ALDÉBARAN Development Package) is a software engineering toolbox for the design of communication protocols and distributed systems. It is based upon the Formal Description Technique LOTOS [18, 1], although it can also deal with systems described as networks of communicating finite-state machines. CADP offers a wide range of functionalities, including compilation, simulation, formal verification, and testing[1].

CADP is jointly developed by VASY (*Validation of Systems*), a common research group of INRIA Rhône-Alpes and DYADE (the joint-venture between BULL and INRIA for advanced research in information technology), and the VERIMAG laboratory.

This article is a follow-up to a previous paper [13] presented in June 1996 at the First COST 247 International Workshop on Applied Formal Methods in System Design, held in Maribor (Slovenia). Since this workshop, significant improvements have been brought to the CADP toolbox, especially in the framework of the European-Canadian project EUCALYPTUS-2 and in the framework of the BULL/INRIA collaboration. Two new versions of CADP have been released (version Z in December 1996 and version 97a in May 1997).

[1]On-line information about CADP is available from `http://www.inrialpes.fr/vasy/cadp.html`

This article is organized as follows. Section 2 presents the most significant new features in the latest versions of CADP. Section 3 gives a (non-exhaustive) list of industrial applications tackled using CADP. Finally, Section 4 gives some concluding remarks and draws the scientific directions of VASY for future work on CADP.

2 New features

The two latest versions (Z and 97a) of CADP introduce many improvements. For the user, the most visible changes concern the EUCALYPTUS graphical user-interface, which has been carefully improved, taking into account the double feedback received from engineers using this interface on a daily basis and from computer-science students using it occasionally during lab exercises. Other visible changes concern the XSIMULATOR tool for interactive simulation, which has been entirely rewritten in TCL/TK, and the MONITOR tool, a new TCL/TK-based tool that displays in real-time the number of states, transitions and the list of labels encountered during the generation of a Labelled Transition System (LTS).

However, there have been also deeper "semantical" changes, five of which are worth mentioning:

- (i) The functionality of the CÆSAR tool has been enriched. Initially, CÆSAR was a verification tool for generating the LTS corresponding to a LOTOS description [11, 14]. Compared to other existing model-checkers, a distinctive feature of CÆSAR is its ability to handle complex data types, including data structures of unbounded size (such as lists, sets, binary trees, etc.) provided that these types are restricted to a finite subset.

  In a second step, the purpose of CÆSAR has been extended beyond model-checking verification: the compiling algorithms implemented in CÆSAR have been reused in the framework of the OPEN/CÆSAR architecture and programming interfaces [12]. This enabled many new functionalities, such as interactive simulation, random execution, on-the-fly verification (deadlock detection, sequence search, μ-calculus formula evaluation, etc.), test case generation, etc.

Recently, a third step was made, by modifying CÆSAR in order to allow the generation of C code for embedded applications. This new functioning mode (called EXEC/CÆSAR) translates a LOTOS description (usually, a controller for some external device or system) into a C program which can be compiled and linked with external code. Each visible gate of the LOTOS description is mapped to a user-defined C function, and each rendezvous on a visible gate triggers a call to the associated C function, possibly with *in* and/or *out* parameters to express the values received and/or sent during the rendez-vous. Therefore, a formal description written in LOTOS can be used not only for verification purpose, but for generating code to be embedded into a prototype (or even a final product).

- (ii) The use of CÆSAR and EXEC/CÆSAR on several real-size applications pointed out that the generated C code was not as fast as hand-written C code. To address this problem, the generated C code was carefully analyzed using profiling tools and ten optimizations were implemented, leading to a considerable increase in speed: with the new version of CÆSAR, transition firing is about 8 times faster and, more generally, the time needed to generate a whole transition system has been divided in a factor ranging from 2 to 160.

