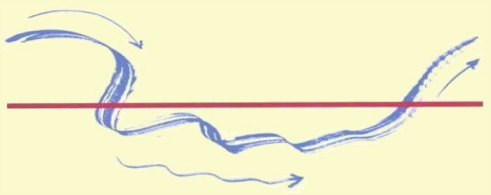


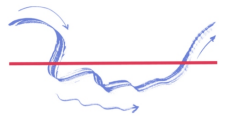
10th International Conference on
Reliable Software Technologies
Ada-Europe 2005

York, UK, June 20–24, 2005

ADVANCE PROGRAMME

http://www.ada-europe.org/conference2005.html

The University of York



Presentation

This year the 10^{th} International Conference on Reliable Software Technology takes place in York during the week of June 20^{th}–24^{th}. The conference has a number of highlights including three keynote presentations by John McDermid, Martyn Thomas and Bev Littlewood, a full three day technical programme of fully refereed papers (including one day on industry focused presentations), an industrial exhibition with more than ten exhibitors, two days of tutorials including a special half day presentation on the new Ada 2005 language by four of its designers: John Barnes, Alan Burns, Pascal Leroy and Tucker Taft.

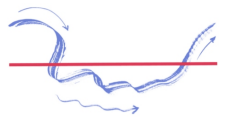
The conference offers an international forum for researchers, developers and users of reliable software technologies. Presentation and discussions cover applied and theoretical work currently conducted to support the development and maintenance of software systems.

York is a beautiful and historical (small) city in the north of the UK. It has a first class university with one of the best Computer Science departments in the world. The department has been involved with the development of programming languages for a number of years (indeed it ran the first series of technical meetings on Ada in the 1970s). It is pleased to host this meeting on reliable software technology.

York can be reached easily by train from London (approximately two hours) or Manchester airport. The conference is held at the Royal York Hotel which is adjacent to the York train station a few minutes from the centre of York and the Minster (Cathedral).

Overview of the Week

<table>
<tr><th></th><th>Morning</th><th>Late Morning</th><th>After Lunch</th><th>Afternoon</th></tr>
<tr><td rowspan="3">Monday
June 20th
Tutorials</td><td colspan="2">J.-P. Rosen
Developing Web-aware Applications in Ada with AWS</td><td colspan="2">J. L. Tokar & B. Lewis
SAE Architecture Analysis and Design Language</td></tr>
<tr><td colspan="2">P. Amey & N. White
Correctness by Construction—A Manifesto for High Integrity Engineering</td><td colspan="2">B. Dobbing
High-Integrity Ravenscar Using SPARK</td></tr>
<tr><td colspan="4">B. M. Brosgol
Real-Time Java for Ada Programmers</td></tr>
<tr><td rowspan="2">Tuesday
June 21st
Sessions &
Exhibition</td><td rowspan="2">Invited Talk:
John McDermid
Model-based Development of Safety-critical Software: Opportunities and Challenges</td><td>Applications</td><td>Formal Methods</td><td>Ada and Education,
Distributed Systems</td></tr>
<tr><td>Design and Scheduling Issues</td><td colspan="2">Vendor Session</td></tr>
<tr><td>Wednesday
June 22nd
Sessions &
Exhibition</td><td>Invited Talk:
Martyn Thomas
Extreme Hubris</td><td>Industrial
Presentations</td><td>Industrial
Presentations</td><td>Industrial
Presentations</td></tr>
<tr><td>Thursday
June 23rd
Sessions &
Exhibition</td><td>Invited Talk:
Bev Littlewood
Assessing the Dependability of Software-based Systems: A Question of Confidence</td><td>Certification and
Verification</td><td>Language Issues</td><td>Ravenscar Technology</td></tr>
<tr><td rowspan="3">Friday
June 24th
Tutorials</td><td colspan="2">J. Barnes, A. Burns, P. Leroy, S. T. Taft
Ada 2005</td><td colspan="2">M. Heaney
Programming with the Ada 2005 Standard Container Library</td></tr>
<tr><td colspan="2">W. Bail
Requirements Engineering for Dependable Systems</td><td colspan="2">P. Rogers
Software Fault Tolerance</td></tr>
<tr><td colspan="4">J. McDermid, R. Weaver
Software Safety Cases</td></tr>
</table>



Invited Speakers

Model-based Development of Safety-critical Software: Opportunities and Challenges

John McDermid,
University of York, UK

Tuesday June 21st, 9:00

There is a growing use of model-based software development, including the automatic generation of code from the models, in mainstream software engineering. There is a perception that the use of such technology could reduce the cost of safety critical software development. The use of model-based approaches offers some benefits, including better validation of designs, and shorter change cycles. However, it also has some problems, including the trustworthiness of code generation methods and tools, and the comparatively low level of abstraction at which the tools operate. The talk will discuss the opportunities and challenges of model-based development, including the role of program-level verification, aiming to identify the likely future development of such technology.

Presenter

John McDermid has been Professor of Software Engineering at the University of York since 1987. He runs the High Integrity Systems Engineering research group which is probably the world's largest academic group focusing on systems and software safety. He has undertaken research on a range of topics including hazard and safety analysis, safety cases, requirements engineering and formal methods, and has been successful in transferring research results into industry, especially the aerospace and defence industries. He has been influential in the development of standards for safety critical systems and software, including guiding the development of Issue 3 of Defence Standard 00-56.

Extreme Hubris

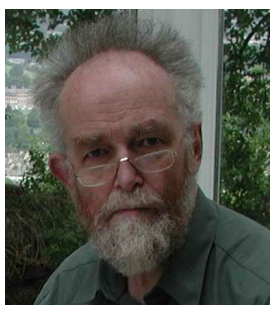
Martyn Thomas,
Thomas Associates, UK

Wednesday June 22nd, 9:00

The software industry frequently behaves like a fashion business, with crazes taking hold on the basis of novelty, an attractive name, and a few memorable slogans. The current infatuation with "extreme programming" is one such craze that combines several valuable but largely unoriginal approaches to software development with several that are original but unhelpful or actively dangerous.

This talk examines the principles of XP, distinguishing the sane from the absurd, and proposes an alternative Manifesto for Dependable Software Development.

