ADA USER JOURNAL

Volume 36
Number 4
December 2015

Contents

Editorial Policy for Ada User Journal

Publication

Ada User Journal — The Journal for the international Ada Community — is published by Ada-Europe. It appears four times a year, on the last days of March, June, September and December. Copy date is the last day of the month of publication.

Aims

Ada User Journal aims to inform readers of developments in the Ada programming language and its use, general Ada-related software engineering issues and Ada-related activities. The language of the journal is English.

Although the title of the Journal refers to the Ada language, related topics, such as reliable software technologies, are welcome. More information on the scope of the Journal is available on its website at *www.ada-europe.org/auj*.

The Journal publishes the following types of material:

- Refereed original articles on technical matters concerning Ada and related topics.
- Invited papers on Ada and the Ada standardization process.
- Proceedings of workshops and panels on topics relevant to the Journal.
- Reprints of articles published elsewhere that deserve a wider audience.
- News and miscellany of interest to the Ada community.
- Commentaries on matters relating to Ada and software engineering.
- Announcements and reports of conferences and workshops.
- Announcements regarding standards concerning Ada.
- Reviews of publications in the field of software engineering.

Further details on our approach to these are given below. More complete information is available in the website at *www.ada-europe.org/auj*.

Original Papers

Manuscripts should be submitted in accordance with the submission guidelines (below).

All original technical contributions are submitted to refereeing by at least two people. Names of referees will be kept confidential, but their comments will be relayed to the authors at the discretion of the Editor.

The first named author will receive a complimentary copy of the issue of the Journal in which their paper appears.

By submitting a manuscript, authors grant Ada-Europe an unlimited license to publish (and, if appropriate, republish) it, if and when the article is accepted for publication. We do not require that authors assign copyright to the Journal.

Unless the authors state explicitly otherwise, submission of an article is taken to imply that it represents original, unpublished work, not under consideration for publication elsewhere.

Proceedings and Special Issues

The *Ada User Journal* is open to consider the publication of proceedings of workshops or panels related to the Journal's aims and scope, as well as Special Issues on relevant topics.

Interested proponents are invited to contact the Editor-in-Chief.

News and Product Announcements

Ada User Journal is one of the ways in which people find out what is going on in the Ada community. Our readers need not surf the web or news groups to find out what is going on in the Ada world and in the neighbouring and/or competing communities. We will reprint or report on items that may be of interest to them.

Reprinted Articles

While original material is our first priority, we are willing to reprint (with the permission of the copyright holder) material previously submitted elsewhere if it is appropriate to give it a wider audience. This includes papers published in North America that are not easily available in Europe.

We have a reciprocal approach in granting permission for other publications to reprint papers originally published in *Ada User Journal.*

Commentaries

We publish commentaries on Ada and software engineering topics. These may represent the views either of individuals or of organisations. Such articles can be of any length – inclusion is at the discretion of the Editor.

Opinions expressed within the *Ada User Journal* do not necessarily represent the views of the Editor, Ada-Europe or its directors.

Announcements and Reports

We are happy to publicise and report on events that may be of interest to our readers.

Reviews

Inclusion of any review in the Journal is at the discretion of the Editor. A reviewer will be selected by the Editor to review any book or other publication sent to us. We are also prepared to print reviews submitted from elsewhere at the discretion of the Editor.

Submission Guidelines

All material for publication should be sent electronically. Authors are invited to contact the Editor-in-Chief by electronic mail to determine the best format for submission. The language of the journal is English.

Our refereeing process aims to be rapid. Currently, accepted papers submitted electronically are typically published 3-6 months after submission. Items of topical interest will normally appear in the next edition. There is no limitation on the length of papers, though a paper longer than 10,000 words would be regarded as exceptional.

Editorial

In this last issue of Volume 36, we conclude the celebration of the 200th anniversary of Ada Lovelace with two special contributions. In the first, Gabriela Rino Nesin, from the University of Oxford, UK, provides us a vivid and motivating description of the Ada Lovelace Symposium, one of the main bicentennial celebrating events, which took place 9-10 December 2015. Following that, John Barnes provides us with an overview of his speech at the symposium, which goes from Byron to the Ada language, concluding the special featured section of the Journal.

Organising this bicentennial special section featured in Volume 36 of the Ada User Journal has been indeed a very rewarding experience. I hope that the reader has both appreciated and learnt, as we did.

The issue then continues with a technical paper from the Ada-Europe 2016 industrial track, by Barbara Gallina, of Mälardalen University and Luciana Provenzano, of Bombardier Transportation, Sweden, on the application of a model-driven safety certification method in the domain of railway.

The issue also publishes the Proceedings of the "2nd Workshop on Challenges and New Approaches for Dependable and Cyber-Physical System Engineering", co-located with Ada-Europe 2015. This workshop brought together industry and research participants, for a full-day discussion on dependability and critical issues of Cyber-Physical Systems (CPS). The workshop editorial by Daniela Cancila, from CEA LIST, and Charles Robinson, from Thales, R&T, is followed by a set of technical papers, part of the workshop's program. In the first paper, authors from Softeam, France, present a study on the challenges put forward to the design of CPS, and how model-based design helps tackling those. Then, authors from the Silesian University of Technology and the Institute of Medical Technology and Equipment, Poland, describe an actual dependable medical CPS for telecare, as well as its challenges and requirements. In the third paper of the proceedings, authors from the University of Modena and Reggio Emilia, Italy, Barcelona Supercomputing Center, Spain, and University of Siena, Italy, present the view of the AXIOM project, on virtual platforms modelling for next-generation cyber-physical systems. Finally, in the last paper, Silvia Mazzini, from Intecs, Italy, describes the CONCERTO open source methodology for designing and deploying reliable and safe CPS.

As usual the issue also provides the News Digest, and Calendar and Forthcoming Events sections, prepared by the respective Editors, Jacob Sparre Andersen and Dirk Craeynest. The forthcoming events include information on the 7th Ada Developer Room at the Free and Open source Software Developers' European Meeting (FOSDEM 2016), the 2016 International Real-Time Ada Workshop (IRTAW), which will take place in Benicássim, near Valencia, Spain, April 11-13 2016, and last, but definitely not least, the Ada-Europe 2016 conference, which will take place in Pisa, Italy, in the week of 13-17 June, 2016. These are important events for the community, and that require active contribution from each one of us!

Luís Miguel Pinho
Porto
December 2015
Email: AUJ_Editor@Ada-Europe.org

Quarterly News Digest

Jacob Sparre Andersen

Jacob Sparre Andersen Research & Innovation. Email: jacob@jacob-sparre.dk

Contents

Ada-related Events

[To give an idea about the many Ada-related events organised by local groups, some information is included here. If you are organising such an event feel free to inform us as soon as possible. If you attended one please consider writing a small report for the Ada User Journal. —sparre]

Ada Augusta Lovelace Bicentennial

From: Tom Moran <tmoran@acm.org>
Date: Tue, 15 Sep 2015 17:55:08 +0000
Subject: Ada Lovelace Bicentennial
Newsgroups: comp.lang.ada

I see the Computer History Museum and ACM will have an exhibit in early December to celebrate the bicentennial of Ada Lovelace.

From: Simon Wright <simon@pushface.org>
Date: Tue, 15 Sep 2015 20:23:27 +0100
Subject: Re: Ada Lovelace Bicentennial
Newsgroups: comp.lang.ada

> [...]

There's also a UK programme [1]; I've booked myself in to the Symposium [2].

[1] http://blogs.bodleian.ox.ac.uk/adalovelace/events/

[2] http://blogs.bodleian.ox.ac.uk/adalovelace/symposium/

From: Jacob Sparre Andersen <jacob@jacob-sparre.dk>
Date: Tue, 24 Nov 2015 18:30:28 +0100
Subject: Ada Augusta Lovelaces 200-års fødselsdag
To: Ada in Denmark members

[Translated summary: —sparre]

December 10 it is the 200 years birthday of Ada Augusta Lovelace. We're going to meet to celebrate the first programmer in the world (and the best programming language in the world :-).

Ada-Europe 2016 in Pisa

From: Dirk Craeynest <dirk@cs.kuleuven.be>
Date: Sun, 18 Oct 2015 06:32:16 +0000
Subject: CfP 21st Conf. Reliable Software Technologies, Ada-Europe 2016
Newsgroups: comp.lang.ada, fr.comp.lang.ada, comp.lang.misc

[CfP is included in the Forthcoming Events Section —sparre]

FOSDEM 2016

From: Dirk Craeynest <dirk@cs.kuleuven.be>
Date: Wed, 4 Nov 2015 07:10:24 -0000
Subject: CfP - Ada Developer Room at FOSDEM 2016, Brussels, Belgium
Newsgroups: comp.lang.ada, fr.comp.lang.ada

---

Call for Presentations

7th Ada Developer Room at FOSDEM 2016

Saturday 30 January 2016, Brussels, Belgium

http://www.cs.kuleuven.be/~dirk/ada-belgium/events/16/160130-fosdem.html

Organized in cooperation with Ada-Europe

---

Ada-Belgium [1] is pleased to announce that there will be a one-day Ada Developer Room on Saturday 30 January 2016 at FOSDEM 2016 in Brussels, Belgium. This Ada DevRoom is once more organized in cooperation with Ada-Europe [2].

General Information

FOSDEM [3], the Free and Open source Software Developers' European Meeting, is a free and non-commercial two-day weekend event organized early each year in Brussels, Belgium. It is highly developer-oriented and brings together 5000+ participants from all over the world. No registration is necessary.

The goal is to provide open source developers and communities a place to meet with other developers and projects, to be informed about the latest developments in the open source world, to attend interesting talks and presentations on various topics by open source project leaders and committers, and to promote the development and the benefits of open source solutions.

Ada Developer Room

At previous FOSDEM events, Ada-Belgium has organized very well attended Ada Developer Rooms, offering a full day program in 2006 [4], a two-day program in 2009 [5], and full day programs in 2012 [6], 2013 [7], 2014 [8], and 2015 [9]. An important goal is to present exciting Ada technology and projects also to people outside the traditional Ada community.

Our proposal for another dedicated Ada DevRoom was accepted, and now work continues to prepare the detailed program. We most probably will have a total of 8 schedulable hours between 11:00 and 19:00 in a room which accommodates 60 participants. More information will be posted on the dedicated web-page on the Ada-Belgium site [10], and final announcements will of course also be sent to various forums, lists and newsgroups.

Call for Presentations

Ada-Belgium calls on you to:

- inform us at ada-belgium-board@cs.kuleuven.be about specific presentations you would like to hear in this Ada DevRoom;
- for bonus points, subscribe to the Ada-FOSDEM mailing list [11] to discuss and help organize the details;
- for more bonus points, be a speaker: the Ada-FOSDEM mailing list is the place to be!

Do you have a talk you want to give?

Do you have a project you would like to present?

Would you like to get more people involved with your project?

We're inviting proposals that are related to Ada software development, and include a technical oriented discussion. You're not limited to slide presentations, of course. Be creative. Propose something fun to share with people so they might feel some of your enthusiasm for Ada!

Speaking slots are 20 or 45 minutes, plus 5 minutes for Q&A. Depending on interest, we might also have a session with lightning presentations (e.g. 5 minutes each).

We'd like to put together a draft schedule early December. So, please act ASAP, and definitely by November 29, 2015 at the latest.

We look forward to lots of feedback and proposals!

Dirk Craeynest, FOSDEM Team of Ada-Belgium

Dirk.Craeynest@cs.kuleuven.be (for Ada-Belgium/-Europe/SIGAda/WG9 mail)

[1] http://www.cs.kuleuven.be/~dirk/ada-belgium

[2] http://www.ada-europe.org

[3] https://fosdem.org

[4] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/06/060226-fosdem.html

[5] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/09/090207-fosdem.html

[6] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/12/120204-fosdem.html

[7] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/13/130203-fosdem.html

[8] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/14/140201-fosdem.html

[9] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/15/150131-fosdem.html

[10] http://www.cs.kuleuven.be/~dirk/ada-belgium/events/16/160130-fosdem.html

[11] http://listserv.cc.kuleuven.be/archives/adafosdem.html

Ada-Belgium Celebration of 200th birthday Ada Lovelace

From: Dirk Craeynest <Dirk.Craeynest@cs.kuleuven.be>
Date: Sat, 5 Dec 2015 21:33:53 +0100
Subject: Ada-Belgium Celebrates 200th Birthday Ada Lovelace 10 Dec 2015
To: ada-belgium-info@cs.kuleuven.be

The Computer Science Department of the KU Leuven and the Ada-Belgium organization are pleased to announce a celebration of the

200th Birthday of Ada Lovelace

on Thursday, December 10, 2015, 20:00 at the KU Leuven, Department of Computer Science, Auditorium

Celestijnenlaan 200A, B-3001 Leuven (Heverlee), Belgium

www.cs.kuleuven.be/~dirk/ada-belgium/events/15/151210-ab-ada200.html

----

On the occasion of Ada Lovelace's 200th birthday, the Computer Science Department of the KU Leuven together with Ada-Belgium organize a small celebration, featuring a screening of the "To Dream Tomorrow" documentary on Ada Lovelace.

Synopsis:

"To Dream Tomorrow" is the story of Ada Byron Lovelace and her contributions to computing, over a hundred years before the time usually thought to be the start of the Computer Age.

Daughter of a mathematically gifted, social activist mother and the "mad, bad and dangerous to know" poet Lord Byron, Ada was 17 when she began studying a prototype mechanical calculator designed by mathematician Charles Babbage.

By the time she was 27, she had moved even beyond her famous contemporaries to describe universal computing much as we understand it today.

(Language: English; Duration: 52 min.)

This event will be held at the Computer Science Department of the KU Leuven in Leuven (Heverlee), on Thursday, December 10, 2015.

Starting time is 20:00, and the program consists of a short introduction, a screening of the documentary, and an informal drink to socialize and network.

Context

10 December 2015 is the 200th anniversary of the birth of Ada Lovelace.

She is seen by many as the first programmer due to her work on Charles Babbage's Analytical Engine, a mechanical general-purpose computer he designed but didn't manage to build.

The Ada programming language is named after Ada Lovelace. It is an evolving state-of-the-art programming language especially suitable for large, long-lived applications where safety, security, reliability, and efficiency are critical.

Current application areas include air traffic control and management, airplane engines and systems, railway signalling and transportation, space missions, banking and financial systems, industrial command and control systems, etc. Due to its approach of detecting errors as soon as possible its use offers valuable advantages, even for less demanding applications.

Ada-Belgium is a non-profit organization that aims to be a forum for persons and organizations interested in the Ada programming language, in its applications and in Ada related technologies such as software engineering methods, environments and tools.

Participation

Everyone is welcome, and participation is free, but for practical reasons we'd appreciate if you could inform us of your intent to be there.

Please provide your name, email address and affiliation, by email to <ada-belgium-board@cs.kuleuven.be>.

For directions to the Computer Science Department of the KU Leuven, see <http://wms.cs.kuleuven.be/cs/english/about/directions>.

Ada-related Resources

Ada on Social Media

From: Jacob Sparre Andersen <jacob@jacob-sparre.dk>
Date: Wed Nov 18 2015
Subject: Ada on Social Media

Ada groups on various social media:

- LinkedIn[1]: 2_329 members

- Reddit[2]: 850 readers

- Google+[3]: 546 members

- StackOverflow[4]: 327 followers

- Twitter[5]: 5 tweeters

[1] https://www.linkedin.com/groups?gid=114211

[2] http://www.reddit.com/r/ada/

[3] https://plus.google.com/communities/102688015980369378804

[4] http://stackoverflow.com/questions/tagged/ada

[5] https://twitter.com/search?f=realtime&q=%23AdaProgramming

[See also “Ada on Social Media”, AUJ 36-3, p. 121. —sparre]

Repositories of Open Source Software

From: Jacob Sparre Andersen <jacob@jacob-sparre.dk>
Date: Wed Nov 18 2015
Subject: Repositories of Open Source software

GitHub: 1_064 repositories [1]

280 developers [1]

Rosetta Code: 624 examples [2]

30 developers [3]

Sourceforge: 244 repositories [4]

BlackDuck OpenHUB: 213 projects [5]

Bitbucket: 188 repositories [6]

OpenDO Forge: 23 projects [7]

466 developers [7]

Codelabs: 20+ repositories [8]

AdaForge: 8 repositories [9]

Assembla: 6 projects [10]

[1] https://github.com/search?q=language%3AAda&type=Repositories

[2] http://rosettacode.org/wiki/Category:Ada

[3] http://rosettacode.org/wiki/Category:Ada_User

[4] http://sourceforge.net/directory/language%3Aada/

[5] https://www.openhub.net/tags?names=ada

[6] https://bitbucket.org/repo/all?name=ada

[7] https://forge.open-do.org/

[8] http://git.codelabs.ch/

[9] http://forge.ada-ru.org/adaforge

[10] https://www.assembla.com/tag/ada

[See also "Repositories of Open Source Software", AUJ 36-3, p. 121. —sparre]

Ada-related Tools

ZanyBlue

From: Michael Rohan <michael@zanyblue.com>
Date: Sun, 13 Sep 2015 14:30:32 -0700
Subject: ANN: ZanyBlue v1.2.1 Beta Available
Newsgroups: comp.lang.ada

A new release of ZanyBlue is now available: 1.2.1 Beta. This is an Ada library currently targeting localization support for Ada (along the lines of Java properties) with supporting message formatting and built-in localization for about 20 locales. The properties files are compiled into Ada sources built with your application and use to access application messages at run-time. The run-time locale is used to select localized messages, if they are available.

The changes for this release are

- Updates for building with GNAT 2015 (major driver for this release).
- Updated CLDR to release 26 from release 24.

Please see the project page on Source Forge for download links, documentation, etc,

http://zanyblue.sourceforge.net

This project is licensed under a simple BSD style license.

[See also "ZanyBlue", AUJ 35-2, p. 75. —sparre]

GtkAda

From: Nicolas Setton <setton@adacore.com>
Date: Tue, 6 Oct 2015 13:31:25 -0400
Subject: the GtkAda repository is now on github
Newsgroups: gmane.comp.gnome.gtk+.ada
To: gtkada@lists.adacore.com

GtkAda is now on github, at:

https://github.com/AdaCore/gtkada

We hope this will make everyone's workflows easier and collaboration simpler!

If you have any question, don't hesitate to ask on this list.

In addition to this list, I noticed that people are answering GtkAda questions on stackoverflow.com, and also on comp.lang.ada - many thanks to them!

RAPID

From: Oliver Kellogg <okellogg@users.sourceforge.net>
Date: Sat, 10 Oct 2015 07:07:40 -0700
Subject: Re: RAPID 3.3 is released
Newsgroups: comp.lang.ada

Sadly, I have not been finding time for further work on RAPID.

On the upside, I have made a perl script that translates the RAPID GUI file format to Gtk's Glade-3 Builder UI format, see

http://svn.savannah.nongnu.org/viewvc/*checkout*/trunk/gtk_bin/rapid2glade.pl?root=rapid

Recently, I have extended the script to generate Ada support code which eases the transition from RAPID to GtkAda.Builder.

[See also "RAPID 3.3", AUJ 32-2, p. 85. —sparre]

Simple Components

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Mon, 19 Oct 2015 18:49:45 +0200
Subject: ANN: Simple components 4.10 released
Newsgroups: comp.lang.ada

The current version provides implementations of smart pointers, directed graphs, sets, maps, B-trees, stacks, tables, string editing, unbounded arrays, expression analyzers, lock-free data structures, synchronization primitives (events, race condition free pulse events, arrays of events, reentrant mutexes, deadlock-free arrays of mutexes), pseudo-random non-repeating numbers, symmetric encoding and decoding, IEEE 754 representations support, multiple connections server/client designing tools.

http://www.dmitry-kazakov.de/ada/components.htm

Changes to the previous version:

- ELV/e-Q3 MAX! client wall thermostat support added;
- ELV/e-Q3 MAX! device data available through Get_Device_Data;
- ELV/e-Q3 MAX! interface calls querying topology, device parameters and data are task-safe;
- Socket_Error 11004 workaround added.

[See also "Simple Components", AUJ 36-3, p. 122. —sparre]

GtkAda Contributions

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Tue, 20 Oct 2015 18:23:34 +0200
Subject: *ANN: GtkAda contributions v3.14 released*
Newsgroups: comp.lang.ada

The library extends GtkAda 3.8.3/4. It deals with the following issues:

- Tasking support;
- Custom models for tree view widget;
- Custom cell renderers for tree view widget;
- Multi-columned derived model;
- Extension derived model (to add columns to an existing model);
- Abstract caching model for directory-like data;
- Tree view and list view widgets for navigational browsing of abstract caching models;
- File system navigation widgets with wildcard filtering;
- Resource styles;
- Capturing resources of a widget;
- Embeddable images;
- Some missing subprograms and bug fixes;
- Measurement unit selection widget and dialogs;
- Improved hue-luminance-saturation color model;
- Simplified image buttons and buttons customizable by style properties;
- Controlled Ada types for GTK+ strong and weak references;
- Simplified means to create lists of strings;
- Spawning processes synchronously and asynchronously with pipes;
- Capturing asynchronous process standard I/O by Ada tasks and by text buffers;
- Source view widget support.

http://www.dmitry-kazakov.de/ada/gtkada_contributions.htm

Changes to previous version:

- Minor bug fixes;
- GTK 3.10 compatibility issues.

[See also "GtkAda Contributions", AUJ 36-3, p. 125. —sparre]

Industrial Control Widget Library

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Wed, 21 Oct 2015 18:54:22 +0200
Subject: ANN: Ada industrial control widget library v3.12 released

Newsgroups: comp.lang.ada

The library is provided for design high-quality industrial control widgets for Ada applications. The software is based on GtkAda, Ada bindings to Gtk+ and Cairo. The key features of the library:

- Widgets composed of transparent layers drawn by cairo; - Fully scalable graphics;
- Support of time controlled refresh policy for real-time and heavy-duty applications;
- Caching graphical operations;
- Stream I/O support for serialization and deserialization;
- Ready-to-use gauge, meter, oscilloscope widgets;
- Editor widget for WYSIWYG design of complex dashboards.

http://www.dmitry-kazakov.de/ada/aicwl.htm

The new version is adapted to GNAT 5 now available for Debian.

[See also "Industrial Control Widget Library", AUJ 35-3, p. 157. —sparre]

PragmAda Reusable Components

From: PragmAda Software Engineering <pragmada@pragmada.x10hosting.com>
Date: Tue, 20 Oct 2015 15:09:38 -0700
Subject: New Release of PragmAda Reusable Components Beta Version
Newsgroups: comp.lang.ada

A new release of the beta version of the PragmARCs for compilers that support ISO/IEC 8652:2007 is now available at

https://pragmada.x10hosting.com/pragmarc.htm

The main change is an improved interface for the REM neural network component, and the ability to use multiple tasks to run a network.

[See also "PragmAda Reusable Components", AUJ 35-3, p. 154. —sparre]

AdaControl

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Fri, 6 Nov 2015 17:39:09 +0100
Subject: [Ann] AdaControl 1.17r3 released
Newsgroups: comp.lang.ada

Adalog is happy to announce the release of version 1.17r3 of AdaControl.

This version brings its usual load of improvements and new (sub)rules, but the most important improvements are in better support of Ada 2005/2012 constructs... especially in the handling of (#!?GRR...) anonymous access types that were overlooked in previous versions.

As usual, it can be downloaded from the AdaControl page on

http://www.adalog.fr

Remember: if you are a professional user, a great support contract is available from Adalog. Don't hesitate to get in touch.

[See also "AdaControl", AUJ 36-1, p. 13. —sparre]

GNAT: Ada.Strings.* Omissions

From: Jeffrey R. Carter <jrcarter@acm.org>
Date: Thu, 12 Nov 2015 15:33:40 -0700
Subject: Re: Unicode string comparison functions
Newsgroups: comp.lang.ada

> ...Wide_Wide_Equal_Case_Insensitive --- Testing it out, the compiler's giving me an error saying ""Ada.Strings.Wide_Wide_Equal_Case_Insensitive" is not a predefined library unit", which is good in that it saves me from feeling **REALLY** stupid.

ARM A.4.8 clearly indicates that there should be such a library function, similar to Ada.Strings.Equal_Case_Insensitive from A.4.10 for String.

They were added by Ada 2012, so if you're using a compiler for an earlier version they probably won't exist. If you are using an Ada 2012 compiler, you should probably complain.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 18:10:32 -0600
Subject: Re: Unicode string comparison functions
Newsgroups: comp.lang.ada

> [...]

Right, assuming you spelled it right (with the Wide Wide madness, spelling it wrong isn't that uncommon!).

From: Simon Wright <simon@pushface.org>
Date: Fri, 13 Nov 2015 08:22:07 +0000
Subject: Re: Unicode string comparison functions
Newsgroups: comp.lang.ada

> [...]

GNAT GPL 2015 provides Equal_Case_Insensitive only in plain Strings, Bounded, Fixed, Unbounded.

Limited Bounded Strings

From: Jeffrey R. Carter <jrcarter@acm.org>
Date: Thu, 12 Nov 2015 18:01:21 -0700
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

If anyone's interested, there's now a bounded-string pkg in the PragmAda Reusable Components for ISO/IEC 8652:2007, based on Duff's suggestion [see "Discriminants with Default Values - Limited Types" later in this issue. — sparre]. It's pretty much untested, but feel free to play with it.

https://pragmada.x10hosting.com/pragmarc.htm

[See also "PragmAda Reusable Components" earlier in this section. —sparre]

Ada-related Products

GNAT GPL for Raspberry Pi 2

From: AdaCore Press Center
Date: Tue Sep 1 2015
Subject: AdaCore Introduces GNAT GPL 2015 for the Raspberry Pi 2
URL: http://www.adacore.com/press/adacore-introduces-gnat-gpl-2015-for-the-raspberry-pi-2/

Latest version of AdaCore cross-development environment targets students and other developers of nonproprietary software.

NEW YORK and PARIS, Sep 1, 2015 – AdaCore, the leading provider of commercial software solutions for the Ada programming language, today released a freely downloadable version of its GNAT GPL Ada cross-development environment for the Raspberry Pi 2 micro-PC running Embedded Linux. With this new cross-development environment, professors, students, hobbyists and others can take advantage of Ada 2012's reliability, safety and security benefits for their Raspberry Pi 2 applications.

GNAT GPL provides a complete Ada 2012 development environment, including a comprehensive tool-chain as well as AdaCore's flagship GNAT Programming Studio (GPS) Integrated Development Environment (IDE). GNAT GPL implements the Ada 2012 language standard by default, which includes these important language features:

- Contract-based programming (preconditions, post-conditions, and type invariants) including support for the Liskov Substitution Principle in Object-Oriented Programming
- More general expressions (conditional expressions, quantified expressions, expression functions)
- Enhanced multiprocessor support (multiprocessor affinity and barriers)
- Enhanced integration of concurrency and OOP
- Additional language-defined libraries (vector/matrix libraries)

"With more than 5 million units sold to date, Raspberry Pi is one of the world's most popular single-board computers for young computer innovators and hobbyists," said Jamie Ayre, Marketing Director of AdaCore. "By providing Raspberry Pi 2 users access to the very robust, high-integrity development environment of Ada, we're opening the door to some really creative solutions and the next generation of Ada programmers."

With the release of GNAT GPL for Bare Board ARM in 2014, an implementation on the Raspberry Pi 2 running Linux on ARM was a natural follow-up. It reflects AdaCore's ongoing commitment to the Ada community to provide freely available Ada implementations for developers of nonproprietary software. Fully featured releases of this GNAT technology are also available for GNU Linux, Mac OS X, Bare Board ARM, and Windows.

About Raspberry Pi 2

The Raspberry Pi 2 Model B is the second generation Raspberry Pi, released in February 2015. The 900MHz quad-core ARM Cortex-A7 CPU increases the performance almost 6 times, and the 1GB LPDDR2 SDRAM accommodates larger and faster systems.

The Raspberry Pi 2 Model B retains the various interfaces of its predecessor, such as 4 USB ports, 40 GPIO pins, a full HDMI port, and an Ethernet port. With its low cost / high performance advantages, the Raspberry Pi 2 is an attractive choice in many kinds of systems including Internet of Things (IoT) applications.

[...]

Rapita Verification Suite

From: Rapita Systems
Date: Wed Oct 21 2015
Subject: Rapita Systems launches the latest version of RVS at the MCM ITP Conference, Brighton 2015
URL: https://www.rapitasystems.com/news/rapita-systems-launches-latest-version-rvs-mcm-itp-conference-brighton-2015

The latest version of RVS, version 3.4, brings several new features to the table as part of the ongoing tool suite development. Justifications workflow, incremental coverage and livemaps are all part of the latest release bringing a variety of advantages and benefits to customers of Rapita Systems.

Dr Andrew Coombes, Head of Marketing and Product Development at Rapita Systems, describes RVS 3.4 as 'an exciting development in the RVS product line. We've listened to our customers and these are several of the features they have been asking for. We are very confident that the latest features will benefit our current customers as well as our future customers as we continue to strive for product excellence and build strong relationships with the people who use our tools. '

RVS 3.4 will be launched on October 22, 2015 and more details of the latest features can be found on Rapita Systems website [1].

[1] https://www.rapitasystems.com/rvs34

[See also "Rapita Verification Suite", AUJ 36-2, p. 66. —sparre]

InterCOM DDS

From: Egil Harald Høvik <egil.harald.hoevik@kongsberg.com>
IRC-network: Freenode
IRC-channel: #Ada
Date: Thu Nov 19 2015

< egilhh_> hmmm... I'm doing C right now...

< egilhh_> at least I'm generating Ada :)

< egilhh_> sparre: I would like to do that, but I'm expanding an existing codebase...

< Visaoni> egilhh: You have C generating Ada?

< egilhh_> Visaoni: yup

< Visaoni> Interesting. Can you say what for, generally?

< egilhh_> Visaoni: Ada support for http://www.kongsberggallium.com/products/intercom-dds

< sparre> egilhh: Doesn't sound like a bad task, even if you have to write some C to do it.