- (iii) An improved version of the EVALUATOR tool for on-the-fly evaluation of μ-calculus formulas is available. The new version of EVALUATOR supports a richer formula language (label sets can now be specified using boolean *not* and *or* operations on labels), uses more efficient data structures to store the product states, and provides two different evaluation algorithms, a local one and a global one.

- (iv) The EXHIBITOR tool for the on-the-fly search of execution sequences defined by a pattern of visible actions has been entirely rewritten. The new version of EXHIBITOR accepts a more expressive pattern language, which combines boolean operators and (a subset of) regular expressions, and which allows to characterize deadlock states. The new version also implements two search algorithms: a depth-first search algorithm (which generalizes the algorithm used in the previous version of EXHIBITOR) and a breadth-first search algorithm, able to find the shortest sequence(s) matching a given pattern.

- (v) A new approach for compositional verification has been introduced in the CADP toolbox, as a mean to overcome the *state explosion* problem by using a *divide and conquer* strategy. Compositional verification was already supported in CADP by the ALDÉBARAN, EXP.OPEN, and OPEN/CÆSAR tools, which allow to split a given system into a set of communicating processes, which are individually generated and minimized according to bisimulation relations, then recombined incrementally using LOTOS parallel composition and hiding operators. This recombination can be used either for generating an LTS corresponding to the system (this LTS is equivalent, but smaller, to the LTS which would have otherwise been obtained without a compositional approach), or for verifying properties on-the-fly.

  Although this approach for compositional verification often gives satisfactory results on large examples (see [3] for an industrial application), it is not always sufficient, especially in the cases where some processes are too large for being generated separately from the other processes. This problem is addressed by a more general compositional approach [19] available with the latest version of CADP (namely the two new tools DES2AUT and PROJECTOR). Under this approach, each process can be generated in a constrained manner, by taking into account an *interface*, i.e., an LTS expressing (a superset of) the set of execution sequences permitted for this process by its environment. The process interfaces can be specified manually by the user or even synthesized automatically (in the former case, a warning is emitted if the interface proposed by the user is incorrectly restrictive with respect to the global behaviour of the system).

The above improvements have respectively been brought by: (i) H. Garavel; (ii) H. Garavel and M. Jorgensen; (iii) M. Bozga; (iv) X. Etchevers and H. Garavel; (v) J.-P. Krimm and L. Mounier.

3 Recent applications

The CADP/EUCALYPTUS tools have been used in many different application fields. We give here a (non-exhaustive) list of significant case-studies:

CO4: CO4 is a computer environment dedicated to the incremental and concurrent building of a distributed knowledge base [6]. A CO4 system is a hierarchy of *individual* (leaf) and *group* (node) bases, that communicate using a consensual decision protocol inspired from peer-reviewing policies.

The Co4 protocol was specified in Lotos, starting from a partially formal, but hand-made description. The resulting description (about 1,200 lines) takes advantage of the Apero concise data type notations [25]. The specification task alone pinpointed omissions, imprecisions and inconsistencies in the initial description.

The Cadp toolset was used for validation purpose. State space explosion limits the possibility of exhaustive exploration to simple scenarios: the most complex scenario covers one complete transaction through a hierarchy of five bases, producing about 43,000 states. To overcome this limitation, the Exhibitor tool was used to search on-the-fly for specific execution sequences, e.g., sequences leading to an inconsistent state of the knowledge base.

Besides another bunch of local corrections, the validation work put forth a (foreseen) case of knowledge consistency violation, an (unforeseen) case of inconsistent hierarchy construction, and four cases of unexpected message reception. All those results were reported to the designers of Co4 and are being integrated in a revised description of the protocol. This case-study was tackled by Ch. Pecheur.

DCL: The Departure Clearance (Dcl) protocol is an air-traffic control protocol defined by the European organization Eurocontrol [5]. This protocol aims at providing automated assistance for requesting and delivering departure information and clearances, with the objective of reducing air crew and controller workload as well as clearance delivery delays.