Presenter



Martyn Thomas is an independent consultant software engineer. He specialises in the assessment of large, real-time, safety-critical, software intensive systems, software engineering, and engineering management. He is a member of the Expert Witness Institute and acts as an expert witness where complex software engineering issues are involved.

He is Visiting Professor in Software Engineering at the University of Oxford, and a Visiting Professor at the University of Bristol and the University of Wales, Aberystwyth.

In 1983, he founded Praxis, to exploit modern software development methods. Martyn Thomas serves on the IT policy-making bodies of both UK professional computing institutions, the British Computer Society (BCS) and the Institution of Electrical Engineers (IEE). He chairs the steering committee for the UK's major research collaboration in dependable systems, DIRC, and he is the only European member of a US National Academy of Sciences study into Certifiably Dependable Software.

Assessing the Dependability of Software-based Systems: A Question of Confidence

Bev Littlewood,
City University, London, UK

Thursday June 23rd, 9:00

Issues of uncertainty lie at the heart of dependability assessment. Most obviously, we cannot predict with certainty when a system will fail—that's why we use probabilistic measures of dependability (mean time to failure, failure rate, probability of surviving mission time, etc.). Less obviously—and often ignored—is the uncertainty associated with assessments of uncertainty. Claims such as 'this system has a probability on demand better than 10^{-3}' can never be made with certainty. Uncertainty about the truth of such a claim might come from weakness (or paucity) of supporting evidence for the claim; it might arise because we are uncertain about the truth of some underlying assumptions. In the talk I shall suggest that we need to associate with dependability claims an acknowledgement (and assessment) of this uncertainty about the truth of a claim. Essentially, we need to say how confident we are in the truth of the claim. I shall argue that probability is sometimes an appropriate formalism to represent confidence. I shall speculate about similarities between confidence in arguments and dependability of systems. For example, I shall claim that just as we can use design diversity to increase the

dependability of a system, so we can use argument diversity (multi-legged arguments) to increase confidence in claims. I shall suggest that, in addition to ALARP (As Low As Reasonably Practicable), which concerns the level of dependability of a system, we need ACARP (As Confident As Reasonably Practicable). Needless to say, some of this stuff might be a bit controversial...

Presenter

Bev has worked for many years on problems associated with the modelling and evaluation of dependability of software-based systems, and has published many papers in international journals and conference proceedings and has edited several books. He is a member of IFIP Working Group 10.4 on Reliable Computing and Fault Tolerance, of the BCS Safety-Critical Systems Task Force, and of the UK Computing Research Committee. He is

currently serving his second term as Associate Editor of the IEEE Transactions on Software Engineering, and is on the editorial boards of several other international journals. He is a member of the UK Nuclear Safety Advisory Committee, and of its Research Committee; he is a Fellow of the Royal Statistical Society.
Bev Littlewood has degrees in mathematics and statistics, and a PhD in Statistics and Computer Science. He founded the Centre for Software Reliability at City University, London, in 1983 and was its Director from then until 2003. He is currently Professor of Software Engineering at City University.

Special Tutorial: Ada 2005

This tutorial is at a reduced price for those attending the main conference (and includes lunch).

Ada 2005

John Barnes, Alan Burns, Pascal Leroy, S. Tucker Taft

Friday June 24th, morning

Another ten years has passed, and it is time to update the Ada standard. This tutorial focuses on the new version of the language that is currently being finalised.

Ada people will recall that the development of Ada 95 from Ada 83 was an extensive process funded by the USDoD. Formal requirements were established after comprehensive surveys of user needs and competitive proposals were then submitted resulting in the selection of Intermetrics as the developer under the leadership of Tucker Taft. The whole technical development process was then comprehensively monitored by a distinct body of Distinguished Reviewers. Of course, the process was also monitored by the ISO committee concerned and the new language finally became an ISO standard in 1995.

The development of Ada 2005/6 from Ada 95 has been (and continues to be) on a more modest scale. The work has almost entirely been by voluntary effort with support from within the industry itself through bodies such as the Ada Resource Association and Ada-Europe. The Ada Rapporteur Group (the ARG) is drafting the revised standard. The editor, who at the end of the day actually writes the words of the standard, is the indefatigable Randy Brukardt. The current aim is to have the amendment to the language (the new version of Ada is strictly speaking an amendment of the Ada 95 standard) completed by mid 2005, with formal acceptance of the document defining the changes occurring towards the end of 2005 or early 2006 (so it might be Ada 2006).

This tutorial will give an overview of the Ada 2005/6 language. The presenters are four of the key members of the ARG—*John Barnes*, *Alan Burns*, *Pascal Leroy* (chair of ARG) and *Tucker Taft*.

The scope of the changes can be summarised as follows:

1. Improvements to the OO model, including the use of Java-like Object.Op notation
2. Introduction of Java-like interfaces (**interface** is a new reserved word)
3. More flexible access types
4. Enhanced structure and visibility control, including a new limited with clauses which permit types in two packages to refer to each other.
5. Tasking and real-time improvements, including:
   - new dispatching policies (non-preemptive, round robin and earliest deadline first—EDF) that work individually or together to support hierarchical dispatching,
   - various forms of budget control (for individual tasks and for groups of tasks),
   - dynamic ceilings for protected objects,
   - timing event that are executed when external time reaches a specified value,
   - handlers that are executed when tasks terminate, and
   - protected, task and synchronized interfaces.
6. Improvements to exceptions, generics etc.
7. Extensions to the standard library packages, includes a comprehensive Container library (covered in another tutorial)

Together these changes represent a significant enhancement to Ada. Now is the time to improve your knowledge of Ada 2005.

Tutorials

Developing Web-aware Applications in Ada with AWS

Jean-Pierre Rosen, Adalog, France

Monday June 20th, morning

This tutorial describes AWS, the Ada Web Server, and how to use it for the development of web-aware applications. It describes the principles of AWS, from the most basic functionality to more advanced functions (authentication, SOAP interface, session management, hotplugs, multi-server applications, etc.). The seminar emphasises practical usage of AWS, and presents design patterns that have proved effective for developing existing applications. It compares the development process with AWS to other techniques.

The tutorial provides attendees with the information needed to assess whether AWS is appropriate to their needs, and the necessary knowledge to start writing full-scale Web applications.

Presenter

J.-P. Rosen graduated from ENST in 1975, and obtained his PhD in 1986.