< Visaoni> egilhh: Nice. Looks cool

< egilhh_> it's pretty interesting

[See http://www.kongsberggallium.com/docs/intercom_dds_datasheet.pdf for information about the currently available version. —sparre]

Ada and Operating Systems

Mac OS X: XNAdaLib

From: Pascal Pignard <p.p11@orange.fr>
Date: Thu, 03 Sep 2015 21:12:06 +0200
Subject: [ANN] XNAdaLib 2015 binaries for MacOS 10.9 including GTKAda GPL 2015 and more.
Newsgroups: comp.lang.ada

This is XNAdaLib 2015 built on MacOS 10.9 Mavericks for Native Quartz including:

- GTK Ada GPL 2015 with GTK+ 3.16.0 complete,
- Glade 3.18.3,
- GnatColl GPL 2015,
- Florist GPL 2015,
- AdaCurses 20110404,
- Gate 3-04-b,
- AICWL 3.11 (with Components 4.8 and gtksourceview 3.14.3),
- GNOGA 1.1a,

to be installed for instance at /usr/local:

```
$ cd /usr/local
$ sudo tar xzf xnadalib-gpl-2015-quartz-
x86_64-apple-darwin13.4.0-bin.tgz
```

Update your PATH to include gtkada-config, glade, gate3.sh and other executables in it:

```
$ PATH=/usr/local/xadalib-2015/bin:$PATH
```

Update your GPR_PROJECT_PATH to include gtkada.gpr, adacurses.gpr, florist.gpr, gnatcoll.gpr, gnoga.gpr and other projects in it:

```
$ export
GPR_PROJECT_PATH=/usr/local/
xnadalib-2015/lib/gnat:/usr/local/
xnadalib-2015/share/
gpr:$GPR_PROJECT_PATH
```

Set XDG_DATA_DIRS for GNOME apps:

```
$ export XDG_DATA_DIRS=/usr/local/
xnadalib-2015/share
```

Glade and GPS applications in apps directory must stay in this directory unless you modify the script inside apps.

Then see documentation and examples in share directory and enjoy.

Here is the instructions I used to build XNAdaLib on MacOS:

(French language)

http://blady.pagesperso-orange.fr/telechargements/gtkada/Install-GTKAda-Quartz_wf.pdf

Here is the modifications I made:

http://blady.pagesperso-orange.fr/telechargements/gtkada/xadalib-2015-diff.tgz

XAdaLib binaries have been post on Source Forge:

http://sourceforge.net/projects/gnuada/files/GNAT_GPL%20Mac%20OS%20X/2015-mavericks/

[See also "MacOS X: XNAdaLib", AUJ 35-4, p. 221. —sparre]

Mac OS X: GtkAda

From: Ahlan Marriott <ahlan@marriott.org>
Date: Sat, 5 Sep 2015 12:50:07 -0700
Subject: Re: Gtkada on Mac using Quartz
Newsgroups: comp.lang.ada

In case anyone is interested, I managed to get my non-trivial GUI test program, written using GNAT-GPL-2015 and GPL-2015 GtkAda to work without any source changes on all three of my selected targets, namely Windows (XP), Linux (Ubuntu) and OS X (Yosemite).

Yosemite was the biggest challenge because AdaCore for some reason didn't provide a GtkAda installation as they did for Windows and Linux. Which meant that I had to get and compile Gtk myself. Fortunately the linux installation worked after rewriting my GUI package to use the main thread, the program ran on all platforms. Which, I think, is rather cool!

Windows XP: Working GtkAda Versions

From: Ahlan Marriott <ahlan@marriott.org>
Date: Fri, 18 Sep 2015 08:44:09 -0700
Subject: Gtk 3.8.4 (and after) do not fully support WinXp
Newsgroups: comp.lang.ada

An Ada program that uses GTK for its GUI fails if it references a DLL written in Ada. A Visual C++ Runtime Library pop-up is raised with the cryptic verse "This application has requested to terminate it in an unusual way". This piece of nonsense actually means that the C runtime has raised an exception that is never caught which results in the abort function being called. The problem here is that it doesn't give us much clue as to what the exception is. However I believe that this error message is produced when a missing function is called from a system DLL. I.e. it can't find the entry point within the DLL.

The prime candidate for this is LibGlib-2.0-0.Dll which references GetTickCount64 and InitializeSRWLock from Kernel32.dll. Unfortunately these routines are not present in versions prior to V6 which only came with Windows Vista.

Therefore GTK 3.8.4 provided as part of the AdaCore GtkAda release for GPL 2015 does not fully support Windows XP.

Which is very odd as support of XP was only officially dropped with GTK 3.17.1

However Gnome seem to have broken it already in v3.8.4

For full Windows XP support it is therefore better to revert to v3.8.2 that came bundled with the GtkAda release for GPL 2014.

It is hard to find out what exactly changed between 3.8.2 and 3.8.4 but the numbering seems to indicate that there were no major changes or enhancements. Except dropping XP that is!

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Fri, 18 Sep 2015 18:02:46 +0200
Subject: Re: Gtk 3.8.4 (and after) do not fully support WinXp
Newsgroups: comp.lang.ada

[...] GTK maintainers do not care about backward compatibility. There is a lot of stuff that ceases to work with each new GTK version. I tracked most of that to make GtkAda 3.8.3 working with GTK 3.10.x. I don't have 3.8.4, so I cannot tell, but my GtkAda applications work under both Windows and fully updated Linux. OK on OS X. The other problem is that on Yosemite the Gtk thread has to be the Major offenders are dialog boxes and stock items. They should generate warnings but actually they tend to crash the application. Other things are Get_String and Set_String of the tree and, possibly, of the list model. These corrupt stack and memory pool after several calls. The pixbuff cell renderer works no more because they are going to kill stock items. And so on. Some workarounds I did are here:

http://www.dmitry-kazakov.de/ada/gtkada_contributions.htm

In all cases fill your application with tracing output to localize the problem.

Android: GNAT?

From: Cleverson Casarin Uliana <clcaul@live.com>
Date: Sat, 19 Sep 2015 08:38:01 -0700
Subject: Can I make native Android apps with Ada?
Newsgroups: comp.lang.ada

I'm new to Ada and was already able to make basic programs for Windows. Now I'm looking to see if it was possible to make Android apps. I found a press release - http://www.adaic.org/press/gnat-pro-for-android/

It appears I have to get the GNAT GPL (since I want to build my own free app), but I'm unable to find any basic documentation on how to do it, e.g., what's the correct software pieces to get, what dependencies it requires to build Android apps on Windows, some general steps for building an Android app, etc. Can you help please?

From: Luke A. Guest <laguest@archeia.com>
Date: Sat, 19 Sep 2015 20:50:44 +0000
Subject: Re: Can I make native Android apps with Ada?
Newsgroups: comp.lang.ada

> [...]

You can build GNAT for Android so you don't have to use the GPL version. There are no bindings afaik to any API.

From: David Botton <david@botton.com>
Date: Sun, 20 Sep 2015 06:30:38 -0700
Subject: Re: Can I make native Android apps with Ada?
Newsgroups: comp.lang.ada

The current best way is to use Gnoga (http://gnoga.com) for Ada to develop any mobile / cloud apps. You can easily create a native Android/iOS app that wraps accessing the Gnoga server side if selling on the stores needed, but sadly there are no currently solutions for running client side. (Not exactly true, the .NET version could be used and the result passed through a .NET to ASM.js compiler available on the net, but licensing encumbrances placed on that version main thread. This restriction does not apply for Windows and Ubuntu. However damper any real interest to pursue solutions with it.)

Debian: SPARK: Request for Adoption

From: Євгеній Мещеряков <eugen-8fiUuRrzOP0dnm+yROfE0A@public.gmane.org>
Date: Tue, 27 Oct 2015 21:47:59 +0100
Subject: Bug#803195: RFA: spark -- SPARK programming language toolset
Newsgroups: gmane.linux.debian.packages.ada

I request an adopter for the spark package.

The current upstream version of the package will not be updated. There is a new version of Spark, but it is very different from this one, see also https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=713928

The package description is:

SPARK is a formally-defined computer programming language based on the Ada programming language, intended to be secure and to support the development of high integrity software used in applications and systems where predictable and highly reliable operation is essential either for reasons of safety or for business integrity.

This package contains the tools necessary for checking if programs adhere to the SPARK rules and the tools to show freedom of runtime exceptions in those programs. To compile SPARK programs use any standards-compliant Ada compiler, such as GNAT.

Windows (64 bit): GNAT

From: David Botton <david@botton.com>
Date: Wed, 11 Nov 2015 07:11:00 -0800
Subject: One stop Win 64bit Ada Dev
Newsgroups: comp.lang.ada

I was very pleased today to see that the Git for Windows SDK includes GNAT 5.2

So with one install you get a full MSYS2 64 bit development environment including Ada

https://github.com/git-for-windows/build-extra/releases/tag/git-sdk-1.0.1

Windows 10: Universal Apps

From: David Botton <david@botton.com>
Date: Tue, 17 Nov 2015 19:11:55 -0800
Subject: Building Windows 10 Universal Apps with Ada
Newsgroups: comp.lang.ada

With Gnoga (http://gnoga.com) it is possible to build "modern" Windows apps with Ada. Until some day we have a working up to date (non GPL encumbered) .NET compiler or LLVM backend part of your code will need to be executed either on a server or the local machine as an older Win64 or Win32 service, but you can take full advantage of the platform and run the client side on an XBox, Windows PC, Phone, etc. (event submit to the app store).

Essentially the steps are:

1. Create a new VisualStudio JS Windows Universal App project.
2. Copy boot.js and jquery.min.js from gnoga/js to your_project/js
3. Modify default.html to include the two js files.
4. Modify boot.js to hard code the server location (ws://snake.gnoga.com:8080/gnoga for example).

Done.

You can use any Windows APIs, etc. by simply binding them like other JS script. If there is interest perhaps I'll bind some of the basics in the future. For those not familiar this solution is very different than serving out web pages, your Ada code will have direct interactive control of every element of the Windows app.

Mac OS X: GCC

From: Simon Wright <simon@pushface.org>
Date: Tue, 24 Nov 2015 17:42:12 +0000
Subject: ANN: GCC 5.2.0 for OS X El Capitan
Newsgroups: comp.lang.ada

This compiler (native only at time of writing) is on Sourceforge[1].

There's not a lot of added Ada goodness compared to 5.1.0, but the El Capitan-related problem that David Botton recently encountered (see [2]) is fixed.

If you need PR66509 fixed and also arm-eabi support, put this compiler on your PATH before the 5.1.0 compiler (e.g. PATH=/opt/gcc-5.2.0/bin:/opt/gcc-5.1.0/bin:/the/rest/of/your/path)

I don't have a computer with Yosemite installed, any brave soul care to give it a try?

[1] https://sourceforge.net/projects/gnuada/files/GNAT_GCC%20Mac%20OS%20X/5.2.0/

[2] https://gcc.gnu.org/bugzilla/show_bug.cgi?id=66509

References to Publications

Comparing with C#

From: Stefan Lucks <stefan.lucks@uni-weimar.de>
Date: Wed, 2 Sep 2015 12:59:40 +0200
Subject: Top 10 Worst C# Features
Newsgroups: comp.lang.ada

Eric Lippert from the C# design team has an interesting article on the ten worst features of C#. At least eight of these have been solved in Ada. Did Jean I. time-travel into the future and learned from the C# mistakes, when designing Ada?

Lean back and enjoy!

http://www.informit.com/articles/article.aspx?p=2425867

BTW, the two issues that are, to some degree, applicable to Ada are

#9 Comparison operators for your own arithmetic (e.g., for your own rational numbers).

In Ada, the "/="-operator is automatically the negation of "=". Which is great. But given "=", "<", and ">", why does one have to implement "<=" and ">=", as well. (Five Operators, where three would suffice.) (*)

The situation with C# is even worse. You need to define/override at least nine methods.

#2 Finaliz(ers) are fragile

In Ada, the finalize procedure for an object can be called more than once, Finalize should rather not raise an exception, ... apparently, C# finalizers suffer from similar problems: "any time a finalizer runs, you could argue that the program either has a bug or is in a dangerous state, such as being shut down unexpectedly via a thread abort".

The remaining eight C# issues have been solved in Ada.

(*) One might argue that two operators, e.g., "=" and "<" would suffice, rather than three. If neither A=B nor A<B, then A>B is obvious, isn't it?

But not all sorts of "arithmetic", where you want to take comparisons, have the property that either of A=B, A>B, A<B is true:

- Arithmetic with "special values", such as "Not a Number" for floating point operations. If A is NaN, then neither A=B nor A>B nor A<B would hold, even if B is also NaN.
- Another example would be sets: A<B would indicate "A is a proper subset of B. If, e.g., A={1} and B={2,3}, then neither A=B nor A<B nor A>B.

On the other hand, A <= B should never mean anything different from

(A < B) or (A = B)!

Similarly for A >= B.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Wed, 2 Sep 2015 14:39:39 -0500
Subject: Re: Top 10 Worst C# Features
Newsgroups: comp.lang.ada

> #2 Finaliz(ers) are fragile [...]

Based on the article, the situation in Ada is far less bad than the C# one. His description of C++ finalizers apply to Ada's as well: "[they] run deterministically, run on the current thread, and never run on partially constructed objects." Moreover, they always run when objects are destroyed, while apparently in C# they don't run for explicit calls to Dispose.

Ada's issues come from two things: extremely rare cases involving exceptions where a finalizer might start twice, and the fact that someone can explicitly call it on an object. If one has control over the entire system, neither of these cases need happen; a program like AdaControl can enforce appropriate style rules to avoid problems.

If not, it's relatively easy to make Finalize callable multiple times (you just need a "Valid" bit in the object, which is turned off by Finalize; if it's off, Finalize does nothing).

Of course, Ada gets this by not supporting garbage collection on objects that have non-trivial finalization (such an object cannot be garbage collected until the finalization has run, meaning they have to exist until they go out of scope, meaning no useful garbage collection can happen. Fixing that would probably introduce more problems (although if the points at which garbage collection could happen were appropriately limited it would not be a huge issue; if this issue is ever addressed in Ada, which would have to be the case).

In any case, Ada's situation is nothing like the asynchronous mess he describes for C#. It may not be perfect, but it's a heck of a lot better than C#.

Building High Integrity Applications with SPARK

From: Roderick Chapman <roderick.chapman@googlemail.com>
Date: Thu, 3 Sep 2015 01:10:33 -0700
Subject: New SPARK book is published today
Newsgroups: comp.lang.ada

I'm pleased to say that the all-new SPARK book by John McCormick and Peter Chapin is published today. It covers the SPARK 2014 language and tools in some detail. Available for order on Amazon and all the usual places now. For example:

http://www.amazon.com/Building-High-Integrity-Applications-SPARK/dp/1107656842/ref=sr_1_1?ie=UTF8&qid=1441267363&sr=8-1&keywords=mccormick+chapin+spark

From: Dirk Craeynest <dirk@cs.kuleuven.be>
Date: Thu, 3 Sep 2015 10:08:59 +0000
Subject: Re: New SPARK book is published today
Newsgroups: comp.lang.ada

> [...] Some information is available on the publisher's web-page for the book. See the tab "Contents" on:

<http://www.cambridge.org/us/academic/subjects/computer-science/programming-languages-and-applied-logic/building-high-integrity-applications-spark>

From: Georg Bauhaus <bauhaus@futureapps.de>
Date: Fri, 18 Sep 2015 16:06:46 +0200
Subject: Re: New SPARK book is published today
Newsgroups: comp.lang.ada

A Kindle "sample" may now be received from Amazon; it includes the Preface, with chapter outlines, some of the introductory chapter, and also this TOC:

Comparing SPARK and Ada

From: Claire Dross, AdaCore
Date: Thu Nov 19 2015
Subject: What's the Difference Between Ada and SPARK?
URL: http://electronicdesign.com/dev-tools/what-s-difference-between-ada-and-spark

Ada is a general-purpose language, like C++ or Java, supporting the usual features of modern programming languages, such as data encapsulation, object orientation, templates (called “generics”), exceptions, and tasking. Originally defined in 1983, it has undergone several revisions, the latest in 2012. What sets Ada apart from other general-purpose languages is that it was designed from the start with reliability, safety, and security in mind. Not surprisingly, Ada is used in domains where the correctness of software is critical: space, avionics, air-traffic control, railway, and military.

SPARK is a specialized subset of Ada designed to facilitate the use of formal methods, so that correctness of software or other program properties can be guaranteed with mathematics-based assurance. Therefore, SPARK is used in the same domains as Ada, by those who value the strong guarantees offered by formal methods.

[...]

Ada Inside

MAX! Home Automation

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Mon, 24 Aug 2015 18:53:55 +0200
Subject: ANN: MAX! home automation v1.0 released
Newsgroups: comp.lang.ada

MAX! home automation is a GTK+ application to manage ELV/eQ-3 MAX! cubes. A cube is a gateway to a network of radiator thermostats, shutter contacts etc.

http://www.dmitry-kazakov.de/ada/max_home_automation.htm

LinXtris

From: Pascal Pignard <p.p11@orange.fr>
Date: Wed, 16 Sep 2015 18:47:02 +0200
Subject: LinXtris for GTKAda 3.
Newsgroups: gmane.comp.gnome.gtk+.ada

I've ported LinXtris (http://linxtris.sourceforge.net) from GTKAda 2 to GTKAda 3.

[...]

What's “Outside” the Ada Inside?

From: Tero Koskinen <tero.koskinen@iki.fi>
Date: Wed, 7 Oct 2015 22:42:50 +0300
Subject: Re: what does your Ada + hardware look like?
Newsgroups: comp.lang.ada

> What sorts of hardware are you using with Ada? What sort of CPU, is it on a custom PCBs or off the shelf?

Gallery of my 8-bit AVR hardware can be found at https://www.flickr.com/photos/terokoskinen/albums/72157625466664467

All of these run AVR-Ada. [...]

> Has anyone used IC2 or SPI from a desktop to control circuits that don't have a C firmware?

Desktop PCs rarely have I2C or SPI pins exposed. You better use some microcontroller between the I2C/SPI IC and the desktop.

http://arduino.ada-language.com/tag/i2c.html should give you some idea.

For example, I once used Arduino and Ada to read my (father's) laptop's I2C EEPROM contents:

http://arduino.ada-language.com/recovering-ibm-thinkpad-t42-bios-password-with-avr-ada-and-arduino.html

Natural Language Processing Tools

From: Matteo Grella <matteogrella@gmail.com>
Date: Fri, 16 Oct 2015 05:47:00 -0700
Subject: Natural Language Processing with Ada 2012
Newsgroups: comp.lang.ada

I'm glad to inform you that at the GitHub links below, you'll find Ada 2012 used for Natural Language Processing and more in general for Artificial Intelligence/ Machine Learning.

Ada Dependency Parser (AdaDP)

AdaDP is a very basic multilingual statistical dependency parser written in pure Ada that I use to experiment some ideas. It is based on a shift-reduce transition-based system that produces dependency parse trees for natural language sentences using parsing model learned from annotated corpora.

https://github.com/matteo-grella/mg-research/tree/master/ADADP

NLPColl Deep Parser

NLPColl Deep Parser is a dependency parser written in Ada 2012 that use Deep Learning/Word Embedding algorithms. This project derives from Stanford CoreNLP tools (see https://github.com/stanfordnlp/CoreNLP).

https://github.com/MGMN/NLPColl/tree/master/DeepParser

Machine Learning

Ada 2012 implementations of Multi-layer Perceptron and Averaged Perceptron.

https://github.com/MGMN/ada-relax-component-collection/tree/master/src/arcoll/machine_learning

KDF9 Emulator

From: Bill Findlay <yaldnif.w@blueyonder.co.uk>
Date: Fri, 6 Nov 2015 03:11:50 +0000
Subject: ee9 V2.0, the GNU Ada KDF9 emulator
Newsgroups: alt.folklore.computers, comp.lang.ada

.. is now available for OSX, Windows, Linux 32 & 64 bit, and Raspberry Pi, at:

http://www.findlayw.plus.com/KDF9/#Emulator

Both GPL source code and system-specific binaries are provided, along with KDF9 sample programs and data, including the famous Whetstone benchmark.

This version is written in Ada 2005; I expect the next to be in Ada 2012.

[See also “The GNU Ada KDF9 emulator, now on Raspberry Pi”, AUJ 33-4, p. 244. —sparre]

Indirect Information on Ada Usage

[Extracts from and translations of job-ads and other postings illustrating Ada usage around the world. —sparre]

Job offer [Finland] System Engineer

[...]

In short, Atlas-Elektronik is looking for System Engineer who knows C++, Java, Ada or Python. Location is Tampere, Finland.

The last time I was chatting with them, they had pretty big amount of Ada code in their hands. I guess the other languages are there to get more applicants. I also assume that Finnish language knowledge is not needed if you know Ada well enough.

Job offer [England]

[at Finmeccanica]

Ada in Context

Returning Variable-sized Results

From: Hadrien Grasland <hadrien.grasland@gmail.com>
Date: Fri, 21 Aug 2015 01:37:22 -0700
Subject: How do typical Ada calling conventions work?
Newsgroups: comp.lang.ada

I am currently having fun re-implementing some of Numerical Recipes' code snippets in Ada. Since this is just a personal project, I do not care about scaling to very large matrices/vectors, so I chose to keep things simple by allocating arrays on the stack, rather than on the heap. But then I suddenly found myself wondering what kind of implementation magic made my code work at all.

Let me elaborate: in the C family, as far as I know, programming language rules enforce that the size of almost any variable, including function results, must be known at compile-time, with C99 VLAs as a notable exception.

This makes it easy for implementations to devise calling conventions: the caller can allocate as much space as needed for the return value in registers or on the stack frame, right before pushing parameters. Upon function return, the caller will find the result where it would expect it.

However, as far as I can tell, Ada does not offer this guarantee: if I write this code, for example...

```
type Unconstrained_Array is array
        (Positive range <>) of Integer;
function Make_Array return
        Unconstrained_Array;
```

...there is no way for a caller of Make_Array to tell, in advance, how big the function's result will be. So only the callee knows enough to allocate space for its return value.

Does someone know how typical Ada implementations manage to cope with this, and if it varies a lot from one implementation to another?

From: Markus Schöpflin
Date: Fri, 21 Aug 2015 12:41:01 +0200
Subject: Re: How do typical Ada calling conventions work?
Newsgroups: comp.lang.ada

> [...]

GNAT allocates such objects on the secondary stack. Quote from the user guide:

`-Dnn[k|m]'

This switch can be used to change the default secondary stack size value to a specified size nn, which is expressed in bytes by default, or in kilobytes when suffixed with k or in megabytes when suffixed with m.

The secondary stack is used to deal with functions that return a variable sized result, for example a function returning an unconstrained String. There are two ways in which this secondary stack is allocated.

For most targets, the secondary stack is growing on demand and is allocated as a chain of blocks in the heap. The -D option is not very relevant. It only give some control over the size of the allocated blocks (whose size is the minimum of the default secondary stack size value, and the actual size needed for the current allocation request).

For certain targets, notably VxWorks 653, the secondary stack is allocated by carving off a fixed ratio chunk of the primary task stack. The -D option is used to define the size of the environment task's secondary stack.

From: Niklas Holsti <niklas.holsti@tidorum.fi>
Date: Fri, 21 Aug 2015 15:47:04 +0300
Subject: Re: How do typical Ada calling conventions work?
Newsgroups: comp.lang.ada

> [...]

I stumbled across an interesting, old discussion about GNAT's approach (it seems a secondary stack was not the initial choice) with references to some other compilers: http://computer-programming-forum.com/44-ada/2227f74c82f45451.htm

From: Randy Brukardt <randy@rrsoftware.com>
Date: Mon, 24 Aug 2015 17:03:55 -0500
Subject: Re: How do typical Ada calling conventions work?
Newsgroups: comp.lang.ada

> I'm glad the secondary stack approach won in the end for GNAT, it sounds more efficient, clean and scalable than what they were attempting in the beginning.

I doubt that. The scheme Janus/Ada uses is much simpler: the memory is allocated off of a special storage pool and it is then freed using the same mechanism that does other finalization (indeed, the memory management [which we used in Ada 83] was repurposed to do finalization, rather than the other way around).

This is probably not as efficient as the secondary stack approach, but I find that irrelevant in 99.9% of programs. All functions that return non-elementary types are somewhat more expensive than the similar parameter passing, so if efficiency is a primary concern, one must use procedures rather than functions. The cases where function return by secondary stack would be efficient enough but function return by heap is not efficient enough are going to be quite rare [in the vast majority of cases either both are good enough or neither are] -- thus there are better things to spend effort on.

Advice from Compiler Writers?

From: Luke A. Guest <laguest@archeia.com>
Date: Sun, 23 Aug 2015 11:16:00 -0700
Subject: Requesting advice from compiler writers
Newsgroups: comp.lang.ada

I'm one of the people considering a new Ada compiler, I would like to ask the people here who have actually done it, what their advice would be. i.e. what to aim for, what to avoid doing, etc?

I personally am aiming for a library based multi-backend system. Basically, I'd like to see a full Ada compiler frontend with an Ada backend, but with the ability to poach other backends, specifically LLVM. My initial targets are unusual, Z80, MIPS and x86 (16-bit -> 64-bit) and JVM bytecode.

From: Brian Drummond <brian@shapes.demon.co.uk>
Date: Mon, 24 Aug 2015 11:50:21 +0000
Subject: Re: Requesting advice from compiler writers
Newsgroups: comp.lang.ada

> [...]

I've mentioned before: I'd suggest looking at Gela-Asis as the front end. Natasha forked it to her Github last year, (https://github.com/faelys/gela-asis) and (around the same time) I saw some activity on the original repo (which I can't currently see at http://gela.ada-ru.org/gela_asis) to bring it up to date with Ada-2012.

Then I would take a good look at the GHDL project for several reasons: https://sourceforge.net/projects/ghdl-updates/?source=navbar

- It is the only example I know of, of a complex language compiler successfully developed by a single individual. (Tristan Gingold, who now works for Adacore. Before deciding to use anything from GHDL, it would be worth asking him if he's comfortable that there's no conflict of interest.)
- It supports VHDL and therefore has quite a bit in common with Ada, including explicit parallelism (though a different model to Ada tasking)
- It is written entirely in Ada up to the intermediate representation, an intermediate language called "ortho" (which I believe is Tristan's own).
- It has one back-end also written in Ada - this is a JIT compiler, from "ortho" currently limited to 32-bit x86, and actually performs quite well.
- It also has backends from ortho to both gcc and LLVM, which between them cover a lot of targets, but are clearly not in Ada. (And some debugging facilities to save ortho, replay it through a backend, and to print/examine it.)

I had considered writing a layer between Gela-Asis and ortho, using the GHDL compiler as an example of how to generate ortho. And I'm still considering it...

From: Brian Drummond <brian@shapes.demon.co.uk>
Date: Mon, 24 Aug 2015 12:02:57 +0000
Subject: Re: Requesting advice from compiler writers
Newsgroups: comp.lang.ada

[gela-asis]

see https://www.openhub.net/p/11234

This seems to be active.

A Limitation of Ada.Text_IO

From: Randy Brukardt <randy@rrsoftware.com>
Date: Mon, 24 Aug 2015 17:22:38 -0500
Subject: Re: Creating an empty file with Ada.Text_IO
Newsgroups: comp.lang.ada

[...]

More generally, Text_IO is for processing files in an implementation-determined format. (And that's true to some extent for Sequential_IO and Direct_IO as well.) If you actually care about the bits in the file (as you might when writing a file for some other program), only Stream_IO gives the needed control.

For example, the Janus/Ada compiler only uses our in-house Basic_IO package for file I/O (we invented Basic_IO for Ada 83; if we were starting today we'd use Stream_IO which has essentially the same capabilities). The standard I/O packages just didn't work for our needs, even though we implemented them ourselves. They had too much overhead and too little control.

SPARK: Case Coverage Rules

From: Maciej Sobczak <maciej@msobczak.com>
Date: Fri, 9 Oct 2015 04:38:32 -0700
Subject: SPARK: missing case value
Newsgroups: comp.lang.ada

Consider:

```
type Enum is (A, B, C);
procedure Test (E : in Enum)
  with Pre => E /= C
is
begin
  case E is
    when A => null;
    when B => null;
  end case;
end Test;
```

The Pre contract says that C is never used as a value for E. Still, GNATProve complains about missing case value C in the case statement. The compiler complains, too.

An appropriate subtype (subtype SEnum is Enum range A .. B) can solve this, but it shows some asymmetry between subtypes and contracts, where I would expect that the same subsetting effect can be achieved in both ways. Apparently (and according to AARM), the case statement does not take contracts into account, but my understanding of the rules is that it would not be against the spirit of "covering values that satisfy the predicate".

On the other hand, SPARK is supposed to be a subset of Ada, so even if the above is feasible from the SPARK point of view, it should compile as regular Ada as well and compilers are not required to do this level of static analysis. So, SPARK does not do it, because Ada might not be able to keep the pace.

What are your thoughts on this?

From: Mark Lorenzen <mark.lorenzen@gmail.com>
Date: Fri, 9 Oct 2015 05:35:03 -0700
Subject: Re: SPARK: missing case value
Newsgroups: comp.lang.ada

> [...]

I think it is logical and correct. How would a compiler be able to determine the range of E if your precondition was more complex?

I would change the case statement into something like this:

```
case E is
  when A => null;
  when B => null;
  when C => raise Impossible;
  -- or maybe Pragma Assert (False)
end case;
```

Note that you can use raise statements in SPARK as long as the program is still in SPARK i.e. the raise statement will never be executed.

From: Bob Duff <bobduff@theworld.com>
Date: Fri, 09 Oct 2015 10:39:01 -0400
Subject: Re: SPARK: missing case value
Newsgroups: comp.lang.ada

> [...]

That's illegal Ada, as you noted. And illegal SPARK.

But this works:

```
type Enum is (A, B, C);
subtype A_C is Enum with Predicate =>
        A_C /= B;

procedure Test (E : in A_C) is
begin
  case E is
    when A => null;
    -- "when B" is not needed.
    when C => null;
  end case;
end Test;
```

And that has the advantage that A_C need not be a subrange; it can have holes.

I find that predicates are often better than preconditions, because the same precondition often applies to many parameters, and also to local variables. The predicate allows you to avoid repetition.

("Predicate =>" is a GNAT-specific extension. In Ada, you need to say "Static_Predicate =>". IMHO the "Static_" part is just noise, but I couldn't convince the rest of ARG.)