A Lotos description of the Dcl protocol was produced (about 300 lines) and analyzed using the Eucalyptus toolbox. To avoid state explosion, compositional verification was used: the whole system was divided into three processes; each process was translated into an Lts using the Cæsar and Cæsar.adt compilers; then the Aldébaran tool was used to minimize modulo strong bisimulation the three Ltss. Finally, the Exhibitor tool was used and highlighted the fact that "bad" execution sequences may happen using the full protocol, leading to differences in the departure information between the controller and pilot sides. As a consequence of this analysis, only a part of this protocol will be used, to avoid the possible problems detected by the use of its full functionality. This case-study was performed by Ch. Hernalsteen and T. Massart at the Free University of Brussels.

Equicrypt: Equicrypt is a Trusted Third Party (Ttp) protocol considered in the European Acts project number 051 ("Okapi") dealing with security, cryptography and authentication in computer networks. The Equicrypt protocol establishes authenticated connections between service providers (e.g., video on demand) and customers. In the Okapi project, Lotos was used to specify the Equicrypt protocol (about 1,000 lines) and verify its robustness to attacks by an intruder. At a certain abstraction level, security properties were expressed and verified automatically, on this protocol connected to a generic intruder process. The Eucalyptus toolbox (especially the Aldébaran and Exhibitor tools) was used to perform model-based verification: several unexpected, successful attacks against the Equicrypt protocol have been discovered. More precisely, all properties are fulfilled without the intruder, but some of them are falsified when the intruder is added. The diagnostic sequences can be used almost directly to exhibit the scenarios of two possible attacks on the protocol. This result suggest that model-checking can be appropriate for the verification of security protocols, especially because of its capability of finding the attacks as diagnostic sequences of unsatisfied properties. This case-study was performed at the Research Unit in Networking (Run) of the University of Liège. The subscription protocol was specified and verified by G. Leduc, O. Bonaventure, E. Koerner, L. Léonard, C. Pecheur, and D. Zanetti [20]. The registration protocol has been verified and improved by F. Germeau and G. Leduc [15].

IEEE-1394: The 1394 serial bus, also known as "FireWire", is a Ieee standard [17] for high performance data transmission (e.g., video). The asynchronous transmission mode of the Ieee-1394 link layer protocol was proposed by Jan-Friso Groote as a verification challenge for the 2nd COST 247 International Workshop on Applied Formal Methods in System Design held in Zagreb, Croatia.

Starting from a formal description [22] of the Ieee-1394 physical and link layers using the μCrl process algebra, a formal description in Extended-Lotos [27] of these two layers was produced, completed with an abstraction of the upper layer (transaction layer). This description (about 800 lines) was translated into "standard" Lotos (about 1,000 lines) using the Traian translator (see Section 4). Then, the Cæsar and

CÆSAR.ADT compilers were used to generate the corresponding LTSs for three different kinds of scenarios. Five correctness properties were expressed in the ACTL temporal logic [24] and verified on the LTSs for each set of scenarios using the XTL model-checker [7].

The verification revealed the existence of an "unspecified reception", i.e., a missing transition in one of the state machines given in the IEEE-1394 standard, a situation which can mislead implementors and cause unexpected functioning errors. This case-study was performed by M. Sighireanu and R. Mateescu [28].

Production Cell: Proposed by Claus Lewerentz and Thomas Lindner (FZI Karlsruhe, Germany), the "production cell" case-study [21] is a canonical example for the application of formal methods to industrial problems. A LOTOS description (about 1,000 lines) was elaborated for controlling the various devices and robots of an automated metal plant. Taking advantage of the new EXEC/CÆSAR functionality, this LOTOS description (completed with a small, hand-written interface driver) was used to animate and control the TCL/TK program provided by FZI for simulating the production cell plant. This case-study was performed by H. Garavel and M. Jorgensen.