He started as a software engineer at the computing center of ENST. After a Sabbatical at New York University on the Ada/ED Project, he worked as Professor at ENST, where he was responsible for the teaching of Software Engineering and Ada.

He has now formed Adalog, a company specialized in high level training, consultancy, and software development in the fields of Ada and OOD.

J.-P. Rosen is Chairman of the AFNOR (French standardization body) group for Ada, and the author of "Méthodes de Génie Logiciel avec Ada 95" (Software Engineering Methods with Ada 95) and "HOOD: an industrial approach for software development".

Why should you participate in this tutorial?

AWS is more than a simple Web server, it allows incorporation of Web technology into applications where the Web interface is only part of the problem. By attending this tutorial, participants will gain in-depth understanding of the issues of Web interfaces, and will discover new solutions to common problems, like using a browser as a GUI or providing control through Web interfaces to real-time programs.

Correctness by Construction—A Manifesto for High Integrity Engineering

Peter Amey & Neil White, Praxis High Integrity Systems, UK

Monday June 20th, morning

Two of the major recurring issues in systems and software engineering are unacceptably high defect rates, and lack of ability to make changes over a significant period of time in a reliable manner. These issues frequently cause major impact on the usability and maintainability of systems, and this impact becomes intolerable when components of the system become unmaintainable, or when high levels of security accreditation or safety certification must be achieved.

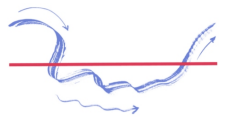
This tutorial presents the Correctness by Construction high integrity approach to systems and software engineering. Correctness by Construction is based on a set of principles, distilled from practical project experience, to realize systems and software engineering outputs with very low defect rate and very high resilience to change. These principles can be applied most effectively to new developments and upgrades. However some of the same principles can also be applied retrospectively to improve the maintainability and upgradability of existing systems.

Topics covered in the tutorial include:

1. The challenges that we face
2. How Correctness by Construction addresses the challenges
   (a) Validating the concepts of operation
   (b) Risk-directed requirements engineering
   (c) Formal specification
   (d) Rigorous design and implementation based on static analysis
   (e) Making the case for safety and security
   (f) Maintaining correctness through long-term upgrades
3. Correctness by Construction in action—project examples

Presenter

Neil White is an experienced software engineer and a specialist in high-integrity systems. He has designed, built, validated, and certified systems in the aerospace, naval, telecommunications and financial domains. The software he has worked on is currently serving on ships, submarines, a variety of aircraft, and in over a million television set-top boxes in homes around the country. Recently he has been developing a rigorous framework for the disciplined use of UML to document designs.



Peter Amey is an aeronautical engineer by original professional training. He served as an engineering officer in the Royal Air Force and spent several years at the Boscombe Down test establishment working on the certification of aircraft armament systems. Peter joined Program Validation Limited in 1992 to develop SPARK and the SPARK Examiner and continues that work today with Praxis High Integrity Systems. As well as developing SPARK he has used it on major programmes including Tornado, Eurofighter and the Lockheed C130J. Peter, now Chief Technical Officer at Praxis, teaches SPARK and Ada on a regular basis and has lectured widely on the development of critical systems.

Why should you participate in this tutorial?

The challenge of producing software that has an acceptably-low defect rate is hard enough. To do so under time and budget pressures is harder still. Yet the pressures grow ever stronger and the trust we place in software-intensive systems continues to rise. There is a pressing need to find ways of producing high-integrity software at an acceptable cost.

Over the past 16 years, Praxis has evolved a practical approach to software development which delivers exceptionally low defect rates with very high productivity. The approach eschews the use of fashionable technologies and silver bullets preferring an engineering approach which favours error prevention and relentless error elimination at every stage of development. The result is a development process that generates a continuous forward flow from requirements to implementation.

The tutorial will describe how this approach, which has come to be called Correctness by Construction works and how it has been applied to significant, real-world projects.

Real-Time Java for Ada Programmers

Benjamin M. Brosgol, AdaCore, USA

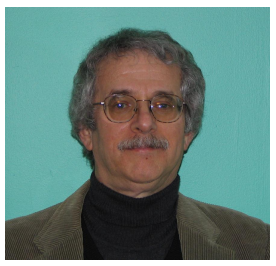
Monday June 20th, full day

Although the term "real-time Java" may sound self-contradictory, serious technical activity has been underway since early 1999 on extending the Java platform to satisfy the requirements for real-time systems, and several implementations exist. This work is relevant to the Ada community as both a challenge and an opportunity. On the one hand, it may compete with Ada in the real-time marketplace, but on the other hand some of its ideas may be worthy of consideration in a future version of the Ada language.

This tutorial will focus on the Real-Time Specification for Java ("RTSJ"), which was developed by the Real-Time for Java Expert Group under the auspices of Sun Microsystems' Java Community Process. The tutorial will analyze/critique the Java platform with respect to real-time support, summarize/illustrate the main elements of the RTSJ, and compare/contrast the design with Ada's real-time features (both in Ada 95 and under consideration for Ada 05). The tutorial will also outline the main aspects of the J-Consortium's "Core Extensions" (a competing real-time Java approach), will describe a proposed high-integrity profile for the RTSJ, and will provide a status update on the real-time Java work and its usage and prospects.

- Introduction
  1. Requirements for real-time programming
  2. Background and goals of real-time Java activities
  3. Summary of Java thread model
  4. Critique of Java platform for real-time support
- Pervasive technical issues
  1. Priority inversion management
  2. Garbage collection
  3. Object Oriented Programming and real-time systems
- The Real-Time Specification for Java
  1. Summary
  2. Concurrency, scheduling and synchronization
  3. Memory management
  4. Asynchrony
  5. Other features
  6. Comparison with "Core Extensions" from the J-Consortium
  7. Comparison with Ada
  8. High-integrity profile
- Conclusions
  1. Status of the definition and implementation of Real-Time Java
  2. Assessment of Real-Time Java
  3. What Ada can learn from Real-Time Java.

Presenter



Dr. Brosgol has over 25 years of experience in the computer software industry, with a focus on programming languages, software development methods, and real-time systems. He was a primary member of the Real-Time for Java Expert Group and a co-author of the Real-Time Specification for Java. He is currently a member of the Technical Interpretations Committee for the RTSJ, and he has also served as a reviewer of the J-Consortium's Core Extensions. He has delivered Java tutorials and courses since 1997, and the proposed tutorial will be an up-to-date version of the "Real-Time Java for Ada Programmers" tutorial that he delivered at Ada Europe 2004 and SIGAda 2004.