Futures/Chained Function Calls?

From: Alejandro R. Mosteo <alejandro@mosteo.com>
Date: Wed, 14 Oct 2015 16:30:08 +0200
Subject: Musings on RxAda
Newsgroups: comp.lang.ada

Having been recently working on Java/Android apps, I have been exposed to the ReactiveX framework [1]. I think it originated on C# though.

Anyway, the gist of it is that you chain function calls (operators in their jargon) to process data asynchronously. This, in Java, with lambda functions and parameter inference leads to extremely compact, legible code that may notably simplify multithreading code.

[...]

For simplicity, I'm going to concentrate on the "map" operator, as seen on this Java example:

```
Observable.just("Hello, world!")
   .map(s -> s + " -Dan")
   .map(s -> s.hashCode())
   .map(i -> Integer.toString(i))
   .subscribe(s -> System.out.println(s));
```

Basically, the map operation takes some input, changes it somehow and outputs another type. The chain begins at an Observable (some data generator) and ends at a subscriber (which does something with the result). The above example takes a string, appends a signature, hashes it, and outputs the hash as a string. This does not necessarily has to happen at the moment of declaration; in general, observables emit data asynchronously.

Moving into Ada, we need an Observable type able to take different map implementations. That could be (not compiled, bear with my mistakes. Allow for 2012 syntax):

```
type Observable is [limited?] tagged
      record;
type Datum is abstract interface;
-- or maybe tagged null record;

function Map (This : in out Observable;
       Map  : access function (X :
       Datum'Class) return Datum'Class)
  return Observable;
-- or access Observable if limited
```

Here the 'Class are needed to avoid multiple dispatch, I think. Then you'll have to declare beforehand any mappings you need (which is more verbose and, lacking lambdas, would somehow defeat the purpose of having the logic in the chain declaration, but I see no way around it, unless the new simple expression functions can appear in-place?). Also, the need for 'Class parameters will force explicit casts in the user mappings, which I find ugly and leaves the type checks to runtime.

I guess an implementation of "RxAda" could provide many Map profile overloads for basic Ada types (from String to String and so on), but still that's a poor's man attempt.

That's as far as I've got, which is not much, but I'm a bit rusty on Ada right now. I can think of even more contrived ways using tagged types plus generics but I'm not sure they lead anywhere. I think it boils down to Ada not having lambda functions nor implicit template types a-la C++.

[1] http://reactivex.io/

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Wed, 21 Oct 2015 18:47:35 +0200
Subject: Re: Musings on RxAda
Newsgroups: comp.lang.ada

[...]

Pipeline is just a method of buffered exchange. As such it is blocking (publisher) when the subscriber is unable to process all published data.

Moreover it is unsuitable for 1-n communications. If you are OK with blocking the publisher, then no buffering is ever needed. Buffering (marshaling) is required by non-blocking but not ensures it.

And no mediator task is needed anyway, not even a protected object. 1-1 FIFO requires none of Ada's synchronization primitives.

From: Hadrien Grasland <hadrien.grasland@gmail.com>
Date: Wed, 21 Oct 2015 12:09:58 -0700
Subject: Re: Musings on RxAda
Newsgroups: comp.lang.ada

So you would essentially store a pipeline of data and operations in some container, then perform all of them on the same thread when the output of the pipeline is requested?

This seems much more complex to implement to me, since your operation queue needs to be able to store data of any type and operation function pointers, all in a type-safe way. As far as I can tell, you cannot use streams for that because a stream requires you to know what you are reading from it.

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Wed, 21 Oct 2015 21:35:17 +0200
Subject: Re: Musings on RxAda
Newsgroups: comp.lang.ada

> So you would essentially store a pipeline of data and operations in some

> container, [...]

I don't know your requirements. In a FIFO scenario, yes, the publisher pushes the object. The subscriber pulls it out. In a blackboard scenario (non-blocking) the publisher pushes the object and the subscribers scan the blackboard for updates.

Of course in a modern typed language like Ada you would not mess with "data and operations." There are objects for that. The type of the object determines the operations.

> [...]

That is because you are trying to think about it in terms of C. Ada is not C, luckily. It is quite straightforward to marshal T'Class objects with a dispatching operation Do_It. And yes, it is as much easy to marshal handles to reference-counted objects if objects are expected large (and you have shared memory).

From: Jacob Sparre Andersen <jacob@jacob-sparre.dk>
Date: Thu, 19 Nov 2015 14:14:56 +0100
Subject: Re: Musings on RxAda
Newsgroups: comp.lang.ada

> Anyway, the gist of it is that you chain function calls (operators in their jargon) to process data asynchronously.

This sounds like how you chain filters in a Unix shell using pipes.

Continuing that thought, an obvious mapping to Ada is tasks (acting as filters) and protected objects (acting as pipes/buffers).

Even though Ada doesn't explicitly have lambda functions, localised scopes allow similar (if somewhat more verbose) formulations.

> Observable.just("Hello, world!")

>   .map(s -> s + " -Dan")

>   .map(s -> s.hashCode())

>   .map(i -> Integer.toString(i))

>   .subscribe(s ->System.out.println(s));

```
Q1 : String_Queues.Queue;

task Hello_World;
task body Hello_World is
begin
   Q1.Enqueue (+"Hello, world!");
end Hello_World;

Q2 : String_Queues.Queue;

task Add_Source;
task body Add_Source is
   Buffer : Unbounded_String;
begin
   Q1.Dequeue (Buffer);
   Q2.Enqueue (Buffer & " -Dan");
end Add_Source;

Q3 : Hash_Queues.Queue;

task Hash_String;
task body Hash_String is
   Buffer : Unbounded_String;
begin
   Q2.Dequeue (Buffer);
   Q3.Enqueue (Hash (Buffer));
end Hash_String;

Q4 : String_Queues.Queue;

task Hash_As_String;
task body Hash_As_String is
   Buffer : Hash_Type;
begin
   Q3.Dequeue (Buffer);
   Q4.Enqueue (+Hash_Type'Image
(Buffer));
end Hash_As_String;

task Output;
task body Output is
   Buffer : Unbounded_String;
begin
   Q4.Dequeue (Buffer);
   Put_Line (+Buffer);
end Output;
```

See <https://bitbucket.org/sparre/ada-2012-examples> ("src/chained_calls.adb") for the full, compilable version.

I'm tempted to make the tasks task types with references to the input and output queues as discriminants, but I'm not sure it really would be an improvement.

Interfaces

From: Eryndlia Mavourneen <eryndlia@gmail.com>
Date: Fri, 30 Oct 2015 12:48:05 -0700
Subject: Re: Task interface and entries with aliased parameters
Newsgroups: comp.lang.ada

Randy Brukardt wrote:

> Sharing interfaces *sounds* nice, but it's impossible in practice ...

I currently am building a design using a family of task interfaces. I find this is a nice way to have related tasks working together, some entries' being the same among the tasks, and some entries' being specific to certain tasks.

Yes, there are other ways to manage this, but I find the family of tasks a clean, easy-to-understand concept; plus, it's fun.

It also is important to remember that in many ways we are limited by our tools: The better abstractions that are available embodied in our tools the better our tools can help us to develop innovative ideas based on those tools.

Iterating over Dates, Primes, Text Files, etc.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Tue, 3 Nov 2015 00:40:37 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

I suspect you could have written this in a lot simpler fashion. Something similar to the Prime_Numbers iterator found in the ACATS foundation (F552A00, look in the support subdirectory, specifically http://www.ada-auth.org/cgi-bin/cvsweb.cgi/acats/support/f552a00.a) would be a starting point. (Thanks again to Brad Moore for creating this foundation and the associated ACATS tests to give all Ada implementers something to use as a correct example of iterators.)

From: Simon Wright <simon@pushface.org>
Date: Wed, 04 Nov 2015 16:19:41 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

And I was hoping to write

```
with Ada.Text_IO;
with F552A00_Prime_Numbers;
procedure Primes is
begin
  for P of F552A00_Prime_Numbers.
    Prime_Number_Set'(Max_Value => 31)
  loop
    Ada.Text_IO.Put_Line (P'Img);
  end loop;
end Primes;
```

but (surprise)

```
primes.adb:5:08: cannot iterate over
"Prime_Number_Set"
```

From: Randy Brukardt <randy@rrsoftware.com>
Date: Wed, 4 Nov 2015 19:20:42 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

Prime_Number_Set is (directly) an iterator, so you use "in" to iterate over it:

```
for P in F552A00_Prime_Numbers.
    Prime_Number_Set'(Max_Value => 31)
loop
```

The "of" form is for iterating over an array or a container whose type has the Default_Iterator aspect (which gives the iterator to use).

I can hear the head slap from here. ;-)

From: Emmanuel Briot <briot@adacore.com>
Date: Thu, 5 Nov 2015 00:34:34 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> The "of" form is for iterating over an array or a container whose type has the Default_Iterator aspect (which gives the iterator to use).

The main drawback, though, is that you then get a Cursor, not an Element, so you still need to call the functions Element or Reference to get to the actual element. This is slightly less convenient (syntax-wise). I wish it was possible to use "of" to indicate that we want to get an element, even when the right-side is an iterator (which for instance would be convenient when writing graph data structures where there really are lots of different ways to iterate, and a single Default_Iterator is not enough.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 12:28:46 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...] The main drawback, though, is that you then get a Cursor, not an Element, [...]

Sure, but not in a case like the OP's, where the "cursor" is the actual data and there is no element. In that case, "of" is just overkill.

> I wish it was possible to use "of" to indicate that we want to get an element, [...]

That doesn't make sense, though, as the iterator interface doesn't contain the information necessary to do "of" iteration. In particular, it doesn't provide the container or Reference function -- nor could it, as there may not be a container.

Personally, I find the "of" form unnecessary; we've iterated arrays for decades without direct access to the element (using the "cursor", that is the array index), so why are containers different? Especially as the indexing form works on all of the language-defined containers (you never need to explicitly call Reference or Element). So an "in" iterator looks just like the array iteration that we've been using from the beginning of time. What's so hard about that?

From: Simon Wright <simon@pushface.org>
Date: Thu, 12 Nov 2015 20:19:47 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

There's one problem for Times; how to determine when to end the loop if the cursor is the actual data?

I ended up with

```
type Cursor (Valid : Boolean := True)
        is record
   case Valid is
      when True =>
         Date : Ada.Calendar.Time;
      when False =>
         null;
   end case;
end record;
```

where both First and Next set Valid to False if the iteration has ended.

I spent a lot too much time without giving Valid a default value; there is an assignment, but it's hidden inside the generated code, so at the end of the loop the discriminant changes ...

I think this may be related to the way that F552A00_Prime_Numbers doesn't support an upper bound of 2; the loop terminates when Is_Prime returns false, and of course 2 + 1 = 3 which .. Is_Prime.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 15:15:45 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

[...]

Iterating on elements of a container is unnecessary overkill. (It's also harmless overkill, unlike, say, anonymous access types).

> [...]

There is always *something* that works as an index. If there isn't, you can't iterate (because you can't figure out a reproducible order for which item is next). In any case, Ada does not support iteration without something being a cursor; the "of" form of iteration is a direct translation of the "in" form of iteration (it's purely syntactic with no semantics of its own).

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Sat, 14 Nov 2015 12:42:28 +0100
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

It would be nice to see a demonstration for iterating lines of a text file... (:-))

[...]

From: Simon Wright <simon@pushface.org>
Date: Sat, 14 Nov 2015 12:37:33 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

It's a pity that Next takes the iterator object as an 'in' parameter. This means that, in general, the cursor needs to hold the state. It certainly needs some way of indicating end-of-iteration; perhaps a sentinel value.

Can I use the same iterator more than once? (So long as the container hasn't changed, of course. - I expect that would count as tampering, though.) If so, that'd explain why Next.Object is an in parameter!

From: Brad Moore <bmoore.ada@gmail.com>
Date: Sun, 15 Nov 2015 10:54:20 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...] Next takes the iterator object as an 'in' parameter. [...]

I found this to be a problem also. I have been working on an AI to potentially add iterators to Ada.Directories and Ada.Environment_Variables for consideration for Ada 202x (Ada 2019?). I was able to get a working implementation using both an Iterator approach, and an Iterable container approach. So far the Iterable container approach seems like a better choice here because it lets the programmer choose between either "in" or "of" syntax for the loops. With the Iterator approach, you can only use "in" loops, although that's really not a big deal, in my opinion.

For both of these approaches, I found I also needed "in out" parameters for these interfaces. In fact, looking at my iterable container implementation it appears that First, Next, and Iterate all need to have 'in out' parameters. For Ada.Directories, it made sense to have the container contain a Search_Type component, and the Iterator type to contain a Directory_Entry component, which contains the current directory entry for the loop. The Next function needs to update its Directory_Entry component to get the next directory entry, so it needs read-write access.

I was able to work around this by using the Rosen trick to acquire read-write access to the "in" parameters for these functions. However, this is awkward, and it tells me that we have the wrong interfaces for iterators, or at least we have missing interfaces. One shouldn't have to jump through such hoops to write iterators for abstract data types.

From: Emmanuel Briot <briot@adacore.com>
Date: Fri, 13 Nov 2015 00:45:46 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> That doesn't make sense, though, as the iterator interface doesn't contain the information necessary to do "of" iteration. In particular, it doesn't provide the container or Reference function -- nor could it, as there may not be a container.

Absolutely, we cannot do that with Ada, as defined in the standard. But that's exactly what I am complaining about. The aspects should have been defined on the Iterator, not on the container itself, as in:

```
   type Graph is tagged private
      with Default_Iterator =>
Depth_First_Search;
   type DFS_Iterator is private
      with Reference => Reference,
         Iterator_Element => Element_Type;
   function Depth_First_Search (Self:Graph)
return DFS_Iterator;

   type BFS_Iterator is private
      with Reference => Reference,
         Iterator_Element => Element_Type;
   function Breadth_First_Search (Self :
          Graph) return BFS_Iterator;
```

With such a separation, I could then do something like:

```
   G : Graph;

   for E of G loop   -- depth first search
      null;
   end loop;

   for E of G.Breadth_First_Search loop
   -- breadth first search, get element
      null;
   end loop;

   for C in G.Breadth_First_Search loop
   -- breadth first search, get cursor
      null;
   end loop;
```

The aspects were put at the wrong level, and iterators end up being hidden magical objects, rather than first class citizens with their own role in the design of containers.

> Personally, I find the "of" form unnecessary; [...]

There are lots of things we have been doing for decades. We have been inserting explicit tests and asserts, and now we are adding pre and post conditions, just to find one example. Languages evolve, as you well know of course, and the goal is to make them more expressive. I like to clearly state what I mean in the code, but on the other hand why should I have to specify "Element (C)" when I can just use "E" ?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Fri, 13 Nov 2015 11:41:02 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

> With such a separation, I could then do something like:

No, you couldn't, because there still is no way for the compiler to identify the container to pass to the Reference aspect.

If you mean to magically extract it from the call to Depth_First_Search, I would be strongly against as there is no need for an actual container object for an iterator (and the model specifically was designed to allow that possibility). It would be *much* more limiting to assume a container.

I suppose you could add a third aspect to specify which parameter is the container, and then use additional magic to add a renames of that parameter (with all of the attendant restrictions on what that parameter could be). Definitely a lot more complicated than what we have (and what we have is too complicated).

```
>    for E of G.Breadth_First_Search loop
   --  breadth first search, get
> element
>        null;
>    end loop;
```

Again, how does the compiler know that G is the container in this iterator? "G.Breadth_First_Search" is just a function call. On top of which, you now have two different and unrelated expected types for the iterator, which would be a new concept in Ada.

> The aspects were put at the wrong level, [...]

Well, I disagree. The iterators are certainly a first-class type, so long as you use the "in" form. And it's impossible to use the "of" form for all possible iterations; it's way too limiting for the general case.

> [...]

Umm, no, the goal is to provide new capabilities to allow Ada to solve additional problems. (Pre and Post are definitely new capabilities - see the class-wide versions and 'Old; pragma Assert was too much of a blunt instrument to be of much use.)

Making a language "more expressive" is code for "easier to write", which never was a goal of Ada. To the extent that we've made the language "more expressive", it's been to make ADTs more like the capabilities of the built-in types (for instance, arrays). Going beyond that is unnecessary and probably harmful.

> I like to clearly state what I mean in the code, but on the other hand why should I have to specify "Element (C)" when I can just use "E"?

Readability, of course. I don't believe that you should ever have been allowed to just specify E. [...]

From: Emmanuel Briot <briot@adacore.com>
Date: Sat, 14 Nov 2015 11:57:07 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

You seem to be mixing the particular implementation you would use with what the standard should allow. There is nothing in the spec of a reference type that mandates knowing the container (and the way it currently is done in the implementations is to use a 'Unrestricted_Access, which is ugly to say the least).

> [...]

If I want my iterator to implement depth_first_search, it would indeed need to know the container (like most iterators would need to know the container, in fact). The reference type itself (the end product of the iteration) doesn't need to know that though. The container is necessary for the iteration or traversal of itself, but certainly not once you manipulate an element.

My example assumed nothing.

The implementation you seem to have in your head seems indeed to assume a lot of unnecessary things.

> [...]

What we have is both too complicated and too limited. Too complicated because nobody that hasn't spent a full week will be able to explain how the aspects combine with iterators to end up with cursors and the for-of loop. They just happen to work for existing containers, but as this thread has proven, anyone who tries to actually implement them for other uses that the few official containers quickly hits severe limitations.

> Again, how does the compiler know that G is the container in this iterator? [...]

It is a function call that returns the iterator. So obviously it is allowed to have a relationship between the return value and the parameters. The iterator is therefore allowed (but not mandated, that would depend on the actual container we are talking about) to have a reference to the container. This is called flexibility. Something lacking in the current standard.

I do not understand your last sentence about the two unexpected types.

> Making a language "more expressive" is code for "easier to write", which never was a goal of Ada.

That's one of the issue here. The goal for Ada, as I understand it, has always been to be "easier to read". There is no reason why "easier to write" should be in opposition to that goal.

[...]

From: Simon Wright <simon@pushface.org>
Date: Mon, 16 Nov 2015 21:50:26 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> Since 'Unrestricted_Access is GNAT junk, it's doesn't have any value for a sample implementation [...]

I can't of course speak for AdaCore, but I think that the problem is to do with obtaining a writable view of an in parameter; for example, in the Bounded Hashed Map (GCC 5) there is

```
function Iterate (Container : Map)
  return Map_Iterator_Interfaces.
         Forward_Iterator'Class
is
   B  : Natural renames
    Container'Unrestricted_Access.all.Busy;
begin
   return It : constant Iterator :=
    (Limited_Controlled with
      Container =>
          Container'Unrestricted_Access)
   do
     B := B + 1;
   end return;
end Iterate;
```

I dare say something could be done with 'Address and Address To Access Conversions.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Tue, 17 Nov 2015 15:33:39 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...] I dare say something could be done with 'Address and Address To Access Conversions.

As Brad noted, you have to use the Rosen technique here. Not hacks like 'Unrestricted_Access. (The Rosen technique is not a "trick", it's an intended part of the language in Ada 2012. I admit it originally was accidental, but it's too widely used to eliminate.)

From: Simon Wright <simon@pushface.org>
Date: Tue, 17 Nov 2015 23:14:17 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...] the Rosen technique [...]

Does that now work for non-limited types? The anti-tampering bits have to be in the container ... how have other vendors managed this?

Brad was talking about using this technique in the iterator.

From: Emmanuel Briot <briot@adacore.com>
Date: Tue, 17 Nov 2015 00:49:43 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]

I am trying to think of things should have been done, so that perhaps we can find a way to fix them in future versions of the language. As this discussion as shown, there are lots of very knowledgeable people who are having difficulties with the current design and its limitations. If you think this is wasting your time, by all means feel free to ignore this thread. I have found it pretty interesting so far.

> [...]

I was talking of the Reference_Type (which doesn't need a container in the public spec, although it is convenient sometimes to have one to ensure the lifetime of the container is greater than that of the reference -- with all the additional performance costs). The Iterator interface is an interface, so of course it doesn't contain anything. An actual Iterator implementation is free to store a reference to the container if it needs it to implement complex algorithms. An iterator could return a reference by building it directly, it doesn't need to go through the Reference function for this.

[...]

> I'm mapping iteration onto the existing containers. That is straightforward with the existing description,

No it is not. Anyone who has already tried to implement their own iterators has failed the first time (at AdaCore, but also Simon Wright in the initial message here, at courses we have given in various places, ...)

[...]

From: Simon Wright <simon@pushface.org>
Date: Thu, 05 Nov 2015 08:45:13 +0000
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...] Prime_Number_Set is (directly) an iterator, so you use "in" to iterate over it: [...]

Yes, but;

```
 1. with Ada.Text_IO;
 2. with F552A00_Prime_Numbers;
 3. procedure Primes is
 4. begin
 5.    for P in F552A00_Prime_Numbers.
       Prime_Number_Set'(
          Max_Value => 31) loop
     >>> expected a discrete type
     >>> found type "Prime_Number_Set"
 defined at f552a00_prime_numbers.ads:74

 6.      Ada.Text_IO.Put_Line (P'Img);
 7.    end loop;
 8. end Primes;
```

Perhaps this is the GNAT problem you spoke of? (FSF GCC 5.1.0, GNAT GPL 2015)

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 12:32:09 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> Oh, it should have been
>
> for P in F552A00_Prime_Numbers.Prime_Number_Set'(Max_Value => 31).Iterate loop

I probably would have made type Prime_Number_Set directly the iterator type (I don't see any other need for it), which would get rid of the need for the ".Iterate". An iterator doesn't necessarily have to have anything to do with a container!

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 15:08:52 -0600
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]
>
> A Prime_Number_Set is a new Prime_Number_Iterator.Forward_Iterator; Iterate returns a Prime_Number_Iterator.Forward_Iterator'Class.
>
> I tried without the .Iterate, as before, and with
>
> for P in Prime_Number_Set'Class (Prime_Number_Set'(Max_Value => 30))
>
> (just in case) and as before got the 'expected a discrete type' error.

It should work either way. But does it work if you write:

```
for P in Prime_Number_Iterator.
        Forward_Iterator'Class
    (Prime_Number_Set'(Max_Value => 30))
```

?? If so, GNAT surely has a bug (the type conversion shouldn't change anything). You could also try:

```
for P in Prime_Number_Iterator.
        Forward_Iterator'
    (Prime_Number_Set'(Max_Value => 30))
```

which also shouldn't change anything.

> Perhaps this is a GNAT bug?

Looks like it to me.

> On the other hand, why did Brad include function Iterate?

It appears that he wanted tracing of the initialization (as that is all it does other than making a new iterator that is a copy of the first). It shouldn't be necessary.

From: Brad Moore <bmoore.ada@gmail.com>
Date: Sun, 15 Nov 2015 09:56:45 -0800
Subject: Re: A few questions
Newsgroups: comp.lang.ada

> [...]
>
> for P in Prime_Number_Iterator.Forward_Iterator'(Prime_Number_Set'(Max_Value => 30)) loop

This doesn't work also.

However, if I declare the iterator as a declared object as in;

```
Iterator : constant Prime_Number_Iterator.
        Forward_Iterator'Class :=
    Prime_Number_Set'(Max_Value => 30);
```

Then I can do this;

```
for Prime in Iterator loop
   Put_Line (Integer'Image (Prime));
end loop;
```

However, if I declare it this way...

```
Iterator : constant Prime_Number_Set :=
Prime_Number_Set'(Max_Value => 30);
```

Then the loop above does not compile.

So it seems to me that there are some GNAT bugs in this area.

> > On the other hand, why did Brad include function Iterate?
>
> [...]

It's been a while since I wrote that example, and I'm not entirely sure why I did it that way, but to be honest, I think I started with the existing container iterators as my starting point example, which all have an Iterate function, and I may have just assumed I needed a function which returned an class-wide object.

If the Prime Number Iterator is useful as an example for others, it would be nice to eliminate the Iterate function from the example. It seems also that there should be some more tests written to cover the failure cases that we are discovering now. I'd be happy to update this ACATS test, if Randy thinks it's worthwhile.

Discriminants with Default Values - Limited Types

From: Bob Duff <bobduff@theworld.com>
Date: Tue, 10 Nov 2015 19:48:40 -0500
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

[...]

Well, I think the Ada.Strings.Bounded package is way overengineered. So "using Ada.Strings.Bounded" = "using bounded strings wrong". ;-)

I suggest rolling your own. No need for generics.

```
type Bounded_String (Max_Length :
        Natural := ...) is limited record
   Length : Natural := 0;
   Chars  : String (1 .. Max_Length);
end record;
```

along with a few trivial operations.

From: Jeffrey R. Carter <jrcarter@acm.org>
Date: Tue, 10 Nov 2015 19:01:38 -0700
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

> [...]

Except that using Natural for Max_Length results in GNAT allocating Integer'Last characters for Chars, which is unlikely to fit on the stack.

From: Bob Duff <bobduff@theworld.com>
Date: Wed, 11 Nov 2015 10:34:53 -0500
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

> [...]

No, the type is limited, so GNAT will allocate space for Max_Length characters, not Natural'Last characters.

From: Jeffrey R. Carter <jrcarter@acm.org>
Date: Wed, 11 Nov 2015 10:36:06 -0700
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

> [...]

I missed the "limited" and concentrated on the default for the discriminant. It does mean that assignment becomes a bit messy, though.

From: Bob Duff <bobduff@theworld.com>
Date: Wed, 11 Nov 2015 14:22:44 -0500
Subject: Re: Bounded String question
Newsgroups: comp.lang.ada

> I missed the "limited" and concentrated on the default for the discriminant.

Right, for limited types, the default for a discriminant is just a default, without the weird magical properties you get when its nonlimited.

> ...It does mean that assignment becomes a bit messy, though.

Yes, but I find that I hardly ever need assignment of these things. They are used to build up a Bounded_String piece by piece, and then do something with it (convert to String, convert to some other type (like Name_Id, if you're familiar with compiler sources), process in place...).

Comparing Aspects and Pragmas

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 13:24:26 -0600
Subject: Re: GNAT and user-defined aspects and pragmas?
Newsgroups: comp.lang.ada

[...] Nothing that is an aspect ever should have been a pragma, and aspects should be considered a replacement for pragmas rather than some sort of parallel syntax.

Ignoring an aspect is illegal as it is considered to be a critical part of the semantics of the program. Note that this follows the model for attributes and representation clauses. Ignoring unknown pragmas is a mistake in many instances (as Robert Dewar pointed out on several occasions). He gave examples like a pragma that sets up interrupts for some processor. That program might compile on some other implementation, but it won't work.

It's better to comment out unknown pragmas and aspects rather than to ignore them. Ada 83 got this wrong for pragmas (although one can find counter-examples where it's helpful).

> [...] wouldn't be able to compile SPARK 2014 programs with anything but GNAT - even when some of AdaCore's competitors release an Ada 2012 compiler.

Definitely. Since the SPARK aspects include some additional syntax, I can't see any way that it would be possible for a compiler that doesn't know about SPARK (say Janus/Ada) to compile a SPARK program. To the extent that SPARK annotations aren't significant to the semantics of the program, they probably should have been pragmas. But since AdaCore doesn't understand/believe in the difference, you'll have to remove them. That should be easy with a tool (it's a simple syntax transformation). [Of course, someone has to write the tool.]

From: Edward R. Fish <onewingedshark@gmail.com>
Date: Thu, 12 Nov 2015 12:37:37 -0800
Subject: Re: GNAT and user-defined aspects and pragmas?
Newsgroups: comp.lang.ada

> [...]

About the only instance where should-have-been-aspects-pragmas make any sense is with Import and Export. Things like, say, OpenGL with its half-million (slight exaggeration) definitions of Color, would be done cleaner with the single Pragma Import than individual aspects.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 12 Nov 2015 15:42:12 -0600
Subject: Re: GNAT and user-defined aspects and pragmas?
Newsgroups: comp.lang.ada

> [...] single Pragma Import than individual aspects.

Using overloading to cut down the number of pragmas one writes is precisely the kind of evil that aspects avoid. The problem is that usually you want all but one Color to be Imported, that one has its own body to provide some Ada benefit (perhaps raising an exception, or using default parameters). And of course you have to search all of the source for a pragma, while an aspect is right on the declaration (since it changes the semantics of the declaration, it should be included as part of the declaration). [Overloaded routines also often need different link names for the different bodies, and that can't be done with a pragma, at least without jumping through hoops.]

In any case, Ada is not about decreasing the amount of typing one needs to do, it's about clearly expressing your intentions. Separating aspects from declarations does not help in such expressions.

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Fri, 13 Nov 2015 11:03:55 +0100
Subject: Re: GNAT and user-defined aspects and pragmas?
Newsgroups: comp.lang.ada

> [...] [Of course, someone has to write the tool.]

Count me in!

(i.e.: I am preparing a new release of AdaSubst; adding that feature would be really easy).

Conference Calendar

Dirk Craeynest
KU Leuven. Email: Dirk.Craeynest@cs.kuleuven.be

This is a list of European and large, worldwide events that may be of interest to the Ada community. Further information on items marked ♦ is available in the Forthcoming Events section of the Journal. Items in larger font denote events with specific Ada focus. Items marked with ☺ denote events with close relation to Ada.

The information in this section is extracted from the on-line *Conferences and events for the international Ada community* at: http://www.cs.kuleuven.be/~dirk/ada-belgium/events/list.html on the Ada-Belgium Web site. These pages contain full announcements, calls for papers, calls for participation, programs, URLs, etc. and are updated regularly.

2016

♦ January 30 — **Ada Developer Room at FOSDEM 2016 Brussels**, Belgium. FOSDEM 2016 is a two-day event (Sat 30 - Sun 31 Jan). This years' edition includes once more a full-day Ada Developer Room, organized by Ada-Belgium in cooperation with Ada-Europe, which will be held on Saturday 30 January.

February 17-19 — 24th **Euromicro International Conference on Parallel, Distributed and Network-Based Processing** (PDP'2016), Heraklion, Crete, Greece. Topics include: embedded parallel and distributed systems; multi- and many-core systems; programming languages and environments; runtime support systems; performance prediction and analysis; shared-memory and message-passing systems; dependability and survivability; real-time distributed applications; formal approaches to parallel and distributed systems; security in parallel, distributed and network-based computing; multi-core and many-core systems for embedded computing; etc.