4 Conclusion and perspectives

Bringing formal methods to a maturity state compatible with industrial needs is a difficult task, which requires both theoretical background and software development skills. The CADP/EUCALYPTUS toolbox targets at this goal: the recent versions bring major improvements and new features, the practical usefulness of which is demonstrated by several real-life applications.

The development of CADP will progress in several directions. The VASY research group will continue its work along the following research lines:

- Various experiments have shown that the LTSs generated by CÆSAR are often "too large", meaning that they are not minimal in terms of strong bisimulation. For instance, the number of states of the LTS generated for the DCL is reduced 100 times when minimizing this LTS modulo strong bisimulation. In order to go beyond the practical limitations of model-checking, one must generate directly "smaller" LTSs, i.e., LTSs already reduced (in part) with respect to strong bisimulation. The introduction in CÆSAR of data flow analysis techniques for this purpose is currently investigated.

- The connection of CADP tools with the FC2TOOLS [2], another verification toolbox developed at INRIA Sophia-Antipolis, is under way. By allowing different implementations of verification algorithms to be used, this interconnection will increase the confidence in the verification results and will provide a basis for scientific comparison of tools and algorithmic performances on a given problem.

- A LOTOS version of the TGV[2] tool [8, 9] is currently developed by the PAMPA project of IRISA/INRIA Rennes in the framework of DYADE, in order to automate test generation for BULL's multiprocessor architectures.

- The development of the XTL[3] model-checker is carried on by Radu Mateescu as part of his PhD thesis. XTL is a functional programming language allowing a compact description of various temporal logic operators to be evaluated over an LTS. The XTL language gives access to all the informations contained in the states and labels of an LTS and offers primitives allowing to explore the transition relation.

  The first version of XTL described in [13, 7] is now operational and has been used for teaching purposes and case-studies [23, 28].

  A second version of XTL is under development, which enriches the XTL language with an extended, value-based, modal μ-calculus built from modal and fixed point operators parameterized by data variables, from regular expressions characterizing LTS action sequences (similar to those used in the PDL logic [10]), and from data handling constructs, similar to those of functional programming languages ("if-then-else", "case", "let", etc.).

  While preserving all the functionalities of the first version (especially the possibility to define non-standard temporal operators), the second version allows a more concise expression of the properties involving the values contained in the LTS actions (under a form very close to standard mathematical notations) and a more efficient evaluation of temporal formulas by means of specialized algorithms dedicated to μ-calculus model-checking.

  Several temporal logics have already been implemented as libraries of XTL operators: CTL [4], ACTL [24], LTAC [26], as well as a branching-time fragment of the μCRL modal logic [16], which allows to handle data values.

[2] *Test Generation based on Verification*
[3] *eXecutable Temporal Language*

- The development of a prototype compiler and verification tool for an implementable subset of E-LOTOS has been undertaken by Mihaela Sighireanu, as part of her PhD thesis, and by Bruno Vivien. This compiler is named TRAIAN and its current version includes a front-end (for lexical, syntactic and static semantics analysis), a pretty-printer, and a translator from (a subset of) E-LOTOS to LOTOS, which allow to reuse the existing CADP tools for compiling and verifying E-LOTOS descriptions. The TRAIAN tool has already been applied for tackling the IEEE-1394 case-study [28].

  Future versions of TRAIAN will extend the subset of E-LOTOS accepted by the translator (e.g., with more complex E-LOTOS constructs such as exceptions). Also, the problem of direct E-LOTOS implementation (i.e., without translation to LOTOS) is under study, the objective being the development of a tool that would be for E-LOTOS what CÆSAR and CÆSAR.ADT are today for LOTOS.

Acknowledgements

The authors are grateful to Guy Leduc, Thierry Massart, and Laurent Mounier for their help and comments about this paper.

References

[1] Tommaso Bolognesi and Ed Brinksma. Introduction to the ISO Specification Language LOTOS. *Computer Networks and ISDN Systems*, 14(1):25–29, January 1988.