Dr. Brosgol is an internationally-recognized expert on Ada. He participated in both the initial language design and the Ada 95 revision, and he is a past chairman of the ACM's Special Interest Group on Ada (SIGAda). He has published numerous papers on Ada, has delivered presentations and tutorials at many Ada Europe and SIGAda conferences, and has been conducting courses on real-time programming in Ada since the late 1980s. He was an invited keynote speaker at the 2003 SIGAda

conference, where his topic was "Ada and Real-Time Java: Cooperation, Competition, or Cohabitation?" He is a senior member of AdaCore's technical staff in the US, in the Boston area.

Why should you participate in this tutorial?

- You will learn the pros and cons of the Java thread model, both in general and for real-time applications.
- You will see how real-time Java addresses the apparent "show stopper" problem of garbage collection.
- You will be able to judge whether real-time requirements can be met by a "pure" Object-Oriented Language.
- You will understand the effect of a dynamic and flexible scheduling approach, in terms of expressibility, predictability, and performance.
- You will discover who is using real-time Java, and for what sorts of. applications.

SAE Architecture Analysis and Design Language

Joyce L. Tokar, Pyrrhus Software, USA, Bruce Lewis, US Army, USA

Monday June 20th, afternoon

The Architecture Analysis and Design Language (AADL) is an architecture description language (ADL) that has been developed under the auspices of the International Society of Automotive Engineers (SAE), Avionics Systems Division (ASD) Embedded Computing Systems Committee (AS-2). The AADL was approved as an SAE standard in November of 2004.

The language has been defined to provide a consistent and concise notation, both textual and graphical, to be used to develop models of complex, real-time, critical systems such as those used in automotive, avionics, medical, robotic, and space-based systems. The AADL provides the notation to perform various types of analysis of the complex critical systems.

In the early stages of design, the AADL enables the definition of the preliminary connectivity between application and execution platform components. As an AADL model is developed, additional components and properties are specified. The properties provide the information needed by analysis tools to determine the behaviour and performance of the system being modelled. The AADL has been designed to facilitate the development of tools that provide automatic code generation of the system both in terms of the application software components and the underlying execution environment. The AADL may be used to verify an actual system against the specified model.

With automatic code generation, the AADL offers a system model that maintains significant information about a system that is useful throughout the lifetime of the system. Thus, the AADL offers support for all stages of system development.

This tutorial will provide an introduction to the AADL language from a textual and graphical perspective. It will also demonstrate the relationship between existing systems and AADL models. The tutorial will present several uses of the AADL in the design and analysis of safety-critical real-time systems. The tutorial will also demonstrate the relationship between AADL models and UML models.

Presenter

Joyce Tokar has invested more than seventeen years of research and development in the improvement of embedded systems technology, high level computing languages, and software and systems architectures. A recognized leader in the embedded systems community, Dr. Tokar has received numerous awards for her contributions including the Outstanding Ada Community Contribution Award 2000. Dr. Tokar has co-authored numerous papers and reports including the Society of Automotive Engineering Architecture Analysis and Description Language standard. Dr. Tokar received her PhD in Computer Engineering from Clemson University. She holds an MS and a BS in Computer Science from the University of Pittsburgh.

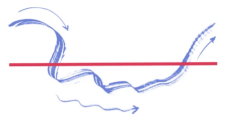
Bruce Lewis is a senior experimental developer for the US Army's Aviation and Missile Command, Research, Development and Engineering Laboratory, Software Engineering Directorate (SED). His work has focused on software architecture, reuse, and system evolution since 1991. He has served as the government lead on various DARPA projects related to ADLs and real-time systems, including those developing MetaH. He is the chairman of the AADL standardization committee.

Why should you participate in this tutorial?

The AADL is designed to support the declaration and analysis of systems comprised of software and hardware modules. As such, an understanding of AADL how software and execution platform components are represented and how they interact in an AADL model is beneficial to a variety of users.

This tutorial is useful to software and systems managers responsible for the development and integration of complex critical systems. This tutorial will provide a tool vendor with the information necessary to develop tools to support the development and analysis of source code and AADL models. Programmers will learn about the relationship between source code and the corresponding AADL specifications. This tutorial will be useful to system integrators who are responsible for developing a definition of the form of components that are acceptable for integration.

This tutorial is suitable for senior software and systems engineers as an introductory course; the tutorial does not presume prior knowledge of AADL. The attendees should have an understanding of the fundamentals of the development of complex, critical real-time sys-



tems.

Attendees may learn more about AADL at www.aadl.info

High-Integrity Ravenscar Using SPARK

Brian Dobbing, Praxis High Integrity Systems, UK

Monday June 20th, afternoon

SPARK is a well-established, unambiguous and fully-analysable annotated subset of Ada. In its original form SPARK excluded all forms of concurrency because weaknesses in the Ada tasking model made it incompatible with the design goals of SPARK. The advent of the Ravenscar Profile has provided an opportunity to extend SPARK to include concurrency and to enable the SPARK Examiner to analyse concurrent programs.

The tutorial will describe the way SPARK has been extended to include the Ravenscar Profile and how static analysis techniques can eliminate all of the erroneous behaviour, bounded errors and implementation-defined behaviour that remain in the concurrency model defined by the Profile.

The outcome of the tutorial will be an appreciation of how the RavenSPARK language and supporting toolset can be used to develop concurrent programs that are suitable for deployment at the highest safety and security integrity levels.

Topics will include:

1. An introduction to the Ravencar Profile
2. An introduction to SPARK
3. Errors and problems inherent in the Profile
4. RavenSPARK in practice
   (a) Program building blocks
   (b) Sample program—how errors are detected statically
   (c) Partition-wide flow analysis

Presenter

Brian Dobbing has presented papers and tutorials at major international conferences for the past twenty years. He has been actively involved in all aspects of Ada since 1979, including language standardization, Ada product development environments, and formal safety certification of Ada runtime systems. Brian is one of the main designers of the Ravenscar Profile Ada tasking subset for high integrity systems, and has played a major role in its inclusion in the forthcoming Ada 2005 standard. Within Praxis High Integrity Systems, Brian is a leading member of the Correctness by Construction team, which includes the SPARK language and product capabilities. Brian was a key designer of the addition of support for the Ravenscar profile to the SPARK language and toolset, which is the subject of this tutorial.