February 19-21 — 4th **International Conference on Model-Driven Engineering and Software Development** (MODELSWARD'2016), Rome, Italy. Topics include: domain-specific modeling, general-purpose modeling languages and standards, syntax and semantics of modeling languages, model-based testing and validation, model execution and simulation, model quality assurance techniques, component-based software engineering, software factories and software product lines, etc.

March 14-17 — 15th **International Conference on Modularity** (Modularity'2016), Málaga, Spain. Topics include: new modularity mechanisms in programming, modeling, and domain-specific languages; evaluation of modularity mechanisms in case studies; role of modularity in the evolution of software systems; modular re-engineering of legacy code; module (feature) interactions; novel module verification and testing techniques; modularity supported by tools, such as view extraction, visualization, recommendation, and refactoring tools; etc. Deadline for submissions: January 11, 2016 (demos, posters), January 15, 2016 (workshop papers), January 19, 2016 (Student Research Competition).

March 14-18 — 23rd IEEE **International Conference on Software Analysis, Evolution, and Reengineering** (SANER'2016), Osaka, Japan. Topics include: program comprehension, software evolution analysis, software architecture recovery, program analysis, program transformation and refactoring, mining software repositories, software analytics, parsing and fact extraction, software reverse engineering and reengineering, language support for software maintenance and evolution, experience reports, education, tools and methods, etc.

March 17-18 — 25th **International Conference on Compiler Construction** (CC'2016), Barcelona, Spain. Topics include: work on processing programs in the most general sense, such as, compilation and interpretation techniques, run-time techniques (memory management, virtual machines, ...), programming tools (refactoring editors, checkers, verifiers, compilers, debuggers, and profilers), techniques for specific domains (secure, parallel, distributed, embedded, ... environments), design and implementation of novel language constructs and programming models.

April 02-08 — 19th **European Joint Conferences on Theory and Practice of Software** (ETAPS'2016), Eindhoven, the Netherlands. Events include: ESOP (European Symposium on Programming), FASE (Fundamental Approaches to Software Engineering), FOSSACS (Foundations of Software Science and Computation

Structures), POST (Principles of Security and Trust), TACAS (Tools and Algorithms for the Construction and Analysis of Systems).

April 03 13th **International Workshop on Formal Engineering approaches to Software Components and Architectures** (FESCA'2016). Topics include: (semi-)formal techniques and their application that aid analysis, design and implementation of software applications; formal modelling of component-based, timed and hybrid systems; temporal properties and their formal verification; interface compliance and contractual use of components; static and dynamic analysis; industrial case studies and experience reports; etc.

April 04-08 31st ACM **Symposium on Applied Computing** (SAC'2016), Pisa, Italy.

April 03-08 **Track on Software Verification and Testing** (SVT'2016). Topics include: new results in formal verification and testing, technologies to improve the usability of formal methods in software engineering, applications of mechanical verification to large scale software, etc.

April 03-08 **Track on Software Engineering** (SE'2016). Topics include: software architecture, and software design patterns; standards; maintenance and reverse engineering; quality assurance; verification, validation, testing, and analysis; safety, security, and risk management; dependability and reliability; fault tolerance and availability; formal methods and theories; component-based development and reuse; empirical studies, and industrial best practices; applications and tools; distributed, embedded, real-time, high-performance, highly dependable systems; etc.

☺ Apr 04-08 **Track on Programming Languages** (PL'2016). Topics include: compiling techniques, domain-specific languages, garbage collection, language design and implementation, languages for modeling, model-driven development, new programming language ideas and concepts, practical experiences with programming languages, program analysis and verification, programming languages from all paradigms, etc.

☺ Apr 04-08 **Track on Multicore Software Engineering, Performance, Applications and Tools** (MUSEPAT'2016). Topics include: software engineering for multicore (CPU or GPU); specification and modeling of multicore systems; programming models, languages, compiler techniques and development tools for multicore; parallel and distributed testing and debugging; evolving sequential software to leverage multicore and manycore hardware; performance and optimization of multicore software; domain- and platform-specific multicore software issues (e.g., issues in scientific computing); etc.

☺ Apr 04-08 **Track on Object-Oriented Programming Languages and Systems** (OOPS'2016). Topics include: aspects and components; code generation and optimization; distribution and concurrency; formal verification; integration with other paradigms; interoperability, versioning and software evolution and adaptation; language design and implementation; modular and generic programming; runtime verification; secure and dependable software; static analysis; testing and debugging; type systems; etc.

April 05-08 13th Working IEEE/IFIP **Conference on Software Architecture** (WICSA'2016), Venice, Italy. Topics include: re-factoring and evolving architecture design decisions and solutions; techniques and tools for technical debt management; managing architecture risk over the life cycle of a system; architecture description languages and model driven architecture; software architecture modelling, analysis methods and tools; software architecture for legacy systems and systems integration; open architectures, product-line architectures, software ecosystems, systems of systems; software architects' roles and responsibilities; training, education, and certification of software architects; industrial experiments and case studies; etc. Deadline for submissions: January 11, 2016 (abstracts), January 18, 2016 (papers).

April 05-08 10th **Conference for Component-Based Software Architectures** (CompArch'2016), Venice, Italy. Topics include: re-factoring and evolving architecture design decisions and solutions; techniques and tools for technical debt management; managing architecture risk over the life cycle of a system; architecture description languages and model driven architecture; software architecture modelling, analysis methods and tools; software architecture for legacy systems and systems integration; open architectures, product-line architectures, software ecosystems, systems of systems; software architects' roles and responsibilities; training, education, and certification of software architects; industrial experiments and case studies; etc. Deadline for submissions: January 11, 2016 (abstracts), January 18, 2016 (papers).

April 06-08 8th **International Symposium on Engineering Secure Software and Systems** (ESSoS'2016), London, UK. Topics include: automated techniques for vulnerability discovery and analysis; programming paradigms, models, and domain-specific languages for security; verification techniques for security properties; security by design; static and dynamic code analysis for security; processes for the development of secure software and systems; embedded software security; etc.

April 10-15 9th IEEE **International Conference on Software Testing, Verification and Validation** (ICST'2016), Chicago, Illinois, USA. Topics include: security testing, embedded software testing, testing concurrent software, testing large-scale distributed systems, testing in multi-core environments, quality assurance, model checking, testing of open source and third-party software, software reliability, formal verification, experience reports, etc. Deadline for submissions: January 12, 2016 (testing tool demos), January 22, 2016 (Ph.D. symposium), February 7, 2016 (tutorials, technical briefings).

♦ April 11-13 18th **International Real-Time Ada Workshop** (IRTAW'2016), Benicàssim, Spain. In cooperation with Ada-Europe. Deadline for submissions: January 22, 2016 (position papers).

April 27-28 11th **International Conference on Evaluation of Novel Approaches to Software Engineering** (ENASE'2016), Rome, Italy. Topics include: comparing novel approaches with established traditional practices and evaluating them against software quality criteria, software and systems development methodologies, software process improvement, software product line engineering, architectural design and frameworks, software quality management, software change and configuration management, application integration technologies, geographically distributed software engineering, formal methods, model-driven engineering, etc. Deadline for submissions: January 5, 2016 (position papers).

April 27-29 XIX **Iberoamerican Conference on Software Engineering** (CIbSE'2016), Quito, Ecuador. Topics include: languages, methods, processes, and tools; reverse engineering and software system modernization; software evolution and maintenance; model-driven engineering; proof, verification, and validation; quality, measurement, and assessment of products and processes; formal methods applied to software engineering; software product families and variability; software reuse; etc.

☺ May 14-22 38th **International Conference on Software Engineering** (ICSE'2016), Austin, Texas, USA. Deadline for submissions: January 13, 2016 (posters proposals), January 22, 2016 (workshop papers).

> May 15 4th FME **Workshop on Formal Methods in Software Engineering** (FormaliSE'2016). Topics include: integration of FMs in the software development life cycle, ability of formal methods to handle real-world problems, formal methods in a certification context, "lightweight" or usable FMs, application experiences, formal approaches to safety and security related issues, cyber physical systems, scalability of FM applications, rigorous software engineering approaches and their tool support, formal approaches to safety and security related issues, case studies developed/analyzed with formal approaches, etc. Deadline for submissions: January 22, 2016.

☺ May 17-20 19th IEEE **International Symposium On Real-Time Computing** (ISORC'2016), York, UK. Topics include: object/component/service-oriented real-time distributed computing (ORC) technology; programming and system engineering (ORC paradigms, languages, model-maintenance, time-predictable systems, ...), system software (real-time kernels, middleware support for ORC, extensibility, synchronization, scheduling, fault tolerance, security, ...), applications (medical devices, intelligent transportation systems, industrial automation systems, embedded systems, ...), system evaluation (timing, dependability, fault detection and recovery time, ...), cyber-physical systems, etc. Deadline for submissions: January 12, 2016.

May 23-27 30th IEEE **International Parallel and Distributed Processing Symposium** (IPDPS'2016), Chicago, Illinois, USA.

May 30 - Jun 06 12th **International Conference on Open Source Systems** (OSS'2016), Gothenburg, Sweden. Topics include: adoption, use, acceptance of FLOSS; expanding scientific research and technology development methods through openness; security of FLOSS; interoperability, portability, scalability of FLOSS; open standards; reuse in FLOSS; FLOSS for education; FLOSS in the public sector; etc. Deadline for submissions: January 4, 2016 (papers, workshops), February 22, 2016 (panels, tutorials). Deadline for early registration: February 29, 2016.

June 01-03 20th **International Conference on Evaluation and Assessment in Software Engineering** (EASE'2016), Limerick, Ireland. Topics include: any aspect of empirical software engineering. Deadline for submissions: January 5, 2016 (abstracts), January 19, 2016 (regular research papers), March 9, 2016 (short papers, work in progress papers, too papers, industry papers), March 30, 2016 (doctoral symposium).

June 01-05 12th **International Conference on integrated Formal Methods** (iFM'2016), Reykjavík, Iceland. Topics include: hybrid approaches to formal modelling and analysis; i.e., the combination of (formal and semi-formal) methods for system development, regarding modelling and analysis, and covering all aspects from language design through verification and analysis techniques to tools and their integration into software engineering practice. Deadline for submissions: January 6, 2016 (papers).

June 05-07 15th **International Conference on Software Reuse** (ICSR'2016), Limassol, Cyprus. Topics include: COTS-based development and reuse of open source assets; generative development; domain-specific languages; software composition and modularization; model-driven development; reengineering for reuse; software product line techniques; quality assurance for software reuse, such as testing and verification; reuse of non-code artifacts (process, experience, etc.); transition to software reuse; industrial experience with reuse; software evolution and reuse; etc. Deadline for submissions: January 28, 2016 (workshop proposals, tutorial proposals, Doctoral Symposium papers, tool demos).

June 07-09 8th **NASA Formal Methods Symposium** (NFM'2016), Minneapolis, Minnesota, USA. Topics include: identifying challenges and providing solutions to achieving assurance in mission- and safety-critical systems; model checking; static analysis; model-based development; design for verification and correct-by-design techniques; applications of formal methods in the development of autonomous systems cyber-physical, embedded, and hybrid systems, ...; use of formal methods in: assurance cases, automated testing and verification, ...; etc. Deadline for submissions: February 19, 2016 (regular and short papers).

June 10-14 40th **Annual** IEEE **Computer Software and Applications Conference** (COMPSAC'2016), Atlanta, Georgia, USA. Event includes: symposiums on Computer Education and Learning Technologies (CELT), Embedded & Cyber-Physical Environments (ECPE), IT in Practice (ITIP), Novel Applications & Technology Advances in Computing (NATA), Security, Privacy and Trust in Computing (SEPT), Software Engineering Technology and Applications (SETA), etc. Deadline for submissions: March 6, 2016 (workshop papers).

♦ June 13-17 21st **International Conference on Reliable Software Technologies - Ada-Europe'2016**, Pisa, Italy. Sponsored by Ada-Europe, in cooperation (pending) with ACM SIGAda, SIGBED, SIGPLAN, and the Ada Resource Association (ARA). Deadline for submissions: January 17, 2016 (papers, tutorials, workshops, industrial presentations).

June 13-17 ACM **Conference on Programming Language Design and Implementation** (PLDI'2016), Santa Barbara, California, USA. Topics include: all areas of programming language research, including the design, implementation, theory, and efficient use of languages.

☺ June 28-30 **International Conference on Reliability, Safety and Security of Railway Systems** (RSSR'2016), Paris, France. Topics include: safety in development processes and safety management; combined approaches to safety and security; system and software safety analysis; formal modelling and verification techniques; system reliability; validation according to the standards; tool and model integration, toolchains; domain-specific languages and modelling frameworks; model reuse for reliability, safety and security; etc. Deadline for submissions: January 20, 2016.

July 05-07 **Software Technologies: Applications and Foundations** (STAF'2016), Vienna, Austria. Successor of the TOOLS federated event.

July 05-07 10th **International Conference on Tests And Proofs** (TAP'2016). Topics include: many aspects of verification technology, including foundational work, tool development, and empirical research; the connection between proofs (and other static techniques) and testing (and other dynamic techniques); verification and analysis techniques combining proofs and tests; program proving with the aid of testing techniques; deductive techniques to support testing: generating testing inputs and oracles, supporting coverage criteria, and so on; program analysis techniques combining static and dynamic analysis; testing and runtime analysis of formal specifications;

	model-based testing and verification; using model checking to generate test cases; testing of verification tools and environments; applications of testing and proving to new domains, such as security, configuration management, and language-based techniques; case studies, tool and framework descriptions, and experience reports about combining tests and proofs; etc. Deadline for submissions: January 29, 2016 (abstracts), February 5, 2016 (papers).
July 17-19	10th **International Symposium on Theoretical Aspects of Software Engineering** (TASE'2016), Shanghai, China. Topics include: theoretical aspects of software engineering, such as abstract interpretation, component-based systems, cyber-physical systems, distributed and concurrent systems, embedded and real-time systems, formal verification and program semantics, integration of formal methods, language design, model checking and theorem proving, object-oriented systems, run-time verification and monitoring, software architecture, software testing and quality assurance, software security and reliability, static analysis of programs, type systems and behavioural typing, tools exploiting theoretical results, etc. Deadline for submissions: January 10, 2016 (research abstracts), January 17, 2016 (research papers).
July 17-23	28th **International Conference on Computer Aided Verification** (CAV'2016), Toronto, Ontario, Canada. Topics include: theory and practice of computer-aided formal analysis methods for hardware and software systems, algorithms and tools for verifying models and implementations, program analysis and software verification, verification methods for parallel and concurrent systems, testing and run-time analysis based on verification technology, applications and case studies in verification, verification in industrial practice, formal models and methods for security, etc. Deadline for submissions: January 17, 2016 (abstracts), January 29, 2016 (papers).
☺ July 18-22	30th **European Conference on Object-Oriented Programming** (ECOOP'2016), Rome, Italy. Topics include: theory, design, implementation, optimization, and analysis of programming languages that enable or enforce abstractions across various programming styles, from object-orientation to reactivity to spreadsheets; innovative and creative solutions to real problems; evaluations of existing solutions in ways that shed new insights; etc. Deadline for submissions: January 22, 2016 (workshops).
August 01-03	IEEE **International Conference on Software Quality, Reliability and Security** (QRS'2016), Vienna, Austria. Merger of SERE (International Conference on Software Security and Reliability) and QSIC (International Conference on Quality Software). Topics include: reliability, security, availability, and safety of software systems; software testing, verification and validation; metrics, measurements, and analysis; software vulnerabilities; formal methods; benchmark, tools, and empirical studies; etc. Deadline for submissions: January 15, 2016 (workshops), March 25, 2016 (regular papers), April 22, 2016 (workshop papers, student papers), April 29, 2016 (fast abstracts).
August 02-05	11th IEEE **International Conference on Global Software Engineering** (ICGSE'2016), Orange County, California, USA. Theme: "Software Bridging Distances Between People". Topics include: industrial offshoring and outsourcing experiences, lean and agile development, methods and processes, mining software repositories and software analytics, open source software communities, security and privacy, software evolution and maintenance, strategic issues in distributed development, tools and infrastructure support, etc. Deadline for submissions: January 22, 2016 (workshops), January 29, 2016 (research, industry and short paper abstracts), February 5, 2016 (research, industry and short papers, tutorials), February 26, 2016 (Doctoral Symposium submissions).
Aug 31 - Sep 09	42nd **Euromicro Conference on Software Engineering and Advanced Applications** (SEAA'2016), Limassol, Cyprus. Topics include: information technology for software-intensive systems; embedded software engineering (ESE); model-based development, components and services (MOCS); software process and product improvement (SPPI); teaching, education and training for dependable embedded and cyberphysical systems (TET-DEC); cyber-physical systems (CPS). Deadline for submissions: February 8, 2016 (abstracts), February 21, 2016 (papers).
December 10	Birthday of Lady Ada Lovelace, born in 1815. Happy Programmers' Day!

Preliminary Call for Participation
7th Ada Developer Room at FOSDEM 2016
Saturday 30 January 2016, Brussels, Belgium

Organized by Ada-Belgium
in cooperation with Ada-Europe

FOSDEM[1], the Free and Open source Software Developers' European Meeting, is a non-commercial two-day weekend event organized early each year in Brussels, Belgium. It is highly developer-oriented and brings together 5000+ participants from all over the world. The 2016 edition takes place on Saturday 30 and Sunday 31 January. It is free to attend and no registration is necessary.

In this edition, Ada-Belgium[2] organizes once more a series of presentations related to Ada and Free or Open Software in a s.c. Developer Room. The "Ada DevRoom" at FOSDEM 2016 is held on the first day of the event. The program offers introductory presentations on the Ada programming language, including features of the new Ada 2012 standard, as well as more specialised presentations on focused topics. An important goal is to present exciting Ada technology and projects also to people outside the traditional Ada community. There's time for discussion and interaction, opportunities to experience the Beaujolais effect(!), and to conclude the day we organize the by now traditional "Adaists dinner" on Saturday evening...

The Ada at FOSDEM 2016 web-page has more details, such as the full list with abstracts of presentations, biographies of speakers, and the concrete schedule. For the latest information at any time, contact <Dirk.Craeynest@cs.kuleuven.be>, or see:

http://www.cs.kuleuven.be/~dirk/ada-belgium/events/16/160130-fosdem.html

[1]https://fosdem.org/2016
[2]http://www.cs.kuleuven.be/~dirk/ada-belgium

18th International Real-Time Ada Workshop – IRTAW 2016

Hotel Voramar, Benicàssim, Spain
11-13th April 2016

http://www.ada-europe.org/irtaw2016

Call for Papers

The International Real-Time Ada Workshop series has provided a forum for identifying issues with real-time system support in Ada and for exploring possible approaches and solutions, and has attracted participation from key members of the research, user, and implementer communities worldwide. Recent International Real-Time Ada Workshop meetings contributed to the Ada 2005/Ada 2012 standards, especially with respect to the tasking features, the real-time and high-integrity systems annexes, and the standardization of the Ravenscar Tasking Profile.

In keeping with this tradition, the goals of IRTAW-18 will be to:

- Review Ada 2012 Issues vis-a-vis real-time systems;
- Examine experiences in the use of Ada 2012 for real-time systems and applications;
- Implementation approaches for Ada 2012 real-time features;
- Consider developing other real-time Ada profiles in addition to the Ravenscar profile;
- Analyze the implications to Ada with multiprocessors in development of real-time systems;
- Investigate paradigms for using Ada for real-time distributed systems, with special emphasis on robustness as well as hard, flexible and application-defined scheduling;
- Analyse specific patterns and libraries for real-time systems development in Ada;
- Evaluate Ada in context of the certification of safety-critical and/or security-critical real-time systems;
- Examine the Real-Time Specification for Java and other languages for real-time systems development, their current implementations and their interoperability with Ada in embedded real-time systems;
- Investigate industrial experience with Ada and the Ravenscar Profile in real-time projects;
- Consider the language vulnerabilities of the Ravenscar and full language definitions;
- Consider testing for compliance with the Real-Time Annex.

Participation at IRTAW-18 is by invitation following the submission of a position paper addressing one or more of the above topics or related real-time Ada issues. Alternatively, anyone wishing to receive an invitation, but for one reason or another is unable to produce a position paper, may send in a one-page position statement indicating their interests. Priority will be given to submitted papers.

Submission Requirements

Position papers should not exceed ten pages in typical IEEE conference layout, excluding code inserts. All accepted papers will appear, in their final form, in the Workshop Proceedings, which will be published as a special issue of Ada Letters (ACM Press). Selected papers will also appear in the Ada User Journal. Authors with a relevant paper submitted to the 21st International Conference on Reliable Software Technologies – Ada-Europe 2016 (deadline 17 January, 2016) may offer an extended abstract of the same material to IRTAW 18. Please submit position papers, in PDF format, to the Program Chair by e-mail: *stephen.michell@maurya.on.ca*

Important Dates

Paper Submission: **22 January, 2016**
Notification of Acceptance: 19 February, 2016
Confirmation of Attendance: 4 March, 2016
Final Paper Due: 25 March, 2016
Workshop: April 11-13, 2016

Program Chair
Stephen Michell, *Maurya Software Inc, Canada*

Workshop Chair
Jorge Real, *Universitat Politècnica de València*

AdaCore The GNAT Pro Company · Ada europe · DISCA DEPARTAMENT D'INFORMÀTICA DE SISTEMES I COMPUTADORS · UNIVERSITAT POLITÈCNICA DE VALÈNCIA

21st International Conference on Reliable Software Technologies
Ada-Europe 2016
13-17 June 2016, Pisa, Italy

Conference Chair
Giorgio Buttazzo
Scuola Superiore Sant'Anna

Program Co-Chairs
Marko Bertogna
Univ. of Modena and Reggio Emilia

Luís Miguel Pinho
CISTER/INESC-TEC, ISEP

Special Session Chair
Eduardo Quiñones
Barcelona Supercomputing Center

Tutorial and Workshop Chair
Jorge Real
Universitat Politècnica de València

Industrial Co-Chairs
Marco Di Natale
Scuola Superiore Sant'Anna

Tullio Vardanega
Università di Padova

Publication Chair
Geoffrey Nelissen
CISTER Research Centre/ISEP

Exhibition Co-Chairs
Paolo Gai
Evidence Srl

Ahlan Marriott
White Elephant GmbH

Publicity Co-Chairs
Mauro Marinoni
Scuola Superiore Sant'Anna

Dirk Craeynest
Ada-Belgium & KU Leuven

Local Chair
Ettore Ricciardi
ISTI-CNR, Pisa

General Information

The **21st International Conference on Reliable Software Technologies – Ada-Europe 2016** will take place in Pisa, Italy. Following its traditional style, the conference will span a full week, including a three-day technical program and vendor exhibition from Tuesday to Thursday, along with parallel tutorials and workshops on Monday and Friday.

Schedule

17 January 2016	Submission of papers, industrial presentations, tutorial and workshop proposals
10 March 2016	Notification of acceptance to all authors
24 March 2016	Camera-ready version of papers required
2 May 2016	Industrial presentations, tutorial and workshop material required

Topics

The conference is a leading international forum for providers, practitioners and researchers in reliable software technologies. The conference presentations will illustrate current work in the theory and practice of the design, development and maintenance of long-lived, high-quality software systems for a challenging variety of application domains. The program will allow ample time for keynotes, Q&A sessions and discussions, and social events. Participants include practitioners and researchers representing industry, academia and government organizations active in the promotion and development of reliable software technologies.

This edition of Ada-Europe features a focused **Special Session on Safe, Predictable Parallel Software Technologies**. Following the intensifying trend of usage of Multi-/Many-core systems, it is increasingly important to assess how reliable software technologies need to adapt to these complex platforms, as well as how parallel models need to adapt to domains in which safety and predictability is a must. Topics include (but are not limited to): **Predictable Parallel Programming Models**, **Parallel Language Technologies, Compiler Support for Parallel Execution**, **Parallel Runtimes and Libraries**, **Automatic Parallelization**, **Safety Issues and Reliability Mechanisms for Parallel Execution**, **Software Modelling and Design Approaches, Hardware Support for Predictability of Parallel Software**.

For the **general track of the conference**, topics of interest include but are not limited to (full list on the website): Real-Time and Embedded Systems, Mixed-Criticality Systems, Theory and Practice of High-Integrity Systems, Software Architectures, Methods and Techniques for Software Development and Maintenance, Formal Methods, Ada Language and Technologies, Software Quality, Mainstream and Emerging Applications, Experience Reports in Reliable System Development, Experiences with Ada.

Accepted papers will be published in the Lecture Notes in Computer Science (LNCS) series by Springer. **Selected papers of the conference will be also invited for special issues of Springer's Computing Journal (general track papers) and Journal of Parallel Programming (special session papers).**

In cooperation with
ACM SIGAda, SIGPLAN, SIGBED

acm In-Cooperation

(approval pending)

and

Ada Resource Association (ARA)

Call for Regular and Special Session Papers

Authors of papers which are to undergo peer review for acceptance are invited to submit original contributions by 17 January 2016. Paper submissions shall not exceed 14 LNCS-style pages in length. Authors for both the general track and the special session shall submit their work via EasyChair at https://easychair.org/conferences/?conf=adaeurope2016. The format for submission is solely PDF.

The International Conference on Reliable Software Technologies is ranked class A in CORE; Microsoft Academic Search has it in the top third for conferences on programming languages. The conference is listed in DBLP, SCOPUS and Web of Science Conference Proceedings Citation index, among others.

Proceedings

Conference proceedings will be published in the Lecture Notes in Computer Science (LNCS) series by Springer, and will be available at the conference. Camera-ready accepted papers are due strictly by 24 March 2016 (format guidelines available at in the conference site). Failure to comply and to register for the conference by that date will prevent the paper from appearing in the proceedings.

Journal Special Issues

Selected papers of the conference will be invited for a special issue of Springer's Computing Journal on the conference topic. Papers of the special session will be invited for a special issue of Springer's Journal of Parallel Programming.

Call for Industrial Presentations

The conference seeks industrial presentations which deliver value and insight but may not fit the process for regular papers. Authors are invited to submit a presentation outline of exactly 1 page in length by 17 January 2016, using EasyChair at https://easychair.org/conferences/?conf=adaeurope2016. The format for submission is solely PDF.

The authors of selected presentations shall prepare a short abstract by 2 May 2016, aiming at a 20-minute talk. Authors will be also invited to submit corresponding articles for publication in the *Ada User Journal* (http://www.ada-europe.org/auj/), which will host the proceedings of the Industrial Program. For further information please contact the Industrial Co-chairs directly.

Awards

Ada-Europe will offer honorary awards for the best regular paper and the best presentation.

Call for Tutorials

Tutorials should address subjects that fall within the scope of the conference and may be proposed as either half- or full-day. Proposals should include a title, an abstract, a description of the topic, a detailed outline of the presentation, a description of the presenter's expertise in general and with the proposed topic in particular, the duration (half day or full day), the intended level of the tutorial (introductory, intermediate, or advanced), the recommended audience experience and background, and a statement of the reasons for attending. Proposals should be submitted by e-mail to the Tutorial Chair. Authors of accepted full-day tutorials will receive a complimentary conference registration and a fee for every paying participant in excess of 5; for half-day tutorials, these benefits will be accordingly halved. The *Ada User Journal* will offer space for the publication of summaries of the tutorials.

Call for Workshops

Workshops on themes that fall within the conference scope may be proposed. Proposals may be submitted for half- or full-day events, to be scheduled at either end of the conference week. Proposals should be submitted to the Tutorial and Workshop Chair. The workshop organizer shall also commit to preparing proceedings for timely publication in the *Ada User Journal*.

Call for Exhibitors

The commercial exhibition will span the three days of the main conference. Vendors and providers of software products and services should contact the Exhibition Chair for information and for allowing suitable planning of the exhibition space and time.

Grants for Reduced Student Fees

A limited number of sponsored grants for reduced fees is expected to be available for students who would like to attend the conference or tutorials. Contact the Conference Chair for details.

Venue

The conference will take place at Scuola Superiore Sant'Anna (left images, including the aula magna where sessions will take place), in the heart of Pisa, Italy. June is full of events in Pisa, including in the conference week the Saint Patron's festivities (San Ranieri) with the Luminara on the night of June 16 (thousands of candles burn and reflect on the river – right image). Plan in advance! It is absolutely worth it!

The Ada Lovelace Symposium in Oxford

Gabriela Asli Rino Nesin
Computer Science, University of Oxford, UK; email: gabriela.rinonesin@cs.ox.ac.uk

1 Introduction

December 10th 2015 marks the bicentennial anniversary of the birth of Ada Augusta Lovelace, née Byron. She was, and still is, such a complex figure that it is hard to summarize her in one sentence, and even a whole two days of talks surrounding her and her achievements could but scratch the surface of such an interesting life. Still, this article aims to shortly summarize the various strands of discussion mentioned during the Ada Lovelace Symposium, which took place in the University of Oxford on the 9th and 10th of December of this year. By the end of the article I hope to have given an idea of the various points of view on who Ada was, what she did and what she signifies now and for the future, but also to have made it clear why this Symposium was organized in Oxford, by a computer science department (and in fact mainly by a woman who is both a computer scientist and a mathematician).

But first, some caveats. It should be said that this will not be a linear summary of who talked about what – for that purpose the reader can find the list of abstracts on this page: https://blogs.bodleian.ox.ac.uk/adalovelace/Symposium/. Instead, I have strived to divide the discussion into the main themes that made up the rich tapestry of the Symposium. As such, equal weight will not necessarily be given to all speakers – it will be coloured by what was given more emphasis in my eyes and may unconsciously reflect some of my opinions on the subject as a woman in computer science. I will assume the reader is familiar with Ada Lovelace and her importance to computing to a degree that can be gleaned from Wikipedia.