[2] Amar Bouali, Annie Ressouche, Valérie Roy, and Robert de Simone. The Fc2Tools set: a Toolset for the Verification of Concurrent Systems. In Rajeev Alur and Thomas A. Henzinger, editors, *Proceedings of the 8th Conference on Computer-Aided Verification (New Brunswick, New Jersey, USA)*, volume 1102 of *Lecture Notes in Computer Science*. Springer Verlag, August 1996.

[3] Ghassan Chehaibar, Hubert Garavel, Laurent Mounier, Nadia Tawbi, and Ferruccio Zulian. Specification and Verification of the PowerScale Bus Arbitration Protocol: An Industrial Experiment with LOTOS. In Reinhard Gotzhein and Jan Bredereke, editors, *Proceedings of the Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification FORTE/PSTV'96 (Kaiserslautern, Germany)*, pages 435–450. IFIP, Chapman & Hall, October 1996. Full version available as INRIA Research Report RR-2958.

[4] E. Clarke, E. A. Emerson, and A. P. Sistla. Automatic Verification of Finite-State Concurrent Systems using Temporal Logic. In *10th Annual Symposium on Principles of Programming Languages*. ACM, 1983.

[5] Eurocontrol. Transition Guidelines for Initial Air/Ground Data Communications Services. Technical report, Eurocontrol, EATCHIP Project, Brussels, October 1996.

[6] Jérôme Euzenat. Building Consensual Knowledge Bases: Context and Architecture. In *Proceedings of the 2nd International Conference on Building and Sharing Very Large-Scale Knowledge Bases (KBKS), Enschede the Netherlands*, pages 143–155, 1995.

[7] Jean-Claude Fernandez, Hubert Garavel, Alain Kerbrat, Radu Mateescu, Laurent Mounier, and Mihaela Sighireanu. CADP (CÆSAR/ALDEBARAN Development Package): A Protocol Validation and Verification Toolbox. In Rajeev Alur and Thomas A. Henzinger, editors, *Proceedings of the 8th Conference on Computer-Aided Verification (New Brunswick, New Jersey, USA)*, volume 1102 of *Lecture Notes in Computer Science*, pages 437–440. Springer Verlag, August 1996.

[8] Jean-Claude Fernandez, Claude Jard, Thierry Jéron, Laurence Nedelka, and César Viho. Using On-the-Fly Verification Techniques for the Generation of Test Suites. In R. Alur and T. A. Henzinger, editors, *Proceedings of the 8th International Conference on Computer-Aided Verification (Rutgers University, New Brunswick, NJ, USA)*, volume 1102 of *Lecture Notes in Computer Science*, pages 348–359. Springer Verlag, August 1996. Also available as INRIA Research Report RR-2987.

[9] Jean-Claude Fernandez, Claude Jard, Thierry Jéron, Laurence Nedelka, and César Viho. An Experiment in Automatic Generation of Test Suites for Protocols with Verification Technology. 1996. Special issue on Industrially Relevant Applications of Formal Analysis Techniques. Also available as INRIA Research Report RR-2923.

[10] M. J. Fischer and R. E. Ladner. Propositional Dynamic Logic of Regular Programs. *Journal of Computer and System Sciences*, (18):194–211, 1979.

[11] Hubert Garavel. Compilation of LOTOS Abstract Data Types. In Son T. Vuong, editor, *Proceedings of the 2nd International Conference on Formal Description Techniques FORTE'89 (Vancouver B.C., Canada)*, pages 147–162. North-Holland, December 1989.

[12] Hubert Garavel. The OPEN/CÆSAR Reference Manual. Rapport SPECTRE C33, Laboratoire de Génie Informatique — Institut IMAG, Grenoble, May 1992.

[13] Hubert Garavel. An Overview of the Eucalyptus Toolbox. In Z. Brezočnik and T. Kapus, editors, *Proceedings of the COST 247 International Workshop on Applied Formal Methods in System Design (Maribor, Slovenia)*, pages 76–88. University of Maribor, Slovenia, June 1996.