Why should you participate in this tutorial?

All software engineers should aspire to the goal of generating programs that have very low numbers of defects and very high resilience to change. This goal should apply not only to high integrity software, but to all professionally-produced programs. As programs become more complex, the use of concurrency and interrupts has caused traditional verification techniques based on testing alone to become insufficient. One of the key components in establishing the absolute correctness of software is static analysis of the design and the code, since this is based on formality rather than on test sampling. If you are involved in complex software engineering, you need to appreciate what the RavenSPARK language can offer in terms of establishing the correctness of programs without resorting to hugely elaborate and exhaustive testing.

Software Safety Cases

John McDermid, Rob Weaver, University of York, UK

Friday June 24th, full day

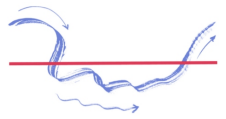
Current best practice in the development of safety critical software requires the construction of a software safety case to demonstrate the acceptability of software in its system context. This is a process that should go hand in hand with software development to ensure that appropriate software safety evidence is obtained. This tutorial explains the role safety cases play in the development and certification of safety critical software. Specifically, the tutorial will consider the construction and presentation of software safety cases.

The tutorial will start with the basics of safety cases and how they can be used to show that safety requirements are satisfied by a set of safety evidence. The role of the safety argument within the safety case will be explained, including how its development can be used to identify suitable items of safety evidence to support the safety case. Building upon this we will examine software safety cases, in particular the principles and standards that affect their development. We shall consider construction and presentation of software safety cases from the perspective of the principles of software safety and the requirements laid down in standards such as Issue 3 of DS 00-56, MoD's main safety standard. The role of a software safety case in the software development lifecycles will be discussed.

The tutorial will combine presentations with exercises to familiarise the attendees with the concepts introduced. Examples will be used throughout to demonstrate how software safety cases can be practically developed. The exercises will give an opportunity for hands on experience of constructing software safety arguments. This will help expand attendee's skills at identifying types of software safety evidence and their understanding of how evidence is combined to show the acceptability of software with respect to safety.

Presenter

John McDermid—see Biography in Invited Speaker Section



Dr. Rob Weaver is a Teaching Fellow in the Department of Computer Science and the University of York. Rob has been a full-time academic since 1999, the focus of his research being Safety. In particular Rob's research concerns the effect of software on system safety and the construction and presentation of safety cases. He lectures on the advanced MSc in Safety Critical Systems Engineering as well as presenting on industrial courses in System Safety Engineering and Management.

Why should you participate in this tutorial?

This tutorial will give you an opportunity to learn about current best practice in software safety case development. The tutorial will explain how, as part of this process, software safety evidence is first identified and then combined to show safety. This process goes hand in hand with software design and implementation, and thus forms an integral part of the safety critical software development lifecycle.

The tutorial will give you a view of the changing emphasis in system safety cases. Previous practice relied on a process-based approach to development of safety critical software, but such standards inevitably lag behind technology, and can be seen as restrictive. Emerging best practice uses an evidence-based approach to demonstrating software safety as part of a system safety case.

Requirements Engineering for Dependable Systems

William Bail, The MITRE Corporation, USA

Friday June 24th, morning

The development of large, complex software-intensive systems has proven to be a significant and persistent challenge, despite continuing optimism about the chances for success. The experience of government and industry has been less than encouraging, yet the need for new complex systems is ever-increasing. We have had some dramatic failures, yet we still have needs for new and improved systems. When we look at systems that have a need to be highly dependable or even safety-critical, we recognize the urgency in finding solutions—the current situation is clearly untenable. Unfortunately, as we mature our understanding of software development and improve our processes, it seems that the complexity of the systems we want to develop grows almost exponentially. In post-mortem analyses of project failures and "near-failures", one common root cause that has been identified has been the (mis)treatment of requirements. This result is not surprising since requirements form the foundation of all development. A fragile foundation does not provide a good basis for a sound, dependable system.

This tutorial will examine in detail the nature and role of requirements. It will discuss the various types of requirements and their role in development, as well as their impact to system success. The different ways that requirements need to be handled will be analyzed, and recommended techniques for process improvements will be discussed. An overview of traditional approaches will be provided, with an assessment of their strengths and weaknesses. In addition, a set of common challenges will be presented, together with strategies for how to manage them. Sample challenges include the impact of changing requirements and uncertain operational environments. While the tutorial will not specifically address particular approaches to requirements definition (such as tools and techniques), it will characterize classes of these techniques, and provide recommendations for how to select the appropriate ones for projects of interest.

The overall goal of the tutorial is to enable the attendees to effectively identify and handle requirements in their systems, as well as to implement improvements to their development processes to avoid common pitfalls that have historically plagued projects. The tutorial will also assist attendees in identifying risk areas within their projects, and in planning for effective mitigation activities.

Presenter

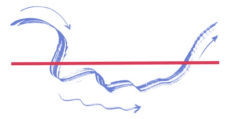
Since 1990, Dr. Bail has worked for The MITRE Corporation in McLean VA as a Computer Scientist in the Software Engineering Center (SWEC). MITRE is a not-for-profit corporation chartered to provide systems engineering services to the U.S. Government agencies, primarily the DoD, the FAA, and the IRS. Within MITRE, the SWEC focuses on supporting various programs with consultation, particularly transitioning emerging technologies into practice.

Dr. Bail's technical areas of focus include dependable software design and assessment, error handling policies, techniques for software specification development, design methodologies, metric definition and application, and verification and validation. At MITRE, Dr. Bail is currently supporting the U.S. Navy, focusing on the practice of software engineering within PEO IWS (Integrated Warfare Systems), particularly as applied to large real-time systems. Prior to 1990, Dr. Bail worked at Intermetrics Inc. in Bethesda MD.

Previously, Dr. Bail taught part-time at The University of Maryland from 1983-1986 in the Computer Science Department for undergraduate courses in discrete mathematics, computer architecture, and programming language theory. Since 1989 he has served as a part-time Adjunct Associate Professor at the University of Maryland University College where he develops instructional materials and teaches courses in software engineering, in topics such as Software Requirements, Verification and Validation, Software Design, Software Engineering, Fault Tolerant Software, and others.