The first theme explored is the issue of Ada Lovelace's family issues and family history, which are seen to have had a great deal of influence not only on her later personal life but also on her science. The contrast between the influence of her mother and that of father is often seen as analogous to the contrast between mathematics and art (poetry in particular of course), or between logic and imagination - Ada's parents are depicted as the embodiment of these two supposedly opposing forces. Whether Ada was constantly buffeted by these opposing winds, or whether she managed to make the two extremes work together to their greatest advantage was another main question discussed during the Symposium. The third theme is that of her personal demons; psychological illness, gambling, drug addiction and promiscuity. Though these are disputed subjects and were not dwelt on very much in the seminar, they were still mentioned repetitively enough to be of note, and there are likely some comments to be made here about historical judgments on the failures of intelligent women. Next I go on to a subject of great biographical importance: her academic relationships to many great minds of the day, a section which provides a nice introduction to the following most important theme: that of Ada's achievements. Much was said in the Symposium about her intellect, confidence and scientific acumen. The questions of why she is special and whether or not she was indeed the first computer programmer were asked. Lastly, there were a number of talks and comments that were related though not central to Ada Lovelace – in other words, offshoots of interest, which I will cover in the last section.

2 Ada's Family Life

Ada (to avoid confusion with her change of last names, I will refer to her by her first name only – this should not be assumed to be a lack of respect. When talking about the programming language, I will specify so) is said to have been born a celebrity due to her famous, or notorious father. Many speakers at the Symposium (Betty Toole, Julia Markus, Richard Holmes, Valerie Barr, Elizabeth Bruton) mentioned the tumultuous and short-lived relationship between her parents, and the conflicts between them seem to have been a great influence in Ada's life. She is often seen as a fundamentally dichotomic figure attempting to reconcile two opposing influences, that of her father and that of her mother. On the side of Lord Byron lie madness, sin but also creativity, poetry and art. The side of her mother, Lady Millbanke, is most often associated with rectitude, reason, mathematics and rationality. Whether or not this dichotomy was as strict as it seems was discussed, as well as whether these two influences were in conflict or reconciled within Ada. Sydney Padua, in her comic "The Thrilling Adventures of Lovelace and Babbage", uses this conflict as a storytelling tool, but accepts that in reality Ada had a foot in both of the "mathematics" and "poetry" camps and did not exclude one to the detriment of the other. Some see Lady Millbanke's strict scientific education of Ada (hiring many tutors in the process) as an effort to counteract the nefarious poetical influence of her father's heredity, but speakers such as Julia Markus argue that not only were humanities not a part of the curriculum for a young lady's education at that time, Lady Millbanke was not nearly as averse to poetry as such claims make it seem – she even jokingly chastises young Ada, in a letter, for being too much of a "nerd", in modern parlance. This idea of dangerous poetry is likely extrapolated from comments made by Lord Byron, who for example jokingly writes to Lady Millbanke that he hopes his daughter has none of the passion which proved such a nefarious influence in his case.

What is sure is that a single mother in the Victorian era was a very rare thing indeed, and that Lady Millbanke is a very impressive character in her own right. There are very difering views on the nature of this mother and daughter relationship – whether it was "cordial distaste" as a third-hand account shown by Sydney Padua implies, or on the contrary very close, was up for debate. The truth, as usual, most probably lies in the middle. For example, Julia Markus tells the story of Ada as a young girl taking an interest in aviation and postal routes as a way to get her letters to her mother, who is recuperating from illness in Brighton, more quickly, and Betty Toole tells the story of her mother paying off Ada's gambling debts (towards the end of Ada's life) and defending one of Ada's affairs.

Ada never met her father, and the influence of heredity being quite exaggerated in the Victorian era is likely not to have left Ada indifferent. Lord Byron also writes yearning poetry to his daughter, reeling her in despite her absence, reports Julia Markus. Not only does her mother worry about raising her without a father (again from Julia Markus's talk), but Ada seems to also have quite a yearning for this relationship, desiring to be buried next to Byron at the end of her life (according to Betty Toole, Ada does not dare to ask her mother for this, and therefore is very relieved when her mother finally suggests it herself).

3 Mathematics versus Poetry, or "Will you concede me poetical science?"

Analogous to and symbolic of the mother/father dichotomy is the science/art dichotomy. Many speakers emphasized how Ada integrated facets of both – being very obsessed with formally understanding the world, but also given to flights of fancy. Richard Holmes recounts for example how young Ada wrote letters in French from the point of view of Madame Puff, her cat. Her interest in avionics and flying machines is in part motivated from being bedridden with the mumps for 18 months – which child wouldn't dream of flying in such a situation?

Even in her academic writing and letters, many speakers showed that Ada had a tendency to use quite flowery language – for example comparing ideas that she couldn't quite grasp to will-o-the-wisps, dubbing herself the Analytical Engine's "high priestess" and writing to her mother that she would soon be "an autocrat", commanding regiments of numbers marching to the tune of musical notes (quoted by Imogen Forbes-Macphail). Her use of vivid imagery in her scientific writing, reports Richard Holmes, was likely influenced by Mary Somerville, another female scientist who acted as her mentor and introduced her to Charles Babbage.

Ada herself does not see imagination as something which is in conflict with reason, but indeed complementary, as evidenced by the quote given by Richard Holmes that those educated in the exact sciences can fly soar even further on the wings of imagination (paraphrased).

4 Ada's Demons

Focusing on Ada's troubles and faults to the detriment of her achievements should not be the point of her commemoration, and indeed these troubles were mentioned in the Symposium mainly to criticize those who brought these "faults" to the forefront and dwelt on them as defining her life (see the section before last of this article), and also to humanize Ada and show what she overcame.

The list of Ada's troubles runs long, and these troubles are sometimes seen as stemming from the negative influence of her father. Among these troubles are gambling (she is said to have been quite cocky about her scientific "system" to win at gambling, despite the fact that she lost much of her husband's and mother's money), adultery and promiscuity, drug use (mainly opiates such as Laudanum, though it is unclear whether she self-medicated for the pain caused by the uterine cancer she later died of).

Many speakers, for example Sydney Padua, discussed the diagnosis of manic or manic-depressive that many biographers give Ada. Her tendency for 14-page missives and forceful language is seen in historical literature as evidence for this illness. The speakers in this Symposium mainly argued that there was no real evidence for this diagnosis, and even that it may have been caused by sexist judgement – Hollings states that compared to Coleridge or (Percy) Shelley, Ada is a model of calm.

5 Ada's professional relationships

Ada's name is rarely mentioned without Charles Babbage's, and this Symposium was no exception. Ada was well-known as a socialite, but she first made a name for herself (or rather initials for herself! Ada signed her academic writing with her initials AAL rather than her full name, presumably to avoid sexist reactions) in scientific circles when she translated Luigi Menabrea's article on Babbage's Difference Engine, and added to them her Translator's Notes which tally up to thrice the original length of the article. Her Translator's Notes are one of the main reasons her name is indelibly tied to Babbage's, and we shall mention them again in the next section.

Many speakers at the Symposium emphasized the friendly relationship between Babbage and Ada, despite the large age gap (they met when he was 45 and she 18). They corresponded extensively about work, and their correspondence is a major source for the historians in this Symposium – as Betty Toole says, sometimes one finds as many as two or three letters in the same day.

Babbage clearly had great admiration for the young woman: the audience was shown an excerpt from a letter from Babbage to Faraday, introducing Ada and saying her intellect is truly astounding and surpasses most men's – though despite this, Faraday refused to correspond with Ada about his work. Some historians have argued that Babbage only said such things to obtain Ada's patronage, but speakers in this Symposium argue against this: Elizabeth Bruton arguing that he posthumously praised her very highly in his memoir, despite having no possibility of getting her patronage after her death. Sydney Padua gives an entertaining argument, also against the claim that Babbage used Ada for her money, saying that having understood Babbage's character quite well through her readings, she can't imagine him tolerating someone he thought a fool for more than five minutes, no matter how rich they were. Another amusing peek into Ada and Babbage's relation provided by Sydney Padua is some evidence that Babbage may have seen the young woman as overly serious and occasionally poked fun at her by telling her tall tales.

Similarly, Ada had a great admiration for Babbage. Where he saw her as somewhat dry, she found him too messy, on the contrary, and in a quote reported by Ursula Martin, wrote to her mother that she wishes Babbage would let her manage his stuff – in Ada's words, to let her be his “whipper-in”. All in all, the duality and cooperation between Ada and Babbage was, throughout the Symposium, taken to be similar to a division of labour between hardware and software, Babbage being seen as very much the mechanical engineer, dealing with the hardware side of things, and Ada thinking of the implications and abstract ideas that could be implemented on the hardware.

Another name strongly associated with Ada's is Mary Somerville, famous natural scientist that Ada met through her mother. Ada met Somerville just before she met Babbage - in fact Somerville is the one who introduced the two. The two women, again despite the age gap, have a very strong relationship, Somerville being both a teacher and a mentor to Ada. Ada often stayed overnight at the Somervilles, and it is clear that there was a deep, almost familial, friendship there.

The last well-known name associated with Ada's mentioned quite often in the Symposium, and concentrated on in Christopher Hollings's talk, was the famous mathematician Augustus De Morgan. Ada and he had an extensive correspondence in which he essentially gave her a distance course in mathematics, free of charge. Suffice to say that there was mutual admiration here as well.

6 Ada's intellect and achievements

There seems to be quite a controversy between historians regarding Ada's intellect, and whether she was in fact the first computer programmer. Dealing with this controversy was the main thrust of Christopher Hollings's talk, where he showed that she was neither stupid, as some argue, nor a genius, the truth lying, as usual, somewhere in between. She did indeed have difficulty grasping some concepts, much of her merit lying in her tenacity and determination to go to the very fundamentals and to relentlessly ask questions until she understood the subject matter. Both Elizabeth Bruton and Betty Toole argued in the Symposium that Ada's Translator's Notes go much further than Menabrea had ever done, and not only add much more detail and examples but also tease out some implications and future applications of the subject matter.

With regards to the controversy around Ada being the first programmer, Elizabeth Bruton argues that the valid question is much more whether what she wrote could be considered a program in the modern sense of the word, than whether she was actually the one to write it. She certainly did do some debugging of one of Babbage's "programs", designed to calculate Bernouilly numbers.

Many of the speakers at the Symposium agreed that Ada's main achievement and the reason she merits to be called a visionary is that she was the one who really recognized the potential of the Analytical Engine: that its merit lies in its universality. This is namely mentioned by Doron Swade and Betty Toole – as hinted at in the section above, Ada recognizes that the Analytical Engine is in essence hardware on which, with some modifications, different kinds of software could be made to run. Hence her questions on what else such an engine could do – could it compose music, write poetry? Ada compares, very justly, the Analytical Engine to the Jacquard loom, where instead of weaving leaves and patterns out of thread, the engine weaves algebraic patterns. Imogen Forbes-MacPhail quotes her, notably, as saying “the machine could act on any objects whose mutual fundamental relations could be expressed by the abstract science of operations”.

The above leads us to another point widely discussed at the Symposium, that of whether or not Ada believed in artificial intelligence and machine learning as we understand them now. Imogen Forbes-MacPhail suggested in her talk that a question more suited to Ada was not whether machines could think, but whether they could create art. After all, as emphasized by many speakers at the Symposium, Ada was not of the opinion that the Analytical Engine could think, as evidenced by a quote that Turing later made famous by calling it her “Lovelace's Objection”: “The analytical engine has no pretensions whatever to originate anything. It can do whatever we know how to order it to perform”, but Turing himself realizes that this limitation is due to the nature and specificities of the Analytical Engine, and not a general claim.

Ada's preoccupation with ethics was also emphasized by both Betty Toole and Moshe Vardi: her famous quote here is: “Far be it from me, to disclaim the influence of ambition and fame;…I wish to add my mite towards… the most effective use of mankind ”.

7 Ada Lovelace as a symbol, pop culture icon and feminist role model

Richard Holmes, in his talk, situated Ada in an era with a rich tradition of female scientists: Sophie Germain, Madame du Châtelet, Mary Somerville, Maria Mitchell and others. Together with these women she has become quite a symbol for women in STEM. Recognition is noted with tokens such as the Ada Lovelace Medal awarded by BCSWomen, but also by hommages such as the naming of the Ada programming language. Despite this, Ada is sometimes overlooked in writings about the history of computer science, or often takes a back seat to Babbage, as Elizabeth Bruton notes. Also, as Ursula Martin pointed out, Ada, being a woman of her era, is not naturally affiliated with any academic institution, which means none "officially" promotes her. The reason this Symposium was held in Oxford is the presence of her archives in the Bodleian, kindly provided by the Earl of Lytton and curated by Mary Clapinson. Such are the consequences of having been a woman scientist in the 19th century.

Being such an unusual character, Ada has found her way into popular culture, namely into the steampunk genre. Besides the obvious reference to Sydney Padua's "The Thrilling Adventures of Lovelace and Babbage", Elizabeth Bruton also mentioned the 1990 novel "The Difference Engine" by William Gibson and Bruce Sterling, among others.

Despite all the positive recognition of Ada's achievements, in the panel discussion chaired by Muffy Calder and including talks by Murray Pittock, Valerie Barr, Cheryl Praeger and Suw Charman-Anderson, certain problematic aspects of her remembrance and of the way we make her a symbol were pointed out. To emphasize the inherent unfairness, Murray Pittock dared the audience to come up with a male scientist whose achievements were so framed by the influence of her parents. Similarly, Suw Charman-Anderson pointed out the need to classify women in a dichotomic way as angels or bad women, and to unjustly point out their flaws – thus, Ada's adultery and other supposed faults have been very much dwelt on as defining aspects of her life, as are those of other women scientists such as Marie Curie and Rosalind Franklin. She asks where this desire to bring women scientists "down a notch" comes from and argues that it is nefarious in terms of the future – it tells young women going into STEM that a woman cannot be both an intelligent scientist and "normal", whatever that may indeed mean. Muffy Calder and Cheryl Praeger both point out how strange it is that we should have as a role model a woman so different from most of us women now in STEM – an aristocrat, living 200 years ago, who has never gone to school. Lastly, Valerie Barr explained to the audience how the dichotomy – either picturing female scientists as superheroes, or as extremely flawed beings, is both false and has harmful effects for potential new scientists.

The difficulties inherent to being a female scientist were also brought to the fore, many mentioning Ada's signing of her translator's notes using her initials in order to hide her gender. Sydney Padua noted that Ada's Translator's Notes have a distinctly androgenous style as opposed to her letters and other writings. This was echoed during Dame Stephanie "Steve" Shirley's inspirational speech during the conference banquet, where she spoke of her own experiences and explained that she had also felt the need to take on a male pseudonym. Her joke about the plight of women in computer science made the whole dinner hall burst out into laughter and applause: "We women in CS can be told apart by the peculiar shape of our heads – flat on the top from being patted on the head so often!".

8 Other related subjects

Besides talking directly about Ada, much happened in the Symposium that was either indirectly tied to Ada and programming, or simply doesn't fit into the above categories. In an attempt not to exclude any of the contributions, I will list some of them here, in no particular order.

First, one cannot go without mentioning the music inspired by Ada: Emily Howard's "Ada sketches", composed algorithmically by encoding numbers into notes, a process she described in her talk and discussion with David De Roure, but also the music performed at the Symposium reception. The two world premieres of "Algorithmic study on ADA" and "ADA" composed by James Whitbourn and performed by brilliant musicians and choir delighted the audience.

Judith Grabiner gave a very wide-ranging coverage of the history of mathematics in what regards the western world's geometric view of the world, in particular Euclidean versus non-Euclidean geometry. Geometry is seen to be central to our understanding of the world (she mentions that Ada claimed she was unable to truly understand a proposition unless she could "see" it in her mind's eye), and in particular Euclid's Elements was mentioned as the foundational book every student of mathematics must digest at the time – its methodology set the tone for logic and mathematics for centuries to come.

John Barnes gave an entertaining historical talk about Byron and Babbage's time at Cambridge, Byron's bear, and Babbage's relationship with trains, thus transitioning, through a mention of system correctness, to the history and creation of the Ada programming language. Ada 2012, now integrating Spark 2014, allows one to focus on proving correctness of a program formally, rather than relying on testing.

Imogen Forbes-MacPhail's talk pointed out fascinating connections in philosophy of science, among others mentioning the mind-body problem and its relation to AI – are we but biological machines? Is there room for free will in Hartley's deterministic model of the mind, or are we merely passive, just like the romantic aeolian harp? She also mentioned the point, held by some thinkers of Ada's era, that creativity was mostly about rearranging and reorganizing ideas in new combinations, rather than coming up with new components. As such, she asks, could the Analytical Engine, or a similar machine, having the capacity both to analyze and synthesize, be seen as creative?

Both June Barrow-Green and Ursula Martin's talks focused on the importance of archives in this type of historical study, and the incredible opportunities that their digitization affords to historians of science.

Moshe Vardi gave the last talk of the conference, looking forward from Ada's time to the present day. He focused on whether CS advances have always been for the "good of mankind" as Ada would have liked. This "good of mankind" is exemplified by job losses and job creation, and automation's hand in this state of affairs. On a more upbeat note, Vardi mentioned incredible AI breakthroughs, including Deep Blue, quadruped robotics, self-driving cars, Watson and others.

Adrian Johnstone gave a fascinating overview of Babbage's quite mystifying mechanical notation – a shorthand he invented to write down and communicate the functioning of his engines – it comprises information about orientation, direction of movement and rotation of various parts of the design.

Both Bernard Sufrin and Sydney Padua explained the functioning of the two engines: Bernard Sufrin an abstract explanation of how the Difference Engine worked, as well as Ada's understanding of it, and Sydney Padua a visualisation of how the Analytical Engine would have functioned.

Conclusion

As one can see from the above, the Ada Lovelace Symposium covered a very wide range of subjects and was thus almost as interdisciplinary as Ada herself was. Scholars from different areas commented on their newfound understanding of other fields – computer scientists for example realizing how much digging through archives was involved in the study of history of science, and enchanted by riveting talks given without the use of a single slide or board (no such thing ever happens in computer science conferences!), humanities scholars delighted to have computer science talks made accessible to them. In all, the ambiance around all the proceedings was one of jovial camaraderie and mutual deep admiration and gratitude – I cannot imagine a better tribute to Ada Lovelace!

I hope to have given a taste of the experience of attending the Ada Lovelace Symposium – the readers who want to watch the talks for themselves can do so at the following link:
http://podcasts.ox.ac.uk/series/ada-lovelace-Symposium-celebrating-200-years-computer-visionary

From Byron to the Ada Language

John Barnes
11 Albert Road, Caversham, Reading, RG4 7AN, UK; Tel: +44 118 947 4125; email: jgpb@jbinfo.demon.co.uk

Abstract

This is an extended version of a talk given by the author at a Symposium in Oxford to celebrate the 200th birthday of Ada Lovelace last December. It starts with a few words about Byron and his Bear at Cambridge. This is followed by some remarks about Babbage who also went to Cambridge and was involved in an important event involving Brunel and Safety on the railways. The main topic is a look at the events that led to the new programming language devised to satisfy the needs of the US Department of Defense and which was named Ada in recognition of the fact that Ada Lovelace was the first programmer.

Keywords: Byron, Babbage, Ada.

1 Byron and his Bear

Byron went up to Trinity College, Cambridge in July 1805. He graduated as an MA in 1808. However, as a Noble he did not have to take any examinations in order to graduate but just had to stay in residence for the required number of terms (presumably nine as it is now). So he did not graduate with honours as a BA as would happen to commoners but jumped straight to MA.

Incidentally, an important examination at Cambridge is known as a tripos. Thus the examination for mathematics is the Maths Tripos. The name derives from the fact that originally the examination started with a viva and the student sitting on a three-legged stool.

Another curiosity of the time was that one had to take the Maths Tripos before any other. Thus if one wanted to read Classics or Theology then one had to pass the Maths Tripos with honours first. Seems very sensible to me. Apart from Classics and Theology there probably wasn't anything else one could read. The sciences had not been discovered much so there was no Natural Philosophy. But the Maths Tripos did cover a lot of mechanics and optics.

Another feature of being a Noble was that Byron did not have to eat with the Commoners but dined on High Table with the Fellows.

Byron wanted to have a dog in college but dogs were (and still are) forbidden. Other animals are permitted and cats are common. So Byron decided that he wanted a Bear. It is quite clear that he did indeed have a Bear but there is some doubt about where Bruin was kept.

Now according to John M F Wright who came up to Trinity in 1813 and writing much later in his anonymously published book, *Alma Mater*, in 1827 he says that Byron kept Bruin in Great Court. And then

"When Lord Byron was at Trinity, he kept in rooms on this staircase, round which you might drive a coach and six, and had, moreover, the use of the small Hexagonal one in the tower."

He is referring to K staircase in Great Court which has a tower/turret which contains a spiral staircase. K staircase and the diagonally opposite A staircase are in corners of the court. Figure 1 is a view of Great Court from the foot of K staircase showing clearly the opposite A staircase and its tower. Note the fountain in the centre of the court; the Master's lodge is to the left of A staircase. Figure 2 shows K staircase (with roadworks).

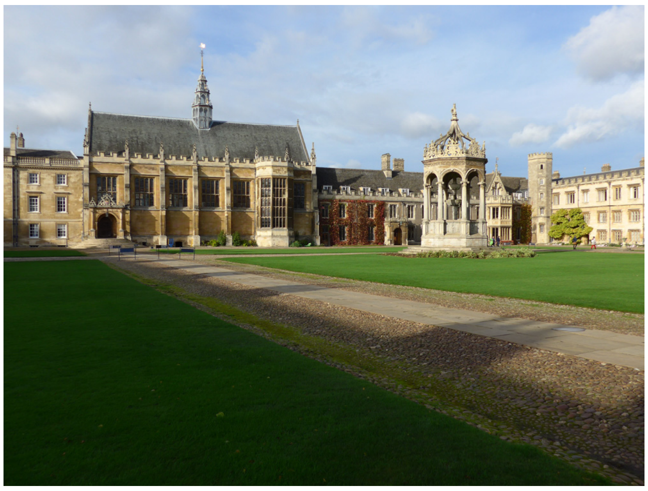


Figure 1 Great Court from K staircase

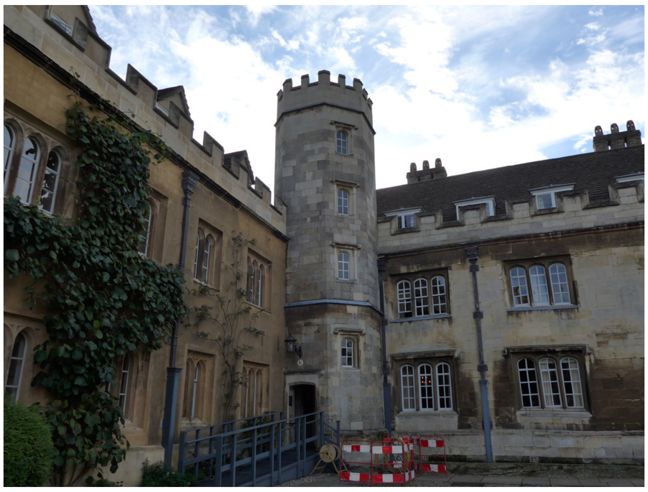


Figure 2 The turret at K staircase

The "hexagonal" room Wright refers to is at the top of the tower. If Byron did live on K staircase then he probably lived in a room on the first floor. Indeed K6 (with window open in the figure) is the best room although it would be tricky to drive a coach and six around it!

About 150 years later, by a strange coincidence, I was an undergraduate at Trinity, Cambridge and in my last year, I lived in Great Court in K6. I was told that I could well be living in Byron's old rooms. However, there was no sign of bears having been in the top of the turret which is now a toilet.

Further evidence is suggested by an item in a small book entitled *The Night Climber's Guide to Trinity* published in 1960. The K corner is known as Mutton-Hole corner for some reason and the book gives guidance on clambering on the roof around it. It says "The Mutton-hole Trail is a long stretch of leads running to the tower of the same name, where Byron once kept his bear." Moreover, this item is prefaced by

"Lo! dusky masses steal in dubious sight
Along the leaguered wall."

Byron, Don Juan, Canto VII

However, I am told by the college archivist that this story regarding the bear is almost certainly not true. As a Noble he would have had extra posh rooms and it is thought that he probably lived in I1 Nevile's Court. Posh but cold. But the bear was real and probably lived in Rams yard where Byron kept his horses.

It is a mystery to me as to why Wright wrote a fanciful story about the bear. Most of his book seems truthful, it contains descriptions of examination papers and advice to parents on which college to choose for their son. Maybe Byron kept the bear temporarily in that turret until it was moved. Some details of Wright and his career will be found in *Mr Hopkins' Men*, by Alex Craik. Mr Hopkins crammed men for the Maths Tripos including many Senior Wranglers. A wrangler is someone who obtains a First in the Maths Tripos and a Senior Wrangler is one who is top of the list in that year. Mr Hopkins' successes included G G Stokes (fluid dynamics) and Arthur Cayley (matrices) who were Senior Wranglers in 1841 and 1842 respectively; also J J Sylvester (2nd in 1837). Another was J W Colenso (2nd in 1836) who became Bishop of Natal.

Anyway, it seems that Wright broke some regulation (perhaps due to being gored by a bull) that prevented him from taking the Maths Tripos although he was a good mathematician and it was likely he would have been a high wrangler.

One curious loose end is that the turrets in Great Court are not hexagonal at all. They appear to be octagonal but a close inspection from above shows that the internal corner is in fact a right angle so there are only seven sides and thus the turret is an irregular heptagon.

Byron seems to have achieved little academically while in Cambridge but wrote some poetry and generally had a good social life. He did many things that undergraduates do, he suffered being thrown into the fountain which is a fate that befalls many (it's a bit cold but not too deep as I remember).

Regarding pets it is interesting to note that Lady Butler, the Master's wife around 1970 had a pet. It looked remarkably like a dog and made barking noises much like a dog. Nevertheless it was classified as a cat and thus permitted!

Years ago, at the assizes, the judge used to stay in A1 in Great Court which is adjacent to the Master's lodge. This is a magnificent room with a glorious bed with silken back embroidered with the letters VR (Queen Victoria slept here). Some years ago it was available as a guest room. Several old chums hired it (in about 1970) for a nostalgic weekend. After a jolly evening they were playing draughts (checkers) by jumping from square to square of the patterned carpet. This disturbed Lady Butler in the Lodge above who came down in her nightgown via a secret door to investigate.

2 Babbage

Charles Babbage was also an undergraduate at Trinity. He came up in 1810 but transferred for some reason to Peterhouse in 1812. Babbage also did not take the Tripos exam for other reasons. In those days, the exam started with a viva with the student sitting on a stool while being asked questions by the Examiner. It is reported that Babbage was unpleasant and maybe blasphemous and was not allowed to sit the examination. He was given an ordinary degree in 1814.

It is conjectured that maybe Babbage deliberately had himself rejected since he did not want to take the exam for fear of being beaten by Herschel. Indeed Herschel was Senior Wrangler in 1813 and the second wrangler was Peacock. John Herschel was later involved in the discovery of the planet Neptune. George Peacock later became Dean of Ely and supervised the restoration of the cathedral.

Babbage was grumpy concerning the educational state of affairs at the time and formed the Analytical Society with Herschel and Peacock while they were still undergraduates. This society was eventually instrumental in stimulating modernization. As we know, Babbage was grumpy with the government many years later regarding the funding of his Analytical Engine.

Babbage was appointed Lucasian professor of mathematics (the same chair that Newton held several centuries earlier) from 1828 to 1839 but never lectured. Incidentally, Newton lived in E staircase in Great Court.

3 Babbage and Brunel

Babbage was a consultant to Brunel during the construction of the Great Western Railway (GWR often known as God's Wonderful Railway). I gather that he was instrumental in helping Brunel over the matter of the gauge which was set at 7 feet as opposed to the standard gauge of 4 feet and 8½ inches as in Roman chariots and used by most railways. The advantage of the broad gauge is that it permits higher speeds and greater comfort. Moreover, the line from London to Bristol is very level and straight and trains did go like the clappers for the time. Indeed, when the HST diesel trains were introduced on that line in 1975, they were the fastest passenger trains in the world outside Japan and ran at 200kph (125 mph).

Sadly, the force of standardization saw the broad gauge replaced in 1892 by standard gauge throughout. It is interesting to note that the first London Underground railway from Paddington to Farringdon was originally broad gauge and steam hauled from 1863.

As a senior consultant, Babbage was entitled to a company train. These days a senior consultant might have a company car. In those days one might have expected a company horse. But Babbage had rights to a company train!

In 1838, the railway only went from Paddington to Maidenhead. A scary event is described in *Red for Danger* by L T C Rolt and in Vol 1 of *History of the Great Western Railway* by E T MacDermot. Briefly, one Sunday morning Babbage arrives at Paddington and demands his train and is told that nothing else is about so he can use either line. He is just about to set out when Brunel arrives unexpectedly in his own special train that he has taken from Maidenhead. Imagine the disaster if they had met and been on the same line.

The story would have been more dramatic if at night and Babbage was going to Maidenhead to meet a lady at the then notorious Skindles hotel. And we can imagine Brunel having galloped into Maidenhead sweaty from surveying the Sonning cutting and eager to go to his London club for a late dinner. They might have seen each other approaching in the darkness at a closing speed of around 100 mph and prayed they were not on the same track.

Indeed, if they had crashed and Babbage had died that fateful day quite early in his collaboration with Ada Lovelace, then there would have been no analytical engine, it would have been the end of working with Ada Lovelace, our language would not be called Ada and the symposium would not have been held and so you would not be reading this paper.

This incident and others laid the thought of the need for safety on the railways through signalling. Even today, railways are one of the few industries who seem to care about software correctness. Another is avionics.

And now I will turn at last to the matter of events leading to the beautiful Ada language. There were initially two threads of activity, one in Europe and one in the US.

4 LTPL-E

In the mid 1970s, a number of different programming languages were in use in Europe for process control and similar embedded system applications. They included Coral 66 from the UK Ministry of Defence, RTL/2 from Imperial Chemical Industries, LTR in France (RTL backwards), and Pearl in Germany. The European Commission felt that it would be a good idea if the same language could be used throughout Europe and so supported many meetings aimed at defining the basis for a new language. This was perhaps the first stirrings of the objective of ever closer union.

Most meetings were in Brussels and the attendees enjoyed excellent lunches helping to reduce the wine lake and beef mountain which were a problem at the time. The meetings were helpful in identifying the requirements for a successor language.