[14] Hubert Garavel and Joseph Sifakis. Compilation and Verification of LOTOS Specifications. In L. Logrippo, R. L. Probert, and H. Ural, editors, *Proceedings of the 10th International Symposium on Protocol Specification, Testing and Verification (Ottawa, Canada)*, pages 379–394. IFIP, North-Holland, June 1990.

[15] François Germeau and Guy Leduc. A Computer Aided Design of a Secure Registration Protocol. SART 97/06/14 of the ACTS 051 "OKAPI" project, University of Liège, 1997. Submitted for publication.

[16] J-F. Groote and S. M. F. van Vlijmen. A modal logic for μCRL. Technical Report 114, Logic Group Preprint Series, Department of Philosophy, Utrecht University, 1994.

[17] IEEE. Standard for a High Performance Serial Bus. IEEE Standard 1394-1995, Institution of Electrical and Electronic Engineers, 1995.

[18] ISO/IEC. LOTOS — A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. International Standard 8807, International Organization for Standardization — Information Processing Systems — Open Systems Interconnection, Genève, September 1988.

[19] Jean-Pierre Krimm and Laurent Mounier. Compositional State Space Generation from Lotos Programs. In Ed Brinksma, editor, *Proceedings of TACAS'97 (Tools and Algorithms for the Construction and Analysis of Systems)*, Enschede, The Netherlands, April 1997. Springer Verlag. Extended version with proofs available as Research Report VERIMAG RR97-01.

[20] G. Leduc, O. Bonaventure, E. Koerner, L. Léonard, C. Pecheur, and D. Zanetti. Specification and Verification of a TTP Protocol for the Conditional Access to Services. In *Proceedings of the 12th Jacques Cartier Workshop on "Formal Methods and their Applications: Telecommunications, VLSI and Real-Time Computerized Control System", Montréal, Canada*, October 1996.

[21] Claus Lewerentz and Thomas Lindner, editors. *Formal Development of Reactive Systems – Case Study Production Cell*, volume 891 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, January 1995.

[22] Bas Luttik. Description and Formal Specification of the Link Layer of P1394. In Ignac Lovrek, editor, *Proceedings of the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (Zagreb, Croatia)*, June 1997.

[23] R. Mateescu. Formal Description and Analysis of a Bounded Retransmission Protocol. In Z. Brezočnik and T. Kapus, editors, *Proceedings of the COST 247 International Workshop on Applied Formal Methods in System Design (Maribor, Slovenia)*, pages 98–113. University of Maribor, Slovenia, June 1996. Also available as INRIA Research Report RR-2965.

[24] R. De Nicola and F. W. Vaandrager. *Action versus State based Logics for Transition Systems.* In *Proceedings Ecole de Printemps on Semantics of Concurrency*, volume 469 of *Lecture Notes in Computer Science*, pages 407–419. Springer Verlag, 1990.

[25] Charles Pecheur. *Improving the Specification of Data Types in* LOTOS. Doctorate thesis, University of Liège, November 1996.

[26] Jean-Pierre Queille and Joseph Sifakis. Fairness and Related Properties in Transition Systems — A Temporal Logic to Deal with Fairness. *Acta Informatica*, 19:195–220, 1983.

[27] Juan Quemada, editor. Working Draft on Enhancements to LOTOS. ISO/IEC JTC1/SC21/WG7 Project 1.21.20.2.3, 1997 January.

[28] Mihaela Sighireanu and Radu Mateescu. Validation of the Link Layer Protocol of the IEEE-1394 Serial Bus ("FireWire"): an Experiment with E-LOTOS. In Ignac Lovrek, editor, *Proceedings of the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (Zagreb, Croatia)*, June 1997. Full version available as INRIA Research Report RR-3172.