Keith Hardy Jim Briggs (Eds.)

# Reliable Software Technologies – Ada-Europe '97

1997 Ada-Europe International Conference
on Reliable Software Technologies
London, UK, June 2-6, 1997
Proceedings

Springer

Volume Editors

Keith Hardy
Ultra Electronics Command & Control Systems Knaves Beech Business Centre
Loudwater, High Wycombe, Buckinghamshire HP10 9UT, U.K.
E-mail: khardy@ueccs.co.uk

Jim Briggs
University of Portsmouth, Department of Information Science
Locksway Road, Southsea, Honts. PO4 8JF, UK
E-mail: briggsjs@sis.port.ac.uk

Cataloging-in-Publication data applied for

**Die Deutsche Bibliothek - CIP-Einheitsaufnahme**

**Reliable software technologies** : proceedings / Ada Europe '97, 1997 Ada Europe International Conference on Reliable Software Technologies, London, UK, June 2 - 6, 1997. Keith Hardy ; Jim Briggs (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1997
(Lecture notes in computer science ; Vol. 1251)
ISBN 3-540-63114-3

CR Subject Classification (1991): D.2, D.1.2-5, D.3, D.4, C.2.4, C.3, K.6

ISSN 0302-9743
ISBN 3-540-63114-3 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author
SPIN 10549909 06/3142 – 5 4 3 2 1 0 Printed on acid-free paper

# Preface

Reliable software is a topic which now embraces not just safety-critical applications, but also those that are mission-critical or environment-critical. As the processing power of microprocessors continues to increase relative to price, the opportunities also increase to use software in situations where it was previously not financially viable. As a result, the impact and risk to the community through software technology rises at an alarming rate. Software developers have a duty therefore to ensure that every reasonable effort is made to provide reliable solutions for the community and surrounding environment.

The papers presented at Ada-Europe'97 offer different, but valuable, approaches to the development of reliable software. At the top end of the criticality scale, the need for specialist tools for safety-critical software is discussed in a number of papers, such as Thornley's paper on experiences with the use of SPARK and Dobbing's paper on the task-safe minimal Ada real-time toolset. These are supported by papers addressing the analysis of such systems, such as one on performance requirements by Pierce et al. For those interested in other approaches to verification there are a number of contributions discussing the special needs of testing reliable systems in a quantifiable manner, such as Wegener's paper on systematic unit testing, and Bell's on an analysis toolset - one of three papers from the Hughes Canadian Automatic Air Traffic Control System (CAATS) project.

It is a difficult time for the Ada community. Although the language definition has completed its transition to the new standard for Ada 95, industry compilers and toolsets for this standard are only just starting to become available. These new tools, although not as immature as the early Ada 83 tools, cannot be expected to be stable and many only contain a limited subset of the Ada 95 language features, with fully-featured compilers not expected until next year when the new validation suite is available. This has not stopped a number of projects exploring the new features of the language, particularly through the GNAT compiler, to gain early exposure and much needed experience with the new features of the language. These projects include the SPIF project described by Dupouy et al, the sort race construction set from Feldman, and using Ada 95 as a basis for the architecture of systems by Ögren. Additionally, work has already been carried out on some of the new features of the language that may present new problems, as demonstrated by the papers from Wellings, Burns & Pazy on task termination, Holzmüller & Plödereder on finite unions, and Gellerich and Plödereder on parameter-induced aliasing.

Ada 95 offers the opportunity to take advantage of more of the object-oriented concepts, however. English highlights some of the potential pitfalls of using inheritance. Two papers discuss other OO aspects, such as the integration of syntactic constructs and structural features for formalised object-oriented methods (Cheung, Chow & Cheung) and the use of Jackson System Design for high integrity Ada software by Yeung. Waterman discusses some of the techniques required for testing some of the new features of the language.

Another new feature of Ada 95 is the way it addresses performance and scheduling constraints. These aspects have been picked up by papers from Gonzalez Harbour, Gutierrez Garcia & Palencia Gutierrez on implementing application-level sporadic server schedulers and Romanovsky, Mitchell & Wellings on programming atomic actions.

Although there may not be many Ada 95 projects available to discuss at this stage, there are some interesting Ada 83 projects, none more so than the CCO MARS 96 project. Pichon describes how it integrated HOOD, Ada and XInAda technologies. Two more contributions, on code-data consistency (Célier) and developing scripting capabilities for simulators (Jovanovic, Sotirovski & Van Aswegen), refer to the CAATS project.

Although it has not been possible to mention all the papers, we have given a flavour of the content of these proceedings. These are exciting times for the reliable software community as we continue to encounter and rise to new challenges - not just from the arrival of Ada 95, but from a constant change in the technology available in terms of tools, methods and techniques, and in the increased complexity and scope of the real world problems we are required to solve. We hope you agree that this conference has gone some way to addressing these.

Jim Briggs & Keith Hardy
May 1997

# Table of Contents