Why should you participate in this tutorial?

If you are responsible for the development of a critical software intensive system, this tutorial will help you plan for and implement effective requirements processes,



helping you to manage your requirements from inception through deployment, and assist in avoiding many of the common pitfalls that many projects have encountered.

Software Fault Tolerance

Patrick Rogers, AdaCore, USA

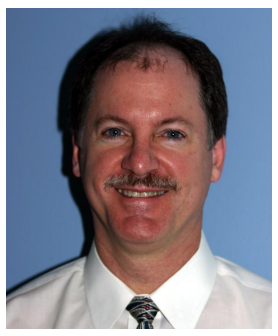
Friday June 24th, afternoon

Software for current safety-critical applications, e.g., flight control systems, is both large and complex, such that full testing is not feasible. Furthermore, complete proofs of correctness are at best inherently limited by the potential for specification errors. The combination of potential specification errors and overall complexity define a problem of handling unanticipated software faults. Software fault tolerance is the use of software mechanisms to deal with these unanticipated software faults.

This half-day tutorial explores the software-based techniques and mechanisms available for tolerating unanticipated software design faults in safety-critical systems. We examine the rationale for tolerating software faults, the similarities to mechanisms for tolerating hardware faults, and the advantages and disadvantages of the common techniques. Special attention is paid to the concept of design diversity as the underlying theory for the most widely-used mechanisms (e.g., N-Version Programming) and, in particular, whether design diversity can achieve the extremely low failure rates required for safety-critical systems. The mechanisms explored are illustrated with concrete implementations using Ada 95. This leads to a discussion of the issues of concurrency and exceptions in safety-critical applications and the appropriate application of language features.

Participants will have an appreciation of the necessity for tolerating software faults as well as a firm foundation for further study and informed application of available mechanisms. A bibliography of suggested reading is provided to that end.

Presenter



Patrick Rogers is a senior Member of the Technical Staff with Ada Core Technologies, specializing in high-integrity application support. A computing professional since 1975 and an Ada developer since 1980, he has extensive experience in real-time applications in Ada and C++ in both embedded and Linux/Unix/POSIX-based environments. An experienced lecturer and trainer since 1981, he has provided numerous tutorials and courses in software fault tolerance, hard real-time schedulability analysis, object-oriented programming, and the Ada programming language. He holds B.S. and M.S. degrees in Computer Science from the University of Houston and a Ph.D. in Computer Science from the University of York in the Real-Time Systems Research Group on the topic of software fault tolerance.

Why should you participate in this tutorial?

Systems software architects and developers responsible for safety-critical software will gain an appreciation for software fault tolerance facilities, including their limitations, and will have a firm foundation for informed application of available mechanisms as well as further study.

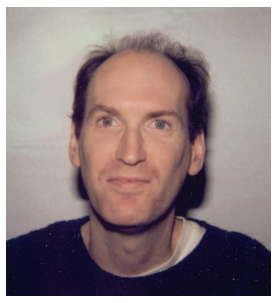
Programming with the Ada 2005 Standard Container Library

Matthew Heaney, On2 Technologies, USA

Friday June 24th, afternoon

This tutorial provides an overview of the standard container library, describing its design and philosophy and presenting techniques for using the library most effectively. Containers are divided into two main categories: sequence containers, to insert elements at specified positions, and associative containers, which insert elements in order by key. The library includes vectors and lists (from the former category), and hashed and sorted sets and maps (from the latter). All containers have variants to support elements (or keys) that have an indefinite subtype. Containers have various mechanisms (including both active and passive iterators) for designating and accessing container elements.

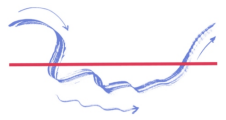
Presenter



Matt is the author of Charles, a container library for Ada closely modeled on the STL. Charles was the basis of the proposal selected by the ARG for the Ada standard container library (described in AI-302). He has given many Ada tutorials on topics that include object-oriented programming, design patterns, and software systems and library design.

Why should you participate in this tutorial?

The standard container library is an important addition to the Ada language, since developers need data structures with semantics more sophisticated than simple arrays or linked lists. For example, one often needs to map an element to some other type (typically String), but an array is not general enough since it only provides support for mapping to a discrete index subtype. The developer also needs abstractions that scale well to large numbers of elements, with specified time behavior. The standard container library solves these kinds of problems, and thus greatly simplifies many programming tasks that would be tedious or just very difficult otherwise. You should attend this tutorial to learn about the standard container library, what features it provides, and how it solves typical programming problems.



Tutorial Schedule

Monday June 20^{th}	**T 1**	morning	Jean-Pierre Rosen, Adalog, France *Developing Web-aware Applications in Ada with AWS*
	T 2	morning	Peter Amey & Neil White, Praxis High Integrity Systems, UK *Correctness by Construction—A Manifesto for High Integrity Engineering*
	T 3	full day	Benjamin M. Brosgol, AdaCore, USA *Real-Time Java for Ada Programmers*
	T 4	afternoon	Joyce L. Tokar, Pyrrhus Software, USA, Bruce Lewis, US Army, USA *SAE Architecture Analysis and Design Language*
	T 5	afternoon	Brian Dobbing, Praxis High Integrity Systems, UK *High-Integrity Ravenscar Using SPARK*
Friday June 24^{th}	**T 6**	morning	John Barnes, Alan Burns, Pascal Leroy, S. Tucker Taft *Ada 2005*
	T 7	full day	John McDermid, Rob Weaver, University of York, UK *Software Safety Cases*
	T 8	afternoon	Matthew Heaney, On2 Technologies, USA *Programming with the Ada 2005 Standard Container Library*
	T 9	afternoon	Patrick Rogers, AdaCore, USA *Software Fault Tolerance*
	T 10	morning	William Bail, The MITRE Corporation, USA *Requirements Engineering for Dependable Systems*

Lunch is included for those attending full-day programmes or two half day tutorials on the same day. Additional lunch tickets are on sale throughout the conference.
Tutorials will take place from 9:00 to 12:30 (morning session), and from 14:00 to 17:30 (afternoon session). There will be half-hour tea/coffee breaks at 10:30 and 15:30 on Monday and Friday.