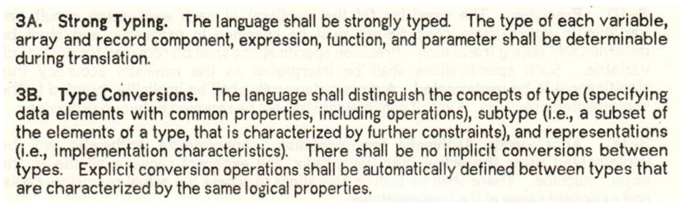
Eventually, the post of chief designer was advertised in the Sunday Times to lead such a development known as LTPL-E (Long Term Procedural Language – Europe). It is said that all those who attended interviews to lead LTPL-E advised that Europe should join with the US in a common development. And indeed the two efforts were merged.

5 The HOL project

In the United States they too thought that there were too many languages. Examples included the Air Force's Jovial (Jules Own Version of the International Algorithmic Language which was based on Algol 58), the Navy's CMS-2 and the Army's Tacpol to name but a few.

Accordingly, the High Order Language project was established under the leadership of Col. William Whitaker. The management team included Philip Wetherall of RSRE (the Royal Signals and Radar Establishment) at Great Malvern in England and David Fisher of IDA (Institute for Defense Analyses) in the US. The first task of the project was to decide what it was all for, that is to define the Requirements. That was an excellent idea – too many projects bash ahead doing something without firmly knowing what it is all for.

So requirements documents emerged and were refined after much deliberation. They were called Strawman, Woodman, Tinman, Ironman and finally Steelman. A sample of Steelman is shown in Figure 3. Note the relatively abstract level of the requirements. These documents are available at http://iment.com/maida/computer/redref/index.htm.



3A. Strong Typing. The language shall be strongly typed. The type of each variable, array and record component, expression, function, and parameter shall be determinable during translation.

3B. Type Conversions. The language shall distinguish the concepts of type (specifying data elements with common properties, including operations), subtype (i.e., a subset of the elements of a type, that is characterized by further constraints), and representations (i.e., implementation characteristics). There shall be no implicit conversions between types. Explicit conversion operations shall be automatically defined between types that are characterized by the same logical properties.

Figure 3 A sample of Steelman

6 The competition

Four contracts were let for initial designs. They were colour-coded to preserve anonymity to ensure unbiased evaluations. They were as follows:

- Green Honeywell, notionally in Minneapolis but the work was really done at CII-Honeywell-Bull in Versailles. The leader was Jean Ichbiah (now sadly deceased).

- Red Intermetrics in Boston. The leader was Ben Brosgol who is now with AdaCore.
- Blue Softech also in Boston. The leader was John Goodenough who has recently retired (he tells me) after many years at the Software Engineering Institute in Pittsburgh.
- Yellow SRI in Silicon Valley. The leader was Jay Spitzen and he was assisted by many academic consultants.

The four initial drafts are shown in Figure 4. As you can see they are completely anonymous.

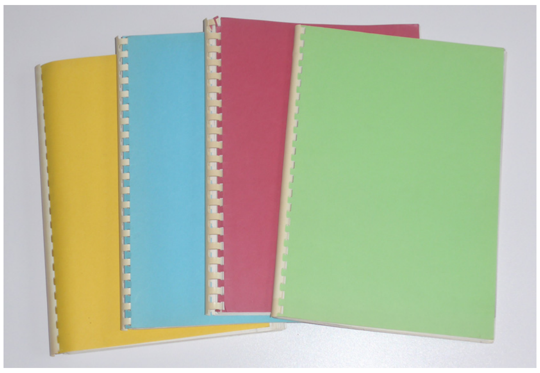


Figure 4 The colourful initial drafts

After one year Blue and Yellow were eliminated. Blue was interesting but considered somewhat strange. Yellow was rejected largely because it failed to meet the requirements.

Green and Red were then given another year to refine their designs. Red somewhat changed direction and leader and was considered overly ambitious whereas Green consolidated its position.

So Green was acclaimed the winner in 1979.

7 The name

The project had gone to plan apart from one vital thing and that was that choosing a name for the new language had not happened.

Eventually, in a wine bar in Paris, a group of management team members chose the name Ada. I understand that the reasons were roughly as follows:

The Pascal language (one of the baselines for Ada) was named after the famous French mathematician Blaise Pascal (1623–1662) well known for his triangle (of binomial coefficients) and his theorem about a hexagon inscribed in a conic. So a good idea to name the new language after a person.

They wanted to honour a woman. Grace Hopper had done much for COBOL.

Ada Lovelace was clearly the world's first programmer. So the name Ada was proposed. Permission was sought from her descendant, the 4th Earl of Lytton. Philip Wetherall from the MoD wrote to the Earl on 10th October 1978. The Earl replied on the 18th to say Yes and observed

that ADA was at the heart of RADAR

But do remember that we always write Ada and not ADA which is the American Dental Association.

The language community were delighted to have Ada as their mascot. Pictures of Ada sprung up on books and documents. And statuettes of Ada continue to be awarded to those making valued contributions to the cause. See Figure 5.

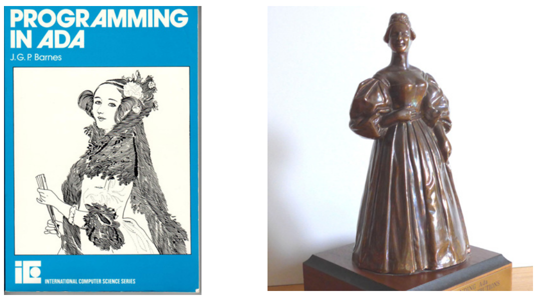


Figure 5 Images of Ada

Many exciting conferences have been held and an especially good one was in London in 1997 when the 5th Earl of Lytton was guest of honour. It was his father that gave permission to use the name Ada.

A particular feature of this conference was entertainment from the New York Village Opera Group. This took the form of a play entitled *The Maiden and the Mandate* which showed the conflict between Lady Ada who wrote excellent software in Ada and the treacherous Senior Hacker who wrote in C. It was based on Trial by Jury by Gilbert and Sullivan. Lady Ada was played by Karen Mason (née Leah). The lovely lady is shown in Figure 6.

Figure 6 Lady Ada

8 The standard

Ada was proclaimed as MIL-STD 1815 on 10th December 1980, the 165th anniversary of the birth of Ada Lovelace in 1815 (note the standard number) at an ACM SIGPLAN conference in Boston. Gosh that's 35 years ago. After a certain amount of polishing Ada became an ANSI standard in 1983 and an ISO standard in 1987.

For a detailed description of the evolution of the project from requirements to international standard including the correspondence with the 4th Earl of Lytton see the paper *Ada – The Project* by William Whitaker in SIGPLAN Notices.

9 Ada now

We all crave freedom. But freedom takes two forms. There is freedom from problems on the one hand and freedom to do whatever you want on the other. These two freedoms clash. Ada aims to provide freedom from problems by detecting difficulties early in the development of software.

Ada is mainly used for software that matters in areas such as avionics, space, and railways. Many applications are somewhat confidential but I can mention iFACTS, part of the Air Traffic Control system now in use over the London area. There is a demonstration in the computer museum at Bletchley. And I am pleased to say that my daughter Janet was one of the system architects.

iFACTS is written in Ada and SPARK. Most readers will be aware of SPARK which is a subset of Ada with additional contracts (historically called annotations) that is supported by static analysis and proof tools. It has its origins long ago in work done at RSRE in the 1970s by Bob Philips and sponsored by a requirements board chaired by Dame Steve Shirley. Bob Philips later worked with Bernard Carré and they created SPADE which later became SPARK.

Ada 2012 incorporates contracts and SPARK 2014 is now integrated into the GNAT Ada toolset. The goal is to show that a program is correct through the use of contracts and formal static proof. Testing can only show the presence of errors and not their absence.

It is pleasing to note that both Dame Steve Shirley and the Earl of Lytton were at the banquet held in Balliol College as part of the symposium celebrations.

10 A sad note

This lecture ends on a sad note. Some years ago when at a conference in Paris, a member of the HOL team said he would take me to the elegant wine bar on the Champs Elysées where the name Ada was chosen and we would celebrate with champagne. But ... alas it was now a Burger King!

Acknowledgements

I have consulted many people in preparing this talk and I am very grateful for their help. I fear that I am almost sure to have omitted someone but I must thank (in alphabetical order) Jamie Ayre, Sarah Baldwin, Janet Barnes, Susan Bond, Ben Brosgol, Bill Carlson, Larry Druffel, Terry Froggatt, John Goodenough, Ursula Martin, Karen Mason, Nick Neve, Ed Schonberg, Jonathan Smith, Doron Swade, Tucker Taft, Betty Toole, John Webb.

Bibliography

The following are referenced in the text.

Anon (1960), *The Night Climber's Guide to Trinity*, 3rd edition, Wetherhead, Cambridge.

Alex D. D. Craik (2007), *Mr Hopkin's Men*, Springer.

E. T. MacDermot, revised by C. R. Clinker (1964), *History of the Great Western Railway*, Vol 1, revised edition, Ian Allen.

L.T.C. Rolt (1966), *Red for Danger*, David and Charles.

William A. Whitaker (1993), *Ada – The Project* in ACM SIGPLAN Notices, Vol 28, No 3.

John M. F. Wright (1827), *Alma Mater, or Seven Years at the University of Cambridge, by a Trinity Man*, 2 Vols, Black, Young and Young.

Deriving Reusable Process-based Arguments from Process Models in the Context of Railway Safety Standards

Barbara Gallina
Mälardalen University, P.O. Box 883, SE- 721 23 Västerås, Sweden; email: barbara.gallina@mdh.se
Luciana Provenzano
Bombardier Transportation, Östra Ringvägen 2, 722 14 Västerås, Sweden; email: luciana.provenzano@rail.bombardier.com

Abstract

In the railway domain, standards such as the EN5012x family prescribe processes to be followed for the management and certification of safety-critical systems. This results in a need to model processes and retrieve process-based arguments to prove that the system achieved the required safety level in order to reduce time and cost spent in the certification process. In this paper, we present the application of the MDSafeCer, i.e. a model-driven safety certification method, for railways. In particular, we model in SPEM 2.0 the safety requirements process according to what described in the safety plan, and we show how it is possible to extract safety evidence to prove the compliance of this process to the EN50128 standard.

Keywords: railway, safety certification, process modelling

1 Introduction

In the context of safety-critical railway systems engineering, various standards (i.e. EN5012x) play a crucial role in prescribing process reference models at system (i.e. EN50126 [5]) as well as at sub-system level (i.e. EN50128 [6]). These models define sets of partially ordered tasks that have to be executed to develop safety-critical railway systems (such as entire vehicle, signalling components, etc.). As also observed in the automotive domain [4], to these partially ordered tasks other core process elements are directly or indirectly associated namely roles, work-products, and guidelines. These core process elements allow process engineers to establish responsibilities by defining roles (who) for producing specified work products (what). Moreover, for the execution of the tasks well-defined principles and techniques supported by guidelines are applied. The rigor and stringency required during the application of these reference models vary with respect to the criticality of the systems, and are subject to interpretations. Compliance with the process reference models constitutes a mandatory requirement for certification purposes in which process-related deliverables are fundamental. Within EN50129 [7], a safety case is defined as a structured justification document that includes the required evidence, i.e. evidence of quality management, evidence of safety management (compliance with the EN50126 RAMS process [5]), and evidence of functional and technical safety. Evidence of quality as well as safety management represents process-related evidence. The provision of such evidence is time-consuming and costly, especially if reuse [3] and semi-automatic generation is not enabled.

To reduce time and cost, we apply MDSafeCer, which was introduced by Gallina [2] in the context of the SafeCer project [1] SYNOPSIS [14]. MDSafeCer is a model-driven certification method for the (semi) automatic generation of process-related deliverables. In this paper we consider a portion of the safety plan, we model it in SPEM (Software Process Engineering Meta-model) 2.0 [11], and then we show how process-based fragments in form of GSN (Goal Structuring Notation)-compliant goal structures [9] of a safety case can be derived from the safety plan model.

The remainder of this paper is organized as follows. Essential background information is recalled in Section 2. The application of MDSafeCer to the safety-requirements process defined within a railway project is described in Section 3. Concluding remarks and perspective for future work are presented in Section 4.

2 Background

In this section, we shortly recall some background on which this work is based. In particular, in Section 2.1, we provide a quick survey of the CENELEC EN5012x family of European standards applicable for the management and certification of safety-critical railway systems. In section 2.2 we recall the main SPEM 2.0 process elements that will be used further in this paper to model the safety requirements process. In section 2.3, we briefly introduce the GSN graphical notation used to build the safety case fragment. Finally, in section 2.4 we introduce MDSafeCer method that we will apply in the railway domain.

2.1 EN5012x standards

The European group of standards EN5012x defines processes that enable the implementation of a consistent approach for the management of safety-critical railway systems. The three main standards are:

- EN50126, which describes a process for the specification and demonstration of the RAMS

(Reliability, Availability, Maintainability, Safety) requirements [5]

- EN50129, which defines a process for safety acceptance and approval [7]
- EN50128, which focuses on processes for the development, deployment and maintenance of safety-related software for railway control and protection applications [6].

Figure shows the scope of the above-mentioned standards compared to the railway product or system under development and/or maintenance.

In order to obtain the safety approval for a given safety-critical railway system or product, the EN50129 standard prescribes that an independent safety assessment is performed based on documentary evidence. The documentary evidence includes the so-called Safety Case, i.e. the documented demonstration that the product complies with the specified safety requirements [7]. The Safety Case addresses the conditions that shall be satisfied to prove that the necessary level of safety has been achieved, i.e. evidence of quality management, evidence of safety management, and evidence of functional and technical safety.

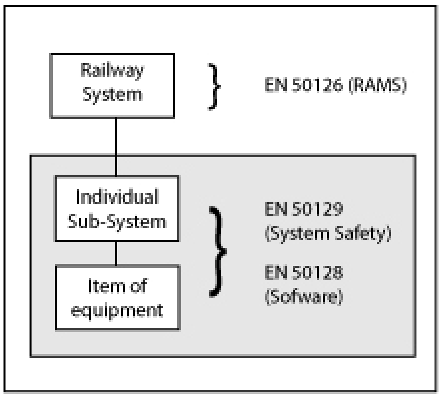


Figure 1 Scope of the EN5012x standards

The Software Requirement Phase is part of the life-cycle model (Figure 2) required by the Software Quality Assurance activities described in chapter 6.5.4.5 of the EN50128 standard [6]. In particular, the standard states that quality concerning the life-cycle model shall address as a minimum:

- activities and elementary tasks consistent with the plans, e.g. Safety Plan, that have been established at the System level;
- entry and exit criteria of each activity;
- inputs and outputs of each activity;
- major quality activities;
- the entity responsible for each activity

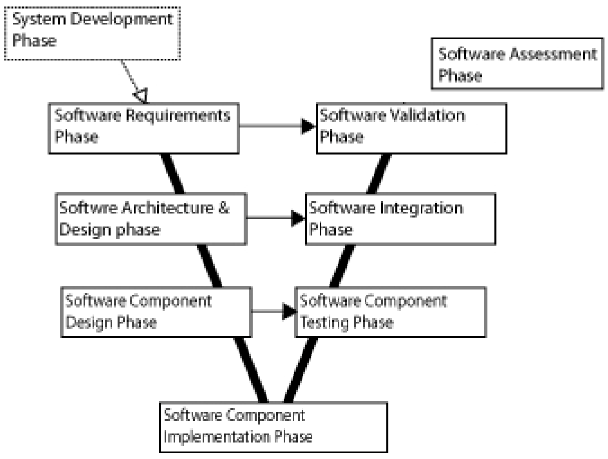

System Development Phase
Software Assessment Phase
Software Requirements Phase
Software Validation Phase
Softwre Architecture & Design phase
Software Integration Phase
Software Component Design Phase
Software Component Testing Phase
Software Component Implementation Phase

Figure 2 Life-cycle phases for a development project extracted from the V-model defined in the EN50128

Moreover, section 7.2 “Software Requirements” of the same standard defines the artifacts that shall be produced at the end of the Software Requirements Phase, i.e.:

- Software Requirements Specification
- Overall Software Test Specification
- Software Requirements Specification Report

By reading these recommendations, it is clear that a process shall be defined which complies with the standard.

2.2 Process modelling through SPEM 2.0

SPEM 2.0 [11] is the OMG standard for systems and software process modelling. Despite it is a general-purpose language, its elements implicitly enable to model safety concerns, as explained by Gallina et al. in [3] and [4].

The following table (Table 1) shows a subset of SPEM 2.0 modelling elements, in particular the ones we will use in Section 3 to model task, roles, guidance, tools and work-products related to the safety requirements process.

Table 1 Icons denoting method content (use) elements

Task	TaskUse	Role	WorkProduct	Tool	Guidance
🏷	🔄	👤	📄	🔨	📐

2.3 Safety case documentation

As summarized by Dardar et al. [12], a safety case can be documented in textual or graphical languages (refer to [8]). GSN [9] is a graphical notation that allows organizing the safety argumentation into flat or hierarchically nested graphs called goal structures. Figure 1 shows the syntax of the core GSN modelling elements that we will use in Section 3.

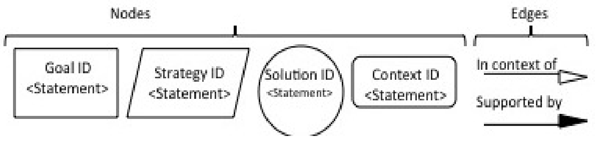


Figure 1 Subset of GSN concrete syntax

SACM (Structured Assurance Case Metamodel) [13] is an OMG standard that represents an effort to unify and standardize the graphical notations, namely GSN and CAE (Claim Argument Evidence) [10], broadly used for documenting safety cases. By providing a meta-model that defines the abstract syntax of a unified argumentation language, SACM thus constitutes a step towards the formalization of these notations.

2.4 Model-Driven Safety Certification

In this subsection we recall essential information on the Model-Driven Safety Certification (MDSafeCer) method [2]. MDSafeCer allows the (semi) automatic generation of process-based evidence from process models. MDSafeCer consists of three iterative tasks in succession, as shown in Figure 2.

The main idea is that a process is modelled (refer to the first task "Safety-process modelling") according to the best practices and the applicable standards. Once the process model is ready, a process-based argument can be generated (refer to the second task "Process-based argument generation") via a model to model transformation. The generated argument may need to be rectified, resulting in iterations back to the previous tasks, and/or completed by a safety argumentation expert (refer to the third task "Process-based argument Check&Completition") [2].

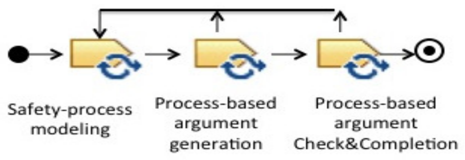

Safety-process modeling
Process-based argument generation
Process-based argument Check&Completion

Figure 2 MDSafeCer overview specified in SPEM 2.0

3 EN50128-compliance via MDSafeCer

In this section, we apply MDSafeCer in the context of safety-critical railway systems to:

- Model the safety requirements process
- Build evidences required to prove the compliance of this process with what described in chapter 7.2 "Software Requirements" of the EN50128:2011 railway standard. These evidences will compose the Safety Case, as explained in Section 2.1.

In particular, we model the process of building the Software Requirements Specification artifact within the Software Requirements Phase.

3.1 Safety-requirements process modelling

The first step is to model in SPEM 2.0 the task concerning the writing of the Software Requirements Specification document (refer to task "Safety-process modeling" in Section 2.4). This task shall be compliant with chapter 7.2 of the EN50128 standard.

To perform this activity, all process elements linked to this task shall be specified, as required by the SPEM 2.0 process elements recalled in Section 2.2. For example, we shall define the work-product associated to this task, the role in charge of creating and maintaining this document, and so on. This information should be described in the Project Safety Plan and/or in other project plans, such as the Quality Assurance plan, etc. that are referred in the Safety Plan. The following table (Table 2) shows the definition of the process elements related to the safety requirements process and in which project plan we find the needed information.

Table 2 Process elements description

SPEM2.0 Process element	Process element description	Information found in…
Work product	Software Requirements Specification	Sub-chapter "Safety life-cycle" of chapter "Safety Management" within the Safety Plan
Role	Requirement Manager	Engineering Project Plan (EPP) that is referenced in sub-chapter "Roles and Responsibilities" of chapter "Safety Management" within the Safety Plan
Tool	IBM DOORS	Sub-chapter "Safety Requirements" of chapter "Safety Management" within the Safety Plan.
Guidance	Software Safety Requirement Guidelines	Requirement Management Plan that is referenced in sub-chapter "Safety Requirements" of chapter "Safety Management" within the Safety Plan
Task	Software Requirements Specification	Sub-chapter "Safety life-cycle" of chapter "Safety Management" within the Safety Plan

Figure 3 depicts the final result of the modeling in SPEM 2.0 of the task Software Requirements Specification.

It is worth noting that SPEM 2.0 also enables the process engineer to define via stereotypes some additional information for each process element (e.g., <<performs, primary>>). This makes possible the addition of important pieces of information, necessary to support the safety justification.

Moreover, in the case of the process element "role", it is possible to specify that the Requirements Manager's competence is substantiated through CV and course attendance certificates. These pieces of information are then included in the final justification, as shown in Section 3.2.

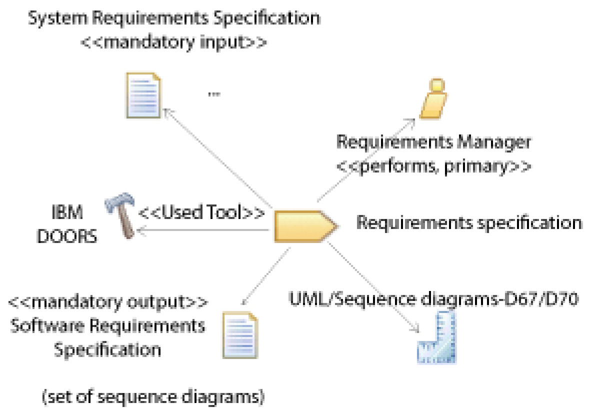

System Requirements Specification
<<mandatory input>>
...
Requirements Manager
<<performs, primary>>
IBM DOORS
<<Used Tool>>
Requirements specification
<<mandatory output>>
Software Requirements Specification
UML/Sequence diagrams-D67/D70
(set of sequence diagrams)

Figure 3 SPEM 2.0 modeling of Software Requirements Specification in EN50128

The same logic used to model in SPEM 2.0 the Software Requirements Specification can be applied to model the remaining work-products within the Software Requirements Phase, i.e. the Overall Software Test Specification and the Software Requirement Specification Report.

3.2 Process-based argument compliance

Based on the information defined in the model in Figure 3 and by applying the transformation rules [2], we can create the safety argumentation (refer to task "Process-based argument generation" in Section 2.4) in the GSN notation, as depicted in Figure 4.

As discussed in [8], the documentation style is a matter of taste and inclination. Text-inclined safety experts/assessors might prefer reviewing textual documentation. To satisfy text-inclined argument-readers, instead of a model to model transformation, a model to text transformation should be provided aiming at generating a safety justification in the shape of a structured prose, as shown in this example:

"This argument establishes the following claim: the task requirement specification has been planned, within the context of EN50128. To establish the top-level claim, four strategies are adopted: (1) argues about roles; (2) argues about work-products; (3) argues about guidelines; (4) argues about tools.

To argue about roles, one sub-claim is established: (1) the requirement manager is certified. This sub-claim is supported by direct evidence in form of CV and course attendance. Etc..."

The above-written text-based argumentation is equivalent to the one given in GSN.

Once the argumentation is available, it is used by a safety expert (refer to task "Process-based argument Check&Completion" in Section 2.4) as basis for creating the final document to be submitted to the authority. The safety expert may improve the confidence of an argument by adding more assumptions and justifications, modify existing goals or develop new goals, as explained in [2].

Once the safety argumentation is entirely checked and finalised, it can be used to prove the compliance of the safety-requirements process with the EN50128 standard and, as a result, included in the Safety Case.

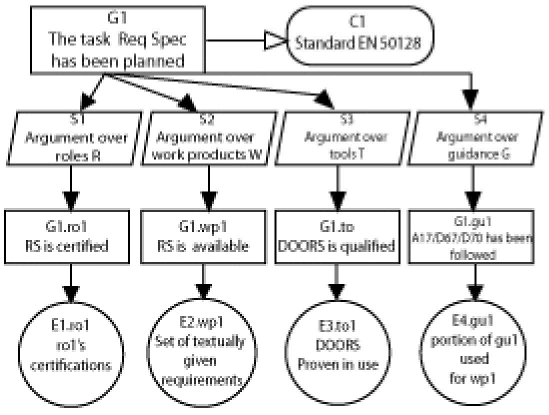


Figure 4 GSN structure arguing about process compliance

4 Conclusions and future work

In this paper, we presented the application of MDSafeCer to railway standards.

In railways, safety standards (EN5012x) prescribe processes to be followed for the management and certification of safety. This requires the definition of well-defined processes and their application and monitoring throughout the entire project life-cycle. Moreover, process-based safety arguments shall be extracted to show process compliance. These activities can be time-consuming, expensive and error-prone if not supported by a structured process modelling and systematic reuse. For these reasons, we explored the possibility of applying MDSafeCer to model the safety requirements process and to build the process-based argument needed to show the compliance of this process with the EN50128 standard.

The first result we obtained is that MDSafeCer can be successfully used for this purpose. By drawing generalizations from this result, we can conclude that MDSafeCer can be applied to the whole life-cycle defined in EN50128. The proposed usage of SPEM2.0 resulted to be promising. SPEM2.0 can be used to model the whole life-cycle defined in EN50128 in a rather intuitive way. This outcome is also valid for all other safety railway standards.

We also observed that the use of MDSafeCer can significantly improve the process quality at very early stage of the project. In fact, MDSafeCer enables to highlight missing information about work-products, roles, responsibilities, etc. by giving an opportunity of tuning the process in the right way. To generate the argument-fragment, MDSafeCer needs to transform process elements into argumentation elements. Whenever process elements are missing MDSafeCer is expected to notify the user. From this perspective we think that MDSafeCer will reduce time and cost for the production of the safety evidence.

In the future, in cooperation with assessors, we plan to fully define a pattern for arguing about process compliance in the context of railways standards. Moreover, we also expect to automate the generation of the argument by using the prototype tool support currently available within the AIT WEFACT tool [15].

Acknowledgements

This work has been partially supported by the Swedish SSF SYNOPSIS project [14].

References

[1] ARTEMIS-JU-269265, *SafeCer - Safety Certification of Software-Intensive Systems with Reusable Components*.

[2] B. Gallina (2014), *A Model-driven Safety Certification Method for Process Compliance, 2nd IEEE International Workshop on Assurance Cases for Software-intensive Systems (ASSURE)*, joint event of ISSRE, Naples, Italy.

[3] B. Gallina, I. Sljivo, O. Jaradat (2012), *Towards a Safety-oriented Process Line for Enabling Reuse in Safety Critical Systems Development and Certification*, Post-proceedings of the 35th IEEE Software Engineering Workshop (SEW-35).

[4] B. Gallina, S. Kashiyarandi, H. Martin and R. Bramberger (2014), *Modelling a Safety- and Automotive-oriented Process Line to Enable Reuse and Flexible Process Derivation*, Proceedings of the 8th IEEE International Workshop on Quality-Oriented Reuse of Software (QUORS), joint workshop at COMPSAC conference, IEEE Computer Society, doi: 10.1109/COMPSACW.2014.84, pp. 504-509,Västerås (Sweden).

[5] BS EN50126-1 (1999), *Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)*

[6] BS EN50128 (2011), *Railway applications – Communication, signalling and processing systems – Software for railway control and protection systems*

[7] BS EN50129 (2003), *Railway applications – Communication, signalling and processing systems – Safety related electronic systems for signalling*

[8] C. Holloway (2008), *Safety case notations: Alternatives for the non-graphically inclined?*, Proceedings of the 3rd IET International Conference on System Safety, IET Press, pp. 1-6.

[9] GSN (2011), *Community Standard Version 1.*

[10] L. Emmet and G. Cleland (2002), *Graphical notations, narratives and persuasion: A pliant systems approach to hypertext tool design*, in Proceedings of the Thirteenth ACM Conference on Hypertext and Hypermedia, ser. HYPERTEXT '02. New York, NY, USA: ACM, pp. 55–64.

[11] Object Management Group (2008), *Software and Systems Process Engineering Meta-Model (SPEM)*, v2.0. Full Specification formal.

[12] R. Dardar, B. Gallina, A. Johnsen, K. Lundqvist, M. Nyberg (2012), *Industrial Experiences of Building a Safety Case in Compliance with ISO 26262*, Proceedings of the 2nd IEEE WoSoCER, joint event of the 23rd International Symposium on Software Reliability (ISSRE), Dallas (Texas), IEEE Computer Society, ISBN 978-1-4673-5048-8, USA.

[13] SACM, *http://www.omg.org/spec/sacm/1.0.*

[14] SYNOPSIS-SSF-RIT10-0070: *Safety Analysis for Predictable Software Intensive Systems,* Swedish Foundation for Strategic Research.

[15] *WEFACT: Workflow Engine for Analysis, Certification and Test*, http://www.ait.ac.at/research-services/research-services-digital-safety-security/verification-and-validation/methods-and-tools/wefact-workflow-engine-for-analysis-certification-and-test/?L=1

Proceedings

Workshop

Challenges and New Approaches for Dependable and Cyber-Physical System Engineering (De-CPS 2015)

Ada-Europe 2015

22 June 2015

Madrid, Spain

Sponsors

2nd Workshop on Challenges and New Approaches for Dependable and Cyber-Physical System Engineering (De-CPS 2015) Editorial

Daniela Cancila
CEA, LIST, CEA Saclay - F91191 Gif-sur-Yvette Cedex - France; email: daniela.cancila@cea.fr
Charles Robinson
THALES R&T, 1 Av. Augustin Fresnel 91767 Palaiseau Cedex - France; email: charles.robinson@thalesgroup.com

The scientific view underlying De-CPS

In June 2015, we organized the second iteration of the workshop 'Challenges and new Approaches for dependable and Cyber-Physical Systems engineering' (De-CPS) [1], as satellite event of the 20th International Conference on Reliable Software Technologies – Ada-Europe 2015.

From the USA to Europe, there is a crescendo of industrial and research interest in managing the complexity of Cyber-Physical Systems (CPS). One distinguishing trait of CPS is that they integrate software control and decision making with signals from an uncertain and dynamic environment. CPS often involve heterogeneous and hierarchical systems, and their design makes extensive use of models. The Horizon 2020 program framework of the European Union devotes considerable attention in the current work program to various challenges associated with developing, integrating and providing assurance concerning CPS.

The workshop gathers together industrial practitioners and researchers concerned with dependable and Cyber-Physical Systems engineering, and use the momentum provided by the 20th International Conference on Reliable Software Technologies to foster further collaborative initiatives.

We encourage equal representation of gender in research and innovation. Showing a strong representation by female scientists in our team will hopefully inspire a new generation of women to pursue an interest in science as a career. Some of the expected impact of the H2020 are:

- Increase the participation of women in research, improve their careers and achieve gender balance in decision making,
- Increase the scientific quality and societal relevance of produced knowledge, technologies and innovations by integrating the human factors of both men and women.

It also contributes to the production of goods and services better suited to potential markets. The topics addressed by the workshop include the following:

[1] www.ada-europe.org/conference2015/De-CPS

- Industrial challenges and experience reports on co-engineering for multiple dependability concerns in CPS engineering.
- Modelling and analysis of Cyber-Physical Systems (CPS) via contract-based approaches.
- Tools and methodologies to guarantee safety-related properties, including real-time and mixed-criticality cohabitation.
- Challenges posed for CPS design and safety verification by multi-core processors.

Several European projects accepted to contribute to the success of the workshop:

- The ITEA safety and security MERgE project. Multi-concerns Interactions System Engineering. The MERgE project sponsored the workshop.
- INTO-CPS H2020 Project. Integrated Tool Chain for Model-based Design of Cyber-Physical Systems Modeling methodologies for Cyber-Physical Systems: Research field study on inherent and future challenges.
- FSF (Safe and Reliable Embedded systems). The importance of a contract-based design to fix dependability.
- U-TEST H2020 project. Testing Cyber-Physical Systems under Realistic and Unknown Uncertainty by Combining Model and Search-Based Approaches.
- The ECSEL CONCERTO project. Guaranteed Component Assembly with Round Trip Analysis for Energy Efficient High-integrity Multi-core Systems.
- The AXIOM H2020 project. Agile, eXtensible, fast I/O Module for the cyber-physical era.
- The PROXIMA FP7 project. Probabilistic real-time control of mixed-criticality multicore and manycore systems.

Moreover, Masumi Toyoshima from DENSO, Japan, Adam Pawlak and Janusz Jezewski, from the Institute of Electronics and the Institute of Medical Technology and Equipment ITAM have actively participated to the workshop.

Sponsor

The workshop has been partially funded by Thales, The MERgE ITEA project and the CEA, LIST.

The *Universidad Politécnica de Madrid* has provided the locals for the workshop and Ada-Europe the overall organisation.

Modeling Methodologies for Cyber-Physical Systems: Research Field Study on Inherent and Future Challenges

Imran Quadri, Alessandra Bagnato, Etienne Brosse and Andrey Sadovykh

Softeam Research & Development Division; Paris, France. Email: FirstName.LastName@softeam.fr

Abstract

This paper presents a field study about the inherent challenges involved in the design of Cyber-Physical Systems and how Model-Based Design (MBD) is currently being utilized in various research directions in order to help in the design and development of these complex systems. The paper then presents the first future directions and challenges deemed to be tackled by H2020 INTO-CPS project. The aim of INTO-CPS project is to create an integrated "tool chain" for comprehensive Model-Based Design of Cyber-Physical Systems (CPSs). The tool chain will support the multidisciplinary, collaborative modeling of CPSs from requirements, through design, down to hardware and software implementation. This will enable traceability at all stages of the development process. The paper aims at analyzing the current State-of-the-Art related to CPSs and be a basis to future extensions of the SysML standard to support CPS modeling within the INTO-CPS project.

Keywords: MBD, Cyber-Physical Systems, INTO-CPS

1 Introduction

Cyber-Physical Systems can be considered as the next general of Embedded Systems. In recent years, the growth of connected Cyber-Physical Systems (CPSs) and Internet of Things (IoT) devices has increased tremendously due to the availability of high-capacity networks (3G and 4G/LTE networks), advanced sensors (e.g. RFID, NFC, etc.), protocols (e.g. IPv6, MQTT, etc.), mobile Internet and wearable devices. This paradigm shift will accelerate in coming years to drive the next technological revolution for CPSs, where a plethora of light-weight interconnected devices will be able to interact, communicate and share vast amounts of data.

Cyber-Physical Systems and especially Model-based CPSs methodologies as an emerging area of increasing relevance require a comprehensive framework for their validation and certification. This includes both, the validation and certification of the embedded devices (sensors and actuators), as well as of the optional Cloud-based services which can take over the computation of critical aspects of the CPSs operation. This paper aims to provide an overview of the current research activities regarding Model-based methodologies for CPSs and includes the related challenges involved in developing these complex systems, along with the objective to determine the current and future directions related to CPSs. The paper will also serve as a foundation for future extensions of the SysML standard to support CPSs modeling and to the clear semantics on SysML usage that will be carried out within the INTO-CPS research project.

2 Cyber-Physical Systems

The wide number of application areas of CPSs demand design technologies able to cover various industrial domains like automotive, industrial control, medical, mobile communication, etc. Each domain has different point of views on the underlying technical and physical details. By observing the different challenges that are inherit in the design of CPSs, it is evident that CPSs need improved multi-disciplinary modeling and specification methodologies able to support static analysis, verification, simulation, performance analysis and implementation technologies [1]. To address these issues, domain specific languages have been developed to cover the design challenges of specific design domains. For example, a famous example can be found in the automotive domain: AUTOSAR (AUTomotive Open System ARchitecture) is the de-facto standard for automotive software and E/E (Electrics/Electronics) architectures [2]. It provides a basic infrastructure to assist with developing vehicular software using Atomic Software Components, running on a standardized middleware layer, called Run-Time Environment (RTE).

Furthermore, it includes the standardization of basic systems functions, enables scalability to different vehicle and platform variants and upgrades over the vehicle's lifetime. Various commercial AUTOSAR system development tools are available on the market: dSpace SystemDesk, ETAS ISOLAR-A, KPIT K-SAR, Mentor Graphics Volcano Vehicle Systems Architect, and Vector Informatik PREEVision [3] and DaVinci Developer, targeting the automotive domain specific solutions. Standards such ANSI/ISA-5.1-1984 have been used to specify CPSs by making use of process models to describe measuring and control devices. While these process models are able to describe different properties of the physical environment, they cannot adequately cover the computational architecture details.

2.1 Modeling of Cyber-Physical Systems

Model-Based Design (MBD) has been identified as a powerful design technique for CPSs [4]. In MBD, models are at heart of the design process. Specifications of system and its underlying components are defined in the form of models able to reflect the evolution of the system. These models can be used for early design analysis; can help in separation of concerns, traceability, trace generation, impact analysis, formal verification, simulation and synthesis. By making use of models, it is possible to have earlier identification of design defects instead of during the prototyping phase at a much higher cost. Additionally, automated or semi-automated processes can also help to synthesize implementations from models, such as automatic code generation and software synthesis on heterogeneous platforms [5]. However, the intrinsic heterogeneity and complexity of CPSs stresses all existing modeling languages and frameworks, and, currently, it is not possible for a single modeling language or tool to adequately address all challenges related to CPSs.

For CPSs modeling, a large number of modeling languages have been utilized to address the underlying aspects such as physical processes and requirements management. A good survey has been made covering languages and tools like Stateflow/Simulink, Modelica, Checkmate and Massaccio; by the Columbus project [6] in order to define an interchange format for CPSs. These languages enable CPSs modeling for design phases such as simulation and verification. In [7], the authors introduce a test bed for collaborative control and information acquisition for maritime applications. An abstraction language in the form of a Domain Specific Language (DSL) for implementation of mission-level controllers has been developed, termed as the Collaborative Sensing Language (CSL).

Recently, high level languages such as UML [8], SysML [9] and MARTE [10] have also been utilized for modeling these complex distributed systems. However, as stated before, none of these languages can singly address all the challenges related to CPSs modeling. UML, traditionally used for modeling of software systems, defines the syntax of model diagrams; it does not offer any specific semantics for CPS modeling. The OMG SysML standard does offer aspects such as requirements management which can be interesting for CPSs, but suffers from having many semantic alternatives, which are usually provided by tool vendors [11] and does not provide manners to define characteristics of real-time embedded systems such as non-functional constraints, and aspects like performance and energy consumption. MARTE, which is the recent OMG standard for real-time embedded systems does enable designers to define non-functional constraints, but suffers from the same pitfall of not having detailed guidelines and semantics, which can be used by system designers. In absence of concrete MARTE usage guidelines, designers can be plagued with the problem of correct utilization of the profile concepts, for example incorrect utilization of a MARTE hardware processor stereotype on a port, which while is possible in the standard, is not logical from the current hardware design point of view.

Additionally, high level languages have also been used for modeling aspects of Cloud computing, software tests or services resulting in development of CloudML [12], UTP [13] and SoaML respectively [14]. UTP can be efficiently applied to foster early testing [15] and to establish test automation by generation of executable test scripts [16]. UTP has been applied to various industrial and research case studies to increase automation in test execution and test design in various domains, such as telecommunications, enterprise services choreographies and eHealth [17].

In short, many modeling languages have been used to describe CPSs or aspects which can be used in CPSs development. However, modeling techniques that address only the software aspects are not able to accurately specify CPSs. The complexity of CPSs design demands the usage of new system models/analytical tools, and software simulation tools, along with modeling languages and appropriate learning mechanisms [18, 19] that are able to take into account aspects related to the physical processes of CPSs. In the modeling process, systems are usually considered as static entities where current or general system characteristics are used to emulate system’s behavior. Thus for the modeling of CPSs, effective semantics are still needed able to integrate any language to reap the benefit offered by MBD. Extending both SysML for CPSs modeling and MARTE for the underlying embedded devices, while integrating both modeling languages under a common, homogeneous framework able to support a holistic modeling of complex heterogeneous CPSs, is still an open problem that needs to be addressed.

2.2 Projects carrying out CPS research

We now look at some of the research related to CPSs in recent years.

The CHESS project [20] focuses on improving Model-based Design (MBD) practices and technologies to better address issues such as safety, reliability, performance, robustness and other extra-functional concerns for real-time and dependable embedded systems. The project addresses the challenges related to compositional structure, interactions and behavior of system components while guaranteeing their correctness and the level of service at run time. The tool set developed in the project supports verification of extra-functional properties of different system components. The project also proposes a multi-concern component modeling language and editor to fits multiple industrial domains. The language proposed in the project extends UML, SysML and MARTE languages.

In the CONTREX project [21], a UML/MARTE methodology for distributed, mixed-critical embedded systems has been proposed. This modeling effort is focused on extending the standards to integrate aspects related to distributed networks and mixed- criticality systems, which are not fully addressed in the standards. While CPSs modeling is not undertaken in the project, the outputs of this project can serve as a foundation for new research activities related to CPSs oriented Model-Based methodologies in the near future.

The CONCERTO project [22] proposes a methodology for enabling correct-by-construct component assembly for multicore systems. The automatic generation of virtual prototypes has been made possible, along with introduction of support for separation of concerns using a meta-model based approach. New run-time monitoring mechanisms have been developed to analysis extra-functional properties such as energy consumption. Finally, the project enables iterative development by enabling back propagation from platform-specific to platform-independent models.

The COMPASS project provides tools and techniques to support a model-based approach to developing Systems of CPSs, also called Systems of Systems (SoSs) by introducing the COMPASS Modeling Language (CML) [23]. They extend SysML by the addition of formal CML notations. COMPASS augments CPSs modeling by means of additional tools and techniques to enable informal SoS development to be undertaken under the guidance of CML analysis techniques, some of which can be presented at the SysML level. The DESTECS project [24] proposes a methodology for defining co-models allowing discrete event (DE) and continuous time (CT) models to be co-simulated. The DE and CT models are linked through a common interface specification that identifies shared (monitored/controlled) variables, design parameters and events. While the project supports co-simulation, verification is not supported.

2.3 Recent methodologies for Cloud Computing

There have also been interesting design methodologies which could help in future CPSs integrating cloud computing aspects.

The DreamCloud project [25] aims to enable dynamic resource allocation in many-core embedded and high performance systems, while ensuring appropriate guarantees on non-functional properties such as performance and energy efficiency. The project focuses on integrating embedded systems with cloud-like capabilities, in order to allow the systems to tune resource usage in a dynamic manner without sacrificing non-functional constraints. The MODAClouds project [26] aims to develop novel methods along with an open-source and IDE run-time environment for the high-level design, early prototyping, semi-automatic code generation and automatic deployment of applications on multi-Clouds with guaranteed QoS. The project aims to use Model-based Design, coupled with risk analysis to monitor applications at run-time and optimize them based on the received feedback.

2.4 Modeling of Cloud-based CPSs

Cloud-based CPSs can be seen as an upcoming new domain that aims to integrate the paradigms of Cloud computing and apply it to CPSs. This merger of the two existing domains has been termed as cyber-physical Cloud computing (CPCC) [27, 28]. Examples of CCPC involve utilization of smart phones in cloud sourcing, Unmanned aerial vehicles (UAVs) equipped with cameras/sensors for gathering data [29], and in-vehicle smart phones serving as internet-connected sensors for monitoring road conditions [30]. In [31], the authors propose a novel service provisioning model in order to make CPCC possible as a service, while in [32], a test bed for CPCC is developed (for simulation and actual prototype testing) that includes a diverse range of hosts: such as automobiles (cars, buses, etc.); people with smart phones; and unmanned aerial vehicles (UAVs).

Additionally, various factors can be seen as challenges to CPCCs. Traditional virtualization technology, one of the key underlying technologies for enabling secure and scalable Cloud computing, typically implements machine/process models which are only deterministic in nature. Thus, for the same input, repeated execution of sequential programs in these models result in the same output, but other non-functional properties such as energy consumption may vary. Therefore, virtualization technology for CPCC must develop solutions that provide deterministic behavior for non-functional properties such as execution time. Factors such as number and location of server machines, resource allocation, effective scheduling and management can also hinder the development of effective CPCCs [33].

2.5 Model-Based Repositories

In [34] the authors presented a survey of several industrial projects where research activities are being carried out related to Model-Based Repository Engineering (MBRE), and it was observed that there is currently no methodological and unified framework for MBRE, which in fact can benefit from well-defined and widely applied Model-based technologies: wide range of modeling languages, transformation techniques, and open source/commercial tools. The MBRE can be effectively used in Model-based CPSs methodologies, where large number of system components need to be handled and may require re-use and deployment.

In the MADES project [35], the authors develop a Model-Driven methodology for embedded systems that developed a component based model repository with the goal of promoting component reuse in the MADES environment for avionics domain. The repository contains knowledge gathered about components during the project design, implementation, validation and verification phases, as well as the information about the various component constraints. The repository also provides querying and persistence of previously available contents. The RAS (OMG Reusable Assets Specification) Container has been used in the project [36].

Modeling tools have also started integrating MBRE in their environments as well. The Modelio environment now offers the notions of worldwide modeling [37] where repositories are managed by means of model fragments, which can be distributed and shared between users, and impact analysis and consistency management can be carried out across the entire scope of a system. In [38] the authors illustrate the research results of collaboration with Cisco to increase the cost-effectiveness of black-box system-level testing for Video Conferencing Systems (VCSs). For this purpose, test case repositories have been developed and automatically generated test cases resulting from the research output are stored separately in two other separate repositories. In addition, two UML repositories at the product line and product levels have been built for generating test cases. Similarly in [39], research collaboration with FMC Technologies, a leading global provider for technology solutions for energy industry has resulted in development of two repositories to manage the requirements at both the product line and product levels. Requirements are specified using a restricted use case modeling approach, which are transformed into restricted textual test case specifications automatically.

2.6 Virtualization of CPSs

The current state of the virtualization technologies is the result of a convergence of several areas including processor architectures, operating systems, compilers and communication networks. Hypervisors are layers of software that exploit the features of the hardware platform to establish independent execution environments. Several virtualization solutions can be distinguished such as full virtualization, OS virtualization support and bare metal hypervisors.

Several projects are nowadays addressing multicore virtualization where time and space isolation of hardware is critical for engineering systems with mixed criticalities (e.g. safety or security). MultiPARTES [40] project is addressing multicore partitioning and introducing a novel Model-Driven approach for virtualization based on multicore partitioning. DREAMS [41] project is generalizing this approach by introducing an architectural style to broaden adoption to further domains. In a similar vein, PROXIMA [42] project is addressing virtualization from a probabilistic perspective. The focus of virtualization has shift recently from low level hardware resources such as memory and processors to communications (e.g. shared buses, networks-on-chips, etc.). The Cloud poses the next frontier to virtualization in CPSs.

2.7 Programming Cloud-based CPSs service units

Several types of software service units, such as for Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), and Data-as-a-Service (DaaS), have been available and widely used in practice. Similarly, several works have been developed for integrating IoTs/CPSs into cloud system such as in [43]. From the programming perspective, some frameworks have been developed to simplify the programming of cloud services, such as JClouds, Boto, Aneka, BOOM and OpenStack. They abstract cloud resources and support different programming models, such as MapReduce and dataflows. However, to date there is a lack of programming models/frameworks that provide a unified way and programmable APIs for executing applications in Cloud-based CPSs. Several works have presented dynamic programming capabilities of networks (e.g., software-defined networks for CPSs), infrastructure [44]. But CPSs application level programming capabilities have not been discussed, although several frameworks allow us to wrap and integrate IoT devices.

2.8 CPSs Validation and Certification

The topic of validation and certification of CPSs covers on one hand the involved numerous embedded devices (i.e. sensors and actuators), and on the other hand Cloud-based services, which would be basically responsible for aspects such as data storage and intensive CPSs related computations in the cloud. In that line of thought, research related to the topics of Cloud testing as well as Test Automation in general should be taken into account.

At first, the focus is set on traditional Testing and Test Automation. Testing and Test Automation is of key importance for the successful and qualitative engineering of networked systems and software intensive systems in general. Commonly, different types of tests can be used during the development process, such as unit testing, integration testing, and regression testing. In recent years, the focus was set on the (semi-) automated generation of test cases from various artifacts (e.g. models) that play a role in the development process. In particular, concepts such as Model-based Testing [45], Risk-based Testing [46], Data-Fuzzing [47], Behavioral-Fuzzing [48] and Evolutionary Testing [49] were researched in the past years. Model-based Testing mainly encompasses the insight of using specially annotated models, e.g. UML models including sequence and interaction diagrams or state machines, in order to automatically generate test cases by different techniques, e.g. different graph traversal algorithms. Such “test annotations” are also pushed towards standardization e.g. resulting in the OMG UML Testing Profile (UTP) [13].

Furthermore, Risk-based Testing describes capturing of different risks and threats, which can potentially hamper the operation of a particular system. Based on these risk models, test cases are automatically generated in order to verify whether the identified risks/threats and corresponding vulnerabilities are indeed/still present in the SUT (System under Test). The goal of Data-Fuzzing and Behavior-Fuzzing is to prove the robustness of a system by generating in/semi-valid input messages. Whilst sending these stimuli to the SUT, its state is monitored, in order to judge on its stability and availability in challenging situations. With respect to evolutionary testing, various optimization algorithms are applied in order to generate new test data and test sequences.

These optimization algorithms are based on biological evolutionary processes. In the scope of the so called test management, i.e. the efficient handling of test suites, test purposes and test strategies, a significant portion of research is concentrated around the topic of traceability [50], i.e. the relations between test results/verdicts, test suites/cases, and the diverse requirements which were driving the development of the system under test. This enables qualitative and quantitative statements with respect to the coverage of the generated test suites, and the quality of an SUT with respect to the initial requirements which were imposed on the SUT. With respect to testing methods widely applied in practice, the family of xUnit frameworks [51] includes a large number of unit testing frameworks such as JUnit [52], CUnit [53], and NUnit [54], to name a few examples.

In addition, integration testing is commonly performed by using technologies which have the capability of validating the overall SUT behavior, e.g. regarding various user input, or relating to the overall operation of a set of communication protocols or modules. For instance, in the domain of web application/service testing, frameworks such as Selenium [55] are intensively used. In addition, in the area of communication systems, TTCN-3 [56] has been established as a widely used description language for testing different communication protocols. Furthermore, sophisticated tools are in place that enables the usage of TTCN-3. Interesting features are given by commercial tools such as TTworkbench [57] or TestCast [58], and by freely available tools such as the Broadbit Test Tool (BTT) [59].

2.9 Simulating Cyber-Physical Systems

Co-Simulation (Co-operative Simulation) is a simulation method that permits simulating individual components using different simulation tools simultaneously and collaboratively. Individual simulation tools exchange information such as system variables and their values, time steps for synchronization, and control signals for orchestrating the co-operative simulation. Thus, engineers can use different simulation tools together to create virtual prototypes of entire Cyber-Physical Systems. In practice, however, significant challenges remain with regard to the syntax and semantics of model and system integration.

Recent effort by the MODELISAR ITEA2 project that developed a tool independent standard called the Functional Mock-up Interface (FMI) [FMI] [60] has gained significant influence, more prominently in the automotive industry. Many vendors have agreed to use FMI and now provide the facility of exporting simulations as reusable shared components. The FMI standard provides a well-defined specification and API to integrate simulation components. All simulation tools participating in the FMI co-simulation follow the defined standard and provide standardized access to model equations. This permits coupling of Continuous-Time, Discrete-Time, and Discrete-Event that are part of a Cyber-Physical System. Another key element for co-simulation via FMI is the Master Algorithm (MA) that orchestrates the steps of the co-simulation: (1) control the data exchange and (2) control time advancement among individual simulations according to the requirements of the integrated simulation of the overall Cyber Physical System.

The foundations of FMI-based co-simulation for simple models are well established [61]. Simulation tool vendors are rapidly integrating FMI export and/or import functionality as an answer to the growing demand for flexible multi-domain solutions. However, integrative solutions often suffer from restrictions and tool-specific workarounds because the tools were not designed as dedicated co-simulation frameworks.

Particular attention needs to be given to FMI's co-simulation variant due to its optimization potential in multi-domain applications (domain specific solver and integration settings, optimized concurrency) [61]. Since the FMI standard does not describe or limit the implementation of the MAs, it leaves out the two fundamental challenges of data exchange and time management. Solution for integrated data and time management in distributed simulations is technically complex and errors can easily lead to performance bottleneck and failures. This complexity pushes designers to adhere to the simplest solutions – losing much of the potential advantages of co-simulation.

3 New advances required for developing effective Cloud-based CPSs

As seen from the various researches carried out in the context of Cloud-based CPSs, one of the goals is to provide a novel Model-Based methodology that integrates the current UML standards and profiles: SysML, CloudML, UTP, SoaML and MARTE; and technologies like Model-based Repositories, in order to: 1) determine and manage requirements related to CPSs (such as interaction between physical processes and computing resources) and Cloud computing (for example, interception of traffic traversing the cloud, interpreting the data, etc.); 2) enable specification of cloud topologies via CloudML and SoaML; 3) carry out embedded systems specification via MARTE (specification of non-functional properties such as performance and energy consumption); 4) make use of Model-based Repositories to promote re-use of system artifacts; 5) support the design, analysis, construction, and documentation of the modeled artifacts to be tested via UTP; and finally 6) provide responses to issues that may arise due to the collective usage of these different standards, semantics and technologies.

For this purpose, the current UML profiles will need to be extended (for example, such as integration of CPSs related notions in MARTE) and a relevant subset from all these standards should be developed to serve as the relevant Cloud-based CPSs modeling language. While past efforts have been carried out to utilize these standards or technologies, such as usage of SysML for CPSs modeling or CloudML for Cloud computing; there has not been a cohesive effort to couple all these aspects to develop a next generation Cyber-Physical System, which is one of the main challenges related to Cloud-based CPSs modeling, for example.

Similarly, for aspects related to virtualization, we need to a) virtualize existing embedded devices by means of offering a secure access (as secure isolated communication channels) and management capabilities to CPSs physical devices; b) integrate virtualized CPSs resources and provide them as software services so that high-level programming models and applications could use them for data analysis, as well as for monitoring and controlling tasks. Such CPSs resources will expose their operational status to detect critical CPSs situations (CPU, memory, etc.) and also other functionality and data that could be utilized for tactical and operational insights and decision making. For validation and verification, we need to build on existing methods for testing, validation and certification of legacy systems. Thereby, the existing methods will need to be extended and further developed as to address the needs of Cloud-based CPSs. The overall validation framework will need to be complemented by research and development towards a protection profile for Cloud-based CPSs. In addition, the tools and tool chains need to be developed in a way as to generate (test) reports which are further usable for certification purposes, thereby reducing the effort for the certification of Cloud-based CPSs. These developments will constitute a clear progress beyond state-of-the-art given the increased need for certification frameworks for Cloud-based systems, and especially for Cloud-based CPSs. Furthermore, the progress beyond state-of-the-art should be constituted by the high degree of automation and traceability among the various involved artifacts, starting from certification rules/requirements and proceeding with Risk Analysis, Model-based Testing (based on UTP), Model-based Fuzzing, Test Execution and Automated Reporting in a form suitable for the certification authorities.

For Co-simulation aspects related to CPSs, it is required to use the strengths of different CPSs simulation environments by flexibly combining them into a co-simulation framework, thus allowing for true multidisciplinary modeling and simulation adequate to the respective domain. From our involvement in state-of-the-art industrial projects, we can deduce a number of key challenges still present to build a strong link between the models, their interoperability requirements and the simulators. For this purpose, a CPSs meta-model and UML-based design environment will be essential to express these properties in a semantically sound way.

The problem of energy efficiency for CPSs needs to be addressed at multiple levels: (1) at application-algorithm level; (2) at technology level (3) at circuit-level, avoiding worst case design, employing adaptive techniques; (4) at system level, using energy-efficient accelerators with build-in trade off QoS vs. energy and minimum required sub-systems. For reliability features, they need to be integrated in the high level modeling languages used for CPSs modeling. Many of the challenges mentioned in the paper are deemed to be addressed in the upcoming HORIZON 2020 INTO-CPS project, INTO-CPS (Integrated Tool Chain for Model-based Design of Cyber-Physical Systems, http://into-cps.au.dk/).

The aim of the INTO-CPS project is to create an integrated tool chain for comprehensive model-based design of Cyber-Physical Systems (CPSs). The tool chain will support the multidisciplinary, collaborative modeling of CPSs from requirements, down to actual implementation both in hardware and software.

INTO-CPS is a currently running project will support the Model-Based Design, enabling modeling of CPSs, that will permit building and analysis of high level system models that would otherwise not be possible using standalone currently available tools. The solution will be advancing the current State-of-the-Art and will revolve centrally around FMI-compatible co-simulation. In this context, in order to support diagrammatic multi-modeling, we will identify and extend a subset of SysML to produce a profile for FMI. The continuous aspects can be modelled in SysML by means of blocks, parametric diagrams, ports and flows. However, the expression language used in SysML parametric diagrams must be fixed prior to formalization. Once the SysML profile for FMI co-modelling will be identified and extended, a semantic mapping to FMI will be defined, and this will require formal foundations that, for instance, allow the primitive types of SysML to be mapped to suitable types in FMI, and handles mismatches between concepts like time and synchronization. This will result in model transformations to automatically map SysML diagrams to FMI contracts.

The INTO-CPS project tool chain will provide powerful analysis techniques for CPSs, including generation and static checking of FMI interfaces; model checking; Hardware-in-the-Loop (HiL) and Software-in-the-Loop (SiL) simulation, supported by automatic code generation. The tool chain will allow for both Test Automation (TA) and Design Space Exploration (DSE) of CPSs. The INTO-CPS technologies will be accompanied by a comprehensive set of method guidelines in order to describe how to adopt the INTO-CPS approach, lowering entry barriers for CPSs development. The tool chain will be tested with four case studies in railways, agriculture, building and automotive domains.

4 CONCLUSION

The paper presents the current State-of-the-Art related to CPSs and provides an overview of the various challenges involved in the design and development of CPSs, such as integration of cloud-computing; automatic testing, raising of design abstraction levels among others. This field study has been carried out to help researchers active in CPSs domain to better understand the various activities and inherent involved challenges, and aims to inform them about future research directions for CPSs. The paper is also to be used as a basis for the future research activities carried out in the INTO-CPS project where we would aim to extending the SysML standard to support CPS modeling and providing clear semantics on SysML usage.

ACKNOWLEDGMENTS

This research presented in this chapter is funded by the European Community's H2020 framework under grant agreement no 644047 (INTO-CPS). The authors wish also to thank the project partners and the research carried out in the following Seventh Framework Program (FP7/2007-2013) and ARTEMIS projects: EU FP7 MADES, ENOSYS, MODACLOUDS, FITTEST and PRESTO.

References

[1] P. Derler, E.A. Lee & A. Sangiovanni-Vincentelli (2012), *Modeling Cyber-Physical Systems*, Proceedings of the IEEE, V.100, N.1, January.

[2] AUTOSAR 2014. Available: http://www.autosar.org/

[3] Vector Informatik (2013), *PREEvision: Model-based Electrical/Electronic development from architectural design to production readiness*.

[4] J.C. Jensen, D.H. Chang & E.A. Lee (2011), *A Model-Based Design Methodology for Cyber-Physical Systems*, Proc. of the First IEEE Workshop on Design, Modeling and Evaluation of Cyber-Physical Systems, Istanbul, IEEE.

[5] H. Posadas, P. Peñil, A. Nicolás & E. Villar (2013), *System synthesis from UML/MARTE models: The PHARAON approach*, Proc. of the Electronic System Level Synthesis Conference, ESLsyn, IEEE.

[6] L. Carloni, R. Passerone, A. Pinto, and A. Sangiovanni-Vincentelli (2006*), Languages and tools for hybrid systems design*, Foundations and Trends in Electronic Design Automation, 1(1/2):1–193.

[7] Pereira, E., Sengupta, R., Hedrick, K. (2013), *The C3UV Testbed for Collaborative Control and Information Acquisition Using UAVs*, American Control Conference, Washington DC, USA.