Exhibition

The exhibition opens in the mid-morning break on Tuesday and runs until the end of the afternoon break on Thursday. The exhibitors include the following vendors: AdaCore, Green Hills, LDRA, Praxis, ILogix, TNI, ARTiSAN, Esterel, PolySpace, Aonix and Silver Software.

The coffee breaks are held in the same exhibition area. Most breaks Tuesday–Thursday are one hour to allow the attendees a comfortable visit to the exhibition.

Guide to the Social Program

Wednesday Evening: Banquet and Museum

The Conference dinner will take place at the National Railway Museum. This York-based Museum is the largest railway museum in the world, responsible for the conservation and interpretation of the British national collection of historically significant railway vehicles and other artifacts. The Museum contains an unrivaled collection of locomotives, rolling stock, railway equipment, documents and records.

Dinner is served amongst the exhibits and there will be time to look over the whole Museum.

Additional tickets for the banquet can be purchased at registration.

Conference Schedule

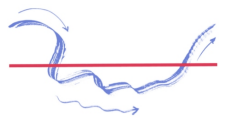
Preliminary version

	Tuesday 21st		Wednesday 22nd	Thursday 23rd
9:00–10:00	Invited Talk: John McDermid, University of York, UK *Model-based Development of Safety-critical Software: Opportunities and Challenges*		Invited Talk: Martyn Thomas, Thomas Associates, UK *Extreme Hubris*	Invited Talk: Bev Littlewood, City University, London, UK *Assessing the Dependability of Software-based Systems: A Question of Confidence*
10:00–11:00	**Exhibition Opening & Coffee**		**Exhibition & Coffee**	**Exhibition & Coffee**
	Applications	**Design and Scheduling Issues**	**Industrial Presentations 1**	**Distributed Systems**
11:00–11:30	N. Rowden *ILTIS—The Legacy of a Successful Product*	S. Sáez, V. Lorente, S. Terrasa, A. Crespo *Efficient Alternatives for Implementing Fixed-priority Schedulers*	A. Calvert (Mondex, Mastercard Int'l) *Design and Maintenance of the MULTOS KMA—A High-Security Ada Application*	D. Ayavoo, M. J. Pont, S. Parker *Observing the Development of a Reliable Embedded System*
11:30–12:00	F. Ortiz, D. Alonso, B. Alvárez, J. A. Pastor *A Reference Control Architecture for Service Robots Implemented on a Climbing Vehicle*	M. Bordin, T. Vardanega *A New Strategy for the HRT-HOOD to Ada Mapping*	J. Rowlands (BAE Systems, UK) *MDA for Ada Software Engineers* J.-P. Rosen (Adalog, France) *On the Benefits for Industrials of Sponsoring Free Software Development*	J. M. Martínez, M. González-Harbour *RT-EP: A Fixed-Priority Real-time Communication Protocol over Standard Ethernet*
12:00–12:30	L. M. Pinho, L. Nogueira, R. Barbosa *An Ada Framework for QoS-Aware Applications*	T. Vergnaud, L. Pautet, F. Kordon *Using the AADL to Describe Distributed Applications From Middleware to Software Components*	O. Hainque (AdaCore, France), R. White (MBDA Missile Systems, UK) *Binding the PowerPC AltiVec Extensions in Ada—Motivation, Development and Industrial Feedback*	M. Masmano, J. Real, A. Crespo, I. Ripoll *Distributing Criticality Across Ada Partitions*
12:30–14:00	**Lunch**		**Lunch**	**Lunch**

Conference Schedule (cont.)

Preliminary version

	Tuesday 21st		Wednesday 22nd	Thursday 23rd
	Formal Methods	**Vendor Session 1**	**Industrial Presentations 2**	**Language Issues**
14:00–14:30	D. Atiya, S. King *Extending Ravenscar with CSP channels*		R.Stråhle, P. Sandberg (SAAB Systems, Sweden) *Living in Towers*	J. Miranda, E. Schonberg, G. Dismukes, *The Implementation of Ada 200 Interface Types in the GNAT Compiler*
14:30–15:00	S. Evangelista, C. Kaiser, C. Pajault, J. F. Pradat-Peyre, P. Rousseau *Dynamic Tasks Verification with Quasar*	See final program for schedule	B. Stadlmann (University of Applied Sciences, Wels, Austria) *Ada Development for a Basic Train Control System for Regional Branchlines*	M. Aldea-Rivas, J. Miranda, M. González-Harbour *Integrating Application-Defined Scheduling with the New Dispatching Policies for Ada Tasks*
15:00–15:30	R. E. Sward, L. C. Baird III *Proving functional equivalence for program slicing in SPARK™*		A. Brandon (ARA) *The Ada Embedded Systems Market Today and Tomorrow*	P. Rogers, A. J. Wellings *The Application of Compile-time Reflection to Software Fault Tolerance Using Ada 95*
15:30–16:00	**Exhibition & Coffee**		**Exhibition & Coffee**	**Exhibition & Coffee**
16:00–16:30				**Ravenscar Technology** J. F. Ruiz *GNAT Pro for On-Board Mission-Critical Space Applications*
	Ada and Education	**Vendor Session 2**	**Industrial Presentations 3**	
16:30–17:00	D. Simon, G. Vogel, E. Plödereder *Teaching Software Engineering with Ada95*		M. Adams (QinetiQ, UK) *Industrial Experiences of Compliance Analysis of Control System Software*	R. Berrendonner, J. Guitton *The ESA Ravenscar Benchmark*
17:00–17:30	B. M. Brosgol *A Comparison of the Mutual Exclusion Features in Ada and the Real-Time Specification for Java™*	See final program for schedule	N. Davidson (BAE Systems, UK) *Safety Critical Software with UML, Artisan and SPARK*	**Closing Session & Awards**
	Certification and Verification			
17:30–18:00	P. Amey, R. Chapman, N. White *Smart Certification of Mixed Criticality Systems*		A. Marriott (White Elephant GmbH, Switzerland) *Ada Bug Finder*	
18:00–18:30	K. Lundqvist, J. Srinivasan, S. Gorelov *Non-intrusive System Level Fault-Tolerance*			
18:30–	**Ada-Europe General Assembly**		**Conference Banquet**	



Registration and Accommodation

Conference Registration

Three days of conference (June 21st–June 23rd) including one copy of the proceedings, coffee breaks, lunches, as well as the museum and banquet on Wednesday 22$^{\text{nd}}$. All prices in British Pounds Sterling.

	member Ada-Europe or ACM SIGAda		non member	
	non academia	academia	non academia	academia
Early registration (by May 23$^{\text{rd}}$)	£380	£340	£410	£380
Late registration (after May 23$^{\text{rd}}$)	£410	£410	£450	£450
Individual day registration (per day)	£190	£190	£210	£210

Tutorial Registration

The prices are per tutorial, including tutorial notes and coffee breaks. Lunches are only included when registered for a full day tutorial or two half day tutorials on the same day.

	Ada 2005 Tutorial	half day	full day or two half days on the same day
Early registration (by May 23$^{\text{rd}}$)	£75	£95	£170
Late registration (after May 23$^{\text{rd}}$)	£85	£105	£190

Please Note

No registration request will be confirmed until the payment has been processed. Cancellations must be given in writing. A cancellation fee of £100 will be applied to all cancellations. No refunds will be given for cancellations received after June 1$^{\text{st}}$. Substitutions will be accepted. To save on administrative costs and postage, receipts will be given out at the conference.

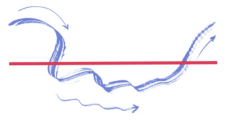
For latest information see the web page at **http://www.ada-europe.org/conference2005.html**. For additional information, please contact: Alan Burns (conference chair), Department of Computer Science, University of York, York, YO10 5DD, UK. Tel: +44 1904 2779, Fax: +44 1904 434331. Email: `burns@cs.york.ac.uk`

Accommodation

We have negotiated specially reduced rates at the Royal York (booking deadline applies) and Marriott. Please consult their own web sites, or the following for more information about these hotels and other establishments before making your booking. There are also numerous online booking agencies which can provide a good rate.

Further information is available on the conference website:
http://www.ada-europe.org/conference2005.html.

Please book accommodation as early as possible
York gets very busy at this time of year.



10th International Conference on Reliable Software Technologies - Ada-Europe 2005

York, UK, June 20-24, 2005

REGISTRATION FORM

Online registration available: http://www.ada-europe.org/conference2005.html

PARTICIPANT *Please use block capitals*

Ms/Mrs [ ] Mr [ ] Title _____

Family name _____ First name _____

Affiliation/Organization _____

Street _____

City _____ Post/Zip code _____ Country _____

Telephone _____ Fax _____ Email _____

Special requirements (e.g. diet) _____

Reduced registration fee

[ ] member Ada-Europe; national organization _____ [ ] academia

[ ] member ACM; membership number _____

Additional Comments _____

Registration time Early registration (by May 23rd) [ ] Late or on site (after May 23rd) [ ]

REGISTRATION FEES (see table on previous page)

Conference registration fee

Three day conference ..... £ _____

Individual days (Tue [ ] Wed [ ] Thu [ ]) ..... £ _____

Tutorial registration Please indicate the tutorials for which you want to register:

Monday, June 14th T1 [ ] T2 [ ] T3 [ ] T4 [ ] T5 [ ]

Friday, June 18th T6 [ ] T7 [ ] T8 [ ] T9 [ ] T10 [ ]

Tutorial registration fee ..... £ _____

Extra Banquet ticket: _____ tickets @ £39 ..... £ _____

Extra proceedings: _____ proceedings @ £20 ..... £ _____

TOTAL PAYMENT DUE ..... £ _____

PAYMENT METHOD

By credit card. Credit card information:

[ ] Visa [ ] Mastercard Credit Card Number: _____

Cardholder Name: _____

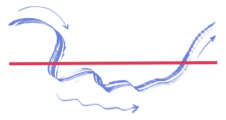
Credit Card Expiration Date: _____

Signature: _____

By bank draft to account number: GB37MIDL40473120898201. The account is for the University of York and is held at HSBC Bank, 13 Parliament Street, York, Y01 8XS, UK. Please give a clear reference to the bank i.e. 'Ada-Europe 2005' and participant's name. Also please attach a copy of the bank draft to this form.

Post or fax this form to: Mrs Sue Helliwell, Department of Computer Science, University of York, Heslington, York, YO10 5DD, U.K. Fax : (+44) 1904 434331

There are a number of other "in cooperation" conferences in this area (such as the ACM SIGAda Conference). Tick this box if you do **not** want to receive information about these conferences: ..... [ ]



Organization

Conference Chair
Alan Burns
University of York
Dept of Computer Science
York, YO10 5DD, UK
burns@cs.york.ac.uk

Program Co-chairs
Tullio Vardanega
Università di Padova
Dipartimento di Matematica
Via Belzoni 7, 35131 Padova, Italia.
tullio.vardanega@math.unipd.it

Andy Wellings
University of York
Dept of Computer Science
York, YO10 5DD, UK
andy@cs.york.ac.uk

Tutorial Chair
Iain Bate
University of York
Dept of Computer Science
York, YO10 5DD, UK
iain.bate@cs.york.ac.uk

Exhibition Chair
Rod Chapman
Praxis High Integrity Systems
20 Manvers Street, Bath,
BA1 1PX, UK
rod.chapman@praxis-his.com

Publicity Co-chairs
Ian Broster
University of York
Dept of Computer Science
York, YO10 5DD, UK
ianb@cs.york.ac.uk

Dirk Craeynest
Aubay Belgium & K.U. Leuven
B-1180 Brussels, Belgium
Dirk.Craeynest@cs.kuleuven.ac.be

Local Organization Administrator
Sue Helliwell
University of York
Dept of Computer Science
York, YO10 5DD, UK
sue@cs.york.ac.uk

Ada-Europe Conference Liaison
Laurent Pautet
ENST Paris, F-75013 Paris, France
pautet@enst.fr

In cooperation with:

ACM · Ada Europe · The University of York

The organisers thank the exhibitors (preliminary list):

AdaCore The GNAT Pro Company · Aonix · ARTiSAN Software · Esterel Technologies · Green Hills Software, Inc. · I-Logix · LDRA Software Technology · PolySpace Technologies · Praxis High Integrity Systems · Silver Software · tNi Europe

http://www.ada-europe.org/conference2005.html