[8] Object Management Group (2011), *Unified Modeling Language 2.4.1* Available: http://www.omg.org/spec/UML

[9] Object Management Group (2012), *System modeling language specification v1.3*. Available: http://www.omg.org/spec/SysML/1.3/

[10] Object Management Group (2011), *UML MARTE profile 1.1*. Available: http://www.omg.org/spec/MARTE/1.1/

[11] E. Lee and H. Zheng (2012), *Operational semantics of hybrid systems*, In M. Morari and L. Thiele, editors, Hybrid Systems: Computation and Control (HSCC), volume 3414 of LNCS.

[12] SINTEF (2014) , CloudML. http://cloudml.org

[13] Object Management Group (2012), *UML Profile for Test (UTP) 1.1.* Available: http://www.omg.org/spec/UTP/1.1/

[14] Object Management Group (2011), *Service oriented architecture Modeling Language (SoaML) Specification*. Available: http://www.omg.org/spec/SoaML/

[15] Paul Baker, Clive Jervis (2007), *Early UML Model Testing using TTCN-3 and the UML Testing Profile,* In: Testing: Academic and Industrial Conference Practice and Research Techniques - MUTATION, 2007. TAICPART-MUTATION 2007, IEEE computer society, pp. 45-54.

[16] Justyna Zander, Zhen Ru Dai, Ina Schieferdecker, George Din (2005), *From U2TP Models to Executable Tests with TTCN-3 – An Approach to Model Driven Testing*, In Testing of Communicating Systems Lecture Notes in Computer Science, 2005, Volume 3502/2005, 364, pp. 289-303.

[17] Stephan Pietsch, BogdanStanca-Kaposta (2008), *Model-based testing with UTP and TTCN-3 and its application to HL7*, In: Testing Technologies 2008.

[18] I. Michailidis, Martin F. Pichler, E. B. Kosmatopoulos (2013), *Multi-Linear State Space Model Identification for Large Scale Building Systems,* Proc. of the Sustainable Building Conference, Graz, Austria.

[19] S. Baldi, I. Michailidis, E. B. Kosmatopoulos and P. A. Ioannou (2011), *A 'Plug-n-Play' Computationally Efficient Approach for Control Design of Large-Scale Nonlinear Systems using co-Simulation*, IEEE Control Systems Magazine.

[20] CHESS ARTEMIS Project (2014). http://www.chess-project.org/

[21] CONTREX EU FP7 Project (2014). Available: https://contrex.offis.de/home/index.php/project/consortium

[22] CONCERTO ARTEMIS Project (2014). Available: http://www.concerto-project.org/

[23] COMPASS EU FP7 Project (2014). Available: http://www.compass-research.eu/approach.html

[24] J.Fitzgerald, P. G. Larsen, K. Pierce, M. Verhoef & S. Wol (2010), *Collaborative Modelling and Co-simulation in the Development of Dependable Embedded Systems*, in D. Mery & S. Merz (Eds.):"Integrated Formal Methods", Lecture Notes in Computer Science, V.6396, Springer.

[25] DREAMCLOUD EU FP7 Project (2014). Available: http://www.dreamcloud-project.org/

[26] MODACloudsEU FP7 Project (2014). Available: http://www.modaclouds.eu/

[27] S. Craciunas, A. Haas, C. Kirsch, H. Payer, H. Rock, A. Rottmann, A. Sokolova, R. Trummer, J. Love, and R. Sengupta (2010), *Information-Acquisition-as-a-Service for Cyber-Physical Cloud Computing*, In Proc. Workshop on Hot Topics in Cloud Computing.

[28] C. Kirsch, E. Pereira, R. Sengupta, H. Chen, R. Hansen, J. Huang, F. Landolt, M. Lippautz, A.

Rottmann, R. Swick, R. Trummer, and D. Vizzini (2012), *Cyber-Physical Cloud Computing: The Binding and Migration Problem*, In Design, Automation and Test in Europe.

[29] A. Ryan and J. Hedrick (2005), *A mode-switching path planner for UAV-assisted search and rescue*m In Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05.

[30] A. Ghose, P. Biswas, C. Bhaumik, M. Sharma, A. Pal, and A.Jha (2012), *Road condition monitoring and alert application: Using in-vehicle Smartphone as Internet-connected sensor*, In Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on, pages 489,491.

[31] Jiangchuan Huang, Clemens Krainer, Christoph M. Kirsch and Raja Sengupta (2012), *Service Provisioning in Cyber-Physical Cloud Computing*, Working Paper. University of California, Berkeley.

[32] Kirsch C et al. (2012), *Cyber-Physical Cloud Computing: The Binding and Migration Problem*, In DATE 2012.

[33] Craciunas, S., Haas, A., Kirsch, C., Payer, H., Röck, H., Rottmann, A., Sokolova, A., Trummer, R., Love, J., and Sengupta, R. (2010), *Information-Acquisition-as-a-Service for Cyber-Physical Cloud Computing*, Proc. Workshop on Hot Topics in Cloud Computing.

[34] Tao Yue et al. (2014), *Exploring Model-Based Repositories for a Broad Range of Industrial Applications and Challenges,* In MODELS 2014.

[35] MADES EU FP7 Project (2012). Available: http://mades-project.org/

[36] Object Management Group (2012), *ReusableAsset v2.2*. Available: http://www.omg.org/spec/RAS/2.2

[37] Softeam (2014), *Modelio Open-Source Modelling Environment*. Available: http://www.modelio.org/

[38] Ali, S., Hemmati, H. (2014), *Model-based Testing of Video Conferencing Systems: Challenges, Lessons Learnt, and Results*, In IEEE International Conference on Software Testing, Verification, and Validation (ICST).

[39] Yue, T., Briand, L., Labiche, Y. (2013), *Facilitating the Transition from Use Case Models to Analysis Models: Approach and Experiments*, Transactions on Software Engineering and Methodology (TOSEM).

[40] MultiPARTES EU FP7 project (2014). Available: http://www.multipartes.eu/

[41] DREAMS EU FP7 project (2014). Available: http://www.dreams-project.eu/

[42] PROXIMA EU FP7 project (2014). Available: http://www.proxima-project.eu/

[43] D. Guinard, V et al. (2010), *Interacting with the SoA-based internet of things: Discovery, query, selection, and on-demand provisioning of web services*, Services Computing, IEEE Transactions on, 3(3):223–235.

[44] S. Kowyra et al. (2013), *Verifiably-safe software-defined networks for cps*, In 2Nd ACM International Conference on High Confidence Networked System, HiCoNS '13, ACM, pp. 101–110.

[45] I. Schieferdecker, J. Großmann, M.-F. Wendland (2010), *Model-Based Testing: Trends*, Encyclopedia of Software Engineering 2010: 577-593.

[46] Felix Jakob, et. al (2012), *Risk-based testing of Bluetooth functionality in an automotive environment*, Automotive 2012 (GI LNI).

[47] A. Takanen, J. DeMott, C. Miller (2008), *Fuzzing for Software Security Testing and Quality Assurance*, Artech House.

[48] M. Schneider, et.al (2012), *Behavioral fuzzing operators for UML sequence diagrams*, in 7th Workshop on System Analysis and Modelling 2012 (SAMWkshp'12). LNCS, vol. 7744, pp. 87–103. Springer.

[49] P. McMinn (2004), *Search-based software test data generation: A survey*, Journal of Software Testing, Verification and Reliability, Wiley, 14(2):105–156.

[50] M.-F. Wendland, J. Großmann, A. Hoffmann (2010), *Establishing a Service-Oriented Tool Chain for the Development of Domain-Independent MBT Scenarios*, ECBS 2010: 329-334.

[51] Meszaros, G. (2007), *XUnit Test Patterns*, Addison-Wesley, Boston.

[52] JUnit (2013). Available: http://junit.org/

[53] CUnit (2014). Available: http://cunit.sourceforge.net/

[54] NUnit (2013). Available: http://www.nunit.org/

[55] Selenium Web Testing Framework (2014). Available: http://docs.seleniumhq.org/

[56] ETSI ES 201 873-1 V4.4.1 (2012-04), *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1*, TTCN-3 Core Language.

[57] TTworkbench - The Reliable Test Automation Platform (2014), Available: http://www.testingtech.com/products/ttworkbench.p.

[58] TestCast (2014), Available: http://www.elvior.com/testcast/introduction.

[59] Broadbit Test Tool (2014), Available: http://www.broadbit.net/portal/?page_id=392. 2014

[60] T. Blochwitz et al (2011), *The Functional Mockup Interface for Tool independent Exchange of Simulation Models*, in 8th International Modelica Conference, Dresden, pp. 20-22.

[61] A. Abel et al. (2012), *Functional Mock-up Interface in Mechatronic Gearshift Simulation for Commercial Vehicles*, proc. of the 9th International Modelica Conference, Munich.

Dependable Medical Cyber-Physical System for Home Telecare of High-Risk Pregnancy

Adam Pawlak*, Janusz Jezewski **, Krzysztof Horoba**

** Institute of Electronics, Silesian University of Technology, Gliwice, Poland; e-mail: adam.pawlak@polsl.pl*

*** Institute of Medical Technology and Equipment ITAM, Zabrze, Poland; e-mail: jezewski@itam.zabrze.pl*

Abstract

A dependable medical cyber-physical system (MCPS) for telecare of pregnant women at home has been presented in the paper. The system consists of the body area network of advanced sensors, the personal area network that is responsible for embedded processing of physical signals, smart alerts, and a transmission channel to the surveillance centre. Design challenges and requirements have been shortly discussed.

Keywords: pregnancy telemonitoring, medical cyber-physical system, dependability requirements.

1 Introduction

Telemedicine is a new discipline that concerns use of clinical health care at a distance. It is enabled by progress in telecommunication, information technologies, automatic control, electronics, medicine and social computing. By breaking distance barriers it improves access to various medical services. In particular, telemedicine enables early recognition of threatening symptoms, remote therapy, long-term monitoring and patient care in the comfort of patients' homes, what has very broad societal impact. Additionally, the progress in wireless sensor networks, healthcare networks, bioengineering, and social computing offers new possibilities for innovation in eHealth.

Telemonitoring has been used for longitudinal home telecare of high-risk pregnancies [7][14][20]. Supported medical procedures are usually very simple and limited to a single instantaneous measurement of a single parameter, like blood glucose level, or acquisition of the electrocardiogram using a simple recorder applied to a patient’s chest. Usually, there are no mechanisms to assist a patient who, in case of home telemonitoring also becomes the operator of a medical procedure, as well as its target.

Recent progress in telecare services provided to patients' homes is due to advances in Cyber-Physical Systems (CPS). CPS constitute a technological chance for new applications in telemedicine assuring more advanced care and treatment of a patient. They can radically change numerous healthcare procedures and medical workflows. A higher new level of integrated intelligence is possible due to CPS that is characterized by interaction and coordination of computing processes with physical ones, as advocated by E. Lee [11].

CPS are being applied in many new domains [17], however those in eHealth in general, and in particular in telemedicine are among most remarkable ones. In fact, a separate class of CPS, namely Medical Cyber Physical Systems (MCPSs) are recognized in the literature [11][12], as interconnected, intelligent systems of medical devices which support a holistic treatment of a patient. The inherent feature of MCPS is a conjunction of embedded software control of networked medical devices with complex safety- and often life-critical physical processes exhibited by patients' bodies.

Numerous problems need to be addressed in designing the dependable telemonitoring MCPS, like: interoperability of heterogeneous medical devices (MDs) involved in pregnancy monitoring, remote control and management of MDs, dependable acquisition and processing of bio signals, reliable transmission to a surveillance centre (SC), assurance of high dependability level due to monitoring of life critical parameters, and coordination of work in a multidisciplinary caregivers team.

The hospital-based systems that assure complete obstetrical care of pregnant women are a sort of standard. A number of producers deliver such systems with a varying spectrum of functionality. This includes the MONAKO system from ITAM [15] that is used in numerous hospitals in Poland. A diagnostic information on a fetal state in MONAKO is acquired during cardiotocographic monitoring, i.e. the analysis of changes in a fetal heart rhythm, as connected to a uterine contraction activity and fetal movements. These data are transmitted from bedside fetal monitors to the surveillance centre where they are displayed and online analyzed. Alerts are generated in case of critical situations [20].

The MCPS technology has matured in last years to a level that enables home-based monitoring of pregnancy with a continuous communication to the surveillance centre [7]. Healthcare networks and social computing can also play an important supporting role for a patient. It ought however to be coordinated with the responsible caregivers, possibly through the MCPS system in use.

The problem, motivation, and approach for monitoring pregnant women at home are explained in the following section. In Section 3 requirements concerning envisioned monitoring MCPS are shortly addressed. Finally, the architecture of the MCPS is presented and conclusions concerning the required approach are drawn.

2 High-risk pregnancy monitoring at home

Women with diabetes problems, intrauterine fetus growth restriction, pregnancy-induced hypertension, or with post-term pregnancy and other diseases during pregnancy with abnormal medical history, are particularly predisposed to home monitoring of the fetal development process [2].

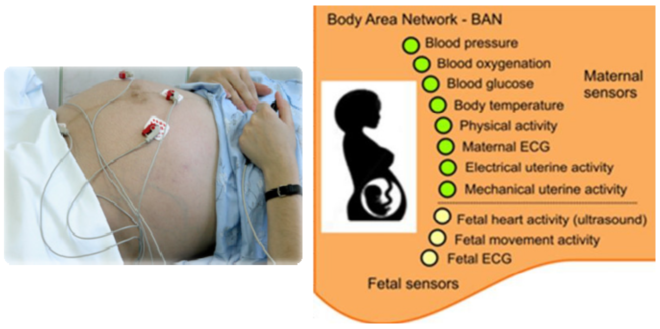


Fig 1. Real-time telemonitoring of pregnant women.

Traditionally, home-based monitoring sessions are assured by professional caregivers. A monitoring session includes: appropriate positioning of sensors on a maternal abdomen and verification of physical signals quality among others.

The other option is a monitoring session that is conducted by a patient alone or with a support of a family member. This patient-centered model is the most demanding one. It constitutes a motivation and sets goals for the research presented in the paper.

In home-based pregnancy monitoring a woman needs to be actively involved in telecare procedures, as she has to operate the monitoring medical equipment. The monitoring system ought to support her in self-control, assurance of an appropriate signal quality, learning and following medical procedures [27]. It ought to be a patient friendly tool, supporting search for the optimal position of the measuring sensor to ensure the best possible quality of the recorded signals [6]. The woman expects support from both the monitoring system and the remote surveillance centre, as well as time to time from a professional caregiver in person. Such support must enable a dynamic adaptation to changing conditions of measurement, since a fetus is somehow hidden in other object - mother [9]. Thus, interpretation made by a woman herself may lead to her unexpected and unjustified reactions. In such situations, the system should react with *smart alerts* [12] that are directed primarily to the surveillance centre, and only carefully provided to a woman.

3 Design challenges

Complexity of Medical CPS which is often life-critical is related to the monitored vital human physical processes [2][6]. MCPS must thus be upmost dependable [1][3]. Below, selected design challenges [24][26] are shortly discussed that have been found especially relevant to Medical Cyber-Physical Systems for telemonitoring of pregnant women as a result of the requirements analysis process.

Dependability of the system. Due to MCPS applications "around a patient" their *dependability* is of upmost importance. It has many dimensions, like: *reliability, security, safety, privacy,* and *trust* that must be resolved and assured through careful design verification, validation and final certification processes. *Dependability* represents the overall ability of the system to deliver service that can justifiably be trusted [1]. It is one of the key objectives of the designed smart MCPS for telemonitoring. The dependable MCPS can be easier achieved if appropriate holistic model-based design methodology is followed.

Interaction of caregivers and a patient with the MCPS system. A patient is time to time supported by either professional or non-professional (e.g. family members) caregivers. Human-Machine-Interface (HMI) is this tangible part of the telecare system that influences its usefulness to a patient and other caregivers. It should enable seamless and natural HM interactions. Its functionality should be *adaptable* to changing HM interactions depending on a caregiver's role, knowledge, experience and a care context.

Patient friendly eLearning. Telemonitoring requires patient consciousness participation to the monitoring process in order to assure validity of signals being transmitted to the surveillance centre. A patient or a family member needs to understand basics of the monitoring action that requires placing correctly sensors and performing a signal quality verification procedure. A patient friendly eLearning support in using the monitoring infrastructure ought to be provided. It can be based on the use of the easy to understand visual medical task patterns.

Support for caregivers. Different sorts of work provided by caregivers need to be coordinated [10] and conducted in a collaborative way at home in order to assure a *holistic healthcare* of a pregnant woman. A caregivers' team is also a *dynamic* one in the sense that its members may change. A direct consequence in changing membership in the caregivers team is the need to learn adopted medical procedures (*medical task patterns*) by a new team member. A monitoring system has thus to enable straightforward learning of new patterns of telecare. Realization of *medical workflows* by the monitoring system should be enabled, as well as increased *interoperability* of all involved medical devices.

Towards medical workflows automation. Medical work procedures represented as medical workflows should be supported by MCPS, and more specifically by their *smart HMInterface*. Medical workflows [10] predefine a flow of operations and medical data that have to be realized by caregivers, patients, and medical devices in order to realize a medical procedure.

Social computing and healthcare networks. A woman using a telecare support at home can also use help offered by *social computing* platforms [22]. Furthermore, she can benefit from healthcare networks [19][20] that offer an additional support that can be provided on-line for home-based care.

Interoperability of medical devices can be achieved by the deployment of the Internet-of-Things (IoT) concepts to medical devices, as demonstrated by the Hydra middleware and MDCF (Medical Device Coordination Framework) [8]. Standardization of interfaces of medical devices (MD) is required in order to enable open frameworks for interconnecting MD into eHealth systems [16]].

4 MCPS for telecare

Figure 2 illustrates a concept of a Medical Cyber-Physical System for telemonitoring of a pregnant woman at home. The physical part comprises a set of networked diverse medical devices (MDs) including sensors and actuators, whereas the cyber part is responsible for control and management of MDs, processing of acquired biosignals, smart alerting, caregivers' decision support system, and communication with the surveillance centre. The cyber part comprises an intelligent Human-Machine Interface (HMI) that enables a caregiver-specific functionality of the interface. The system provides telemonitoring of pregnant women through an easy to operate local infrastructure that is interconnected to the surveillance centre, but also through a Mobile Terminal that provides a contact with an attending physician.

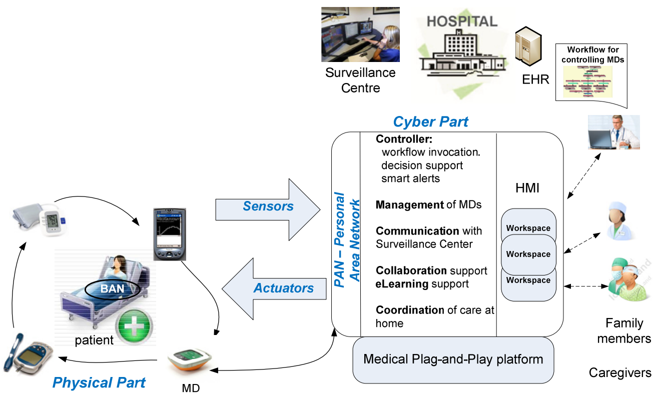


Fig. 2 The system context of the telemonitoring MCPS.

4.1 Body and Personal Area Networks

The straightforward use of the medical CPS at home is of prime importance, as a pregnant woman alone or with a help of her family, and occasionally with support of a nurse should be able to set up her BAN (Body Area Network) [4] and PAN (Personal Area Network) in order to establish a telemonitoring facility with the surveillance centre. BAN comprises advanced sensors that are interconnected on a body of a pregnant woman. Physical signals of different type (mechanical and electrical) are transmitted wirelessly to the cyber part of the platform comprising PAN. Acquisition of these signals with an acceptable quality level is one of the challenges of the whole approach [18] .

PAN is responsible for embedded processing of physical signals, smart alerts, and a transmission channel to the surveillance centre. Since a baby (fetus) is a hidden object, a much more advanced processing than in a case of mother has to be applied for its physical signals acquisition in order to obtain information relevant to a fetal health status.

Medical Device Plug-and-Play architecture is involved to ensure interoperability between the instrumentation measurements channels in a given monitoring scenarios, according to instrumentation descriptions and interacting requirements. Personal Terminal (PDA, Laptop) records and transmits the measurement data through the Internet to the surveillance centre. The terminal controls a matching of the algorithms for automated analysis of acquired signals basing on the patient's medical data (EHR) received from the hospital information system.

4.2 Physical signals acquisition

The ultrasound fetal monitor is used as an input device that verifies the fetal rhythm changes [23], and records the uterine contraction activity by means of the strain gauge transducer. Optionally, the module developed at ITAM [5] can be applied which enables a long-term non-invasive recording of fetal and maternal bioelectrical signals from the maternal abdominal wall. Bioelectrical monitoring of the uterine activity proves to be a more reliable method than the classical recording of mechanical activities. Additionally, it provides an opportunity for early detection of preterm threatening labour. This recorder will be equipped with special functionality that depending on the scope of the fetal ECG analysis (variability of the heart rate only, or the fetal QRS complex morphology as well), enables during the monitoring session to switch between the higher efficiency of heart beats detection and the higher precision of the beat-to-beat period determination [18][25].

4.3 Smart HM interface

The smart HM interface enables role-dependent workspaces that are generated from the elicited during field studies knowledge models of users' interactions with the system. Furthermore, the system should provide support for effective collaboration among members of the caregivers' team, e.g. by enabling coordination of their tasks.

A patient will receive through the HM interface information from both the system itself and the attending physician or other care provider. During a medical tele examination of a patient the physician is guided by certain rules (medical patterns) regarding how medical information should be prepared for a patient. Additionally, he will be supported in dynamic adaptation of information and messages provided to a patient depending on the patient's personality type and her possible reactions.

A physician based on his own experience and system support formulates thus appropriate messages in such a way as not to cause an adverse patient's reaction. A similar mechanism is proposed for system-patient communication when the system is a source of information or alert. Both information and alert messages and their presentation style depend on the assignment of a patient to a particular behavioural class.

5 Conclusions

Designing a dependable medical cyber-physical system for telecare of pregnant women at home requires a holistic approach that takes various patients', professional, as well as nonprofessional caregivers' activities into consideration. MCPS for telecare ought to present a smart HM interface that enables: medical task patterns and workflows, intelligent alerting at home and to the surveillance centre, as well as coordination with healthcare networks and social computing support. The presented MCPS for home-based pregnancy telemonitoring should provide telecare support to women from a group of high-risk pregnancy. In order to assure the required dependability of the system its design team should be a multidisciplinary one that closely collaborates with gynaecologists, obstetricians and psychologists. Design efforts are based on the experience gathered in the development of the pregnancy monitoring system MONAKO [15] that is currently used in over hundred Polish hospitals.

References

[1] A. Avizienis, J.C. Laprie, B. Randall, and C. Landwehr (2004), *Basic concepts and taxonomy of dependable and secure computing*, IEEE Transactions On Dependable And Secure Computing, vol 1, pp 11-33.

[2] R. Czabanski, M. Jezewski, J. Wrobel, J. Jezewski, K. Horoba (2010), *Predicting the Risk of Low Fetal Birth Weight from Cardiotocographic Signals using ANBLIR System with Deterministic Annealing and e-Insensitive Learning*, IEEE Transactions on Information Technology in Biomedicine, vol 14, pp 1062-1074.

[3] DCPS - *Dependable Cyber Physical Systems Network DAAD project*, Brandenburg Tech. Univ., Germany.

[4] M. Hanson, H. Powell Jr, A. Barth, K. Ringgenberg, B. Calhoun, J. Aylor, J. Lach (2009), *Body Area Sensor networks: Challenges and Opportunities*, IEEE Computer.

[5] J. Jezewski, J. Wrobel, K. Horoba, T. Kupka, A. Matonia (2006), *Centralised fetal monitoring system with hardware-based data flow control*, Proc. of IIIrd Int. Conference MEDSIP, Glasgow, vol VII, pp 51-54.

[6] J. Jezewski, A. Matonia, T. Kupka, R. Czabanski (2012), *Determination of the fetal heart rate from abdominal signals: evaluation of beat-to-beat accuracy in relation to the direct fetal electrocardiogram*, Biomedical Engineering, vol 57, no 5, pp 383-394.

[7] J. Jezewski, A. Pawlak, J. Wrobel, K. Horoba, P. Penkala (2015), *Towards a medical cyber-physical system for home telecare of high-risk pregnancy,* Proc. IFAC 13th Conf. on Programmable Devices and Embedded Systems, Cracow, Poland, IFAC-PapersOnLine 48 (4), pp. 466-473.

[8] A. King, D. Arney, I. Lee, O. Sokolsky, J. Hatcliff, S. Procter (2010), *Prototyping Closed Loop Physiologic Control with the Medical Device Coordination Framework,* SEHC (Software Engineering in Health Care), May 3-4, Cape Town, South Africa.

[9] F. Kovacs, C. Horvath, A.T. Balogh, G. Hosszu G. (2011), *Fetal phonocardiography - Past and future possibilities*, Computer Methods and Programs in Biomedicine, vol 104, pp 19-25.

[10] E. Lamine, A.R. Tawil, R. Bastide, H. Pingaud, (2014), *Ontology-based Workflow Design for the Coordination of Homecare Interventions*, Collaborative Systems for Smart Networked Environments, vol. 434, IFIP AICT, (eds) Luis Camarinha-Matos, Hamideh Afsarmanesh, Springer.

[11] E. Lee (2008), *Cyber Physical Systems: Design Challenges,* Proc. 11th IEEE Symp. on Object Oriented Real-Time Distributed Computing (ISORC).

[12] I. Lee, O. Sokolsky, et al. (2012), *Challenges and Research Directions in Medical Cyber-Physical Systems*, Proc. of the IEEE, vol 100, no1.

[13] T. Li, J. Cao, J. Liang, J. Zheng (2014), *Towards context aware medical cyber-physical systems: design methodology and a case study*, Cyber-Physical Systems, DOI: 10.1080/23335777.2014.972686.

[14] A. Di Lieto, M. Campanile, M. De Falco, I. Carbone, G. Magenes, M. Signorini and D. Di Lieto, (2011), *Prenatal Telemedicine: A New System for Conventional and Computerized Telecardiotocography and Tele-Ultrasonography*, Advances in Telemedicine, (eds) G. Graschew and T. A. Roelofs, InTech Publisher, pp 121-154.

[15] ITAM, *MONAKO - obstetrical care system* (2014), http:// itam-system.com/index.php?action=monako

[16] MDP&PIP (2014) - *Medical Device "Plug-and-Play" Interoperability Program* , http://mdpnp.org/

[17] NIST, *Strategic R&D Opportunities for 21st Century Cyber-Physical Systems* (2012), Foundations for Innovation in Cyber-Physical Systems Workshop, US.

[18] A. Pawlak, K. Horoba, J. Jezewski, J. Wrobel, A. Matonia (2015), *Telemonitoring of pregnant women at home - Biosignals acquisition and measurement,*

22nd Proc. Int. Conf. Mixed Design of Integrated Circuits and Systems, MIXDES 2015, Toruń, Poland.

[19] PCN (2014) - *Perinatal Care Network*, http://www.sandiegocounty.gov/content/sdc/hhsa/programs/phs/perinatal_care_network.html

[20] REACTION project (2014), *A Professional Service Platform for Remote Accessibility to Diabetes Management and Therapy in Operational Healthcare Networks*, http://www.reaction-project.eu

[21] D. Roj, K. Horoba, J. Wrobel, M. Kotas, J. Jezewski, T. Przybyła (2009), *Telemedical application for centralized home care of high-risk pregnancy based on control sharing approach*, IFMBE Proc. of the World Congress on Medical Physics and Biomedical Engineering, Munich, IX 2009, vol 25, pp 59-62.

[22] A. Sheth, P. Anantharam, C. Henson (2013), *Physical-Cyber-Social Computing- An Early 21st Century Approach*, IEEE Intelligent Systems.

[23] J. Jezewski, D. Roj, J. Wrobel, K. Horoba (2011), *A novel technique for fetal heart rate estimation from Doppler ultrasound signal*, BioMedical Engineering OnLine, 10, 10:92, doi:10.1186/1475-925X-10-92/.

[24] J. Wrobel, J. Jezewski, K. Horoba, A. Pawlak, R. Czabanski, M. Jezewski, P. Porwik (2015), *Medical cyber-physical system for home telecare of high-risk pregnancy – design challenges and requirements*, J. Med. Imag. Health. In., 5 (6) , pp. 1295-1301.

[25] J. Wrobel, A. Matonia, K. Horoba, J. Jezewski, R. Czabanski, A. Pawlak, P. Porwik (2015), *Pregnancy telemonitoring with smart control of algorithms for signal analysis*, J. Med. Imag. Health. In., 5 (6), pp. 1302-1310.

[26] N. Ndlovu, K. Sibanda (2014), *Innovative Framework Requirements for Remote Maternal and Fetal Health Monitoring in Low Resource Settings: Mobile Phones & Medical Devices*, International Journal of Computer Science Issues, 11 (2), pp. 208-216.

[27] A. Kazantsev, J. Ponomareva, P. Kazantsev, R. Digilov, P. Huang (2012), *Development of e-health network for in-home pregnancy surveillance based on artificial intelligence*, in:Proceedings of the IEEE International Conference on Biomedical and Health Informatics, Hong Kong, pp. 82-84.

Simulating Next-Generation Cyber-Physical Computing Platforms

Paolo Burgio
University of Modena and Reggio Emilia, Italy; email : paolo.burgio@unimore.it
Carlos Alvarez, Eduard Ayguadé, Antonio Filgueras, Daniel Jimenez-Gonzalez, Xavier Martorell and Nacho Navarro
Barcelona Supercomputing Center, Spain; email : {name.surname}@bsc.es
Roberto Giorgi
University of Siena, Italy; email : giorgi@dii.unisi.it

Abstract

In specific domains, such as cyber-physical systems, platforms are quickly evolving to include multiple (many-) cores and programmable logic in a single system-on-chip, while including interfaces to commodity sensors/actuators. Programmable Logic (e.g., FPGA) allows for greater flexibility and dependability. However, the task of extracting the performance/watt potential of heterogeneous many-cores is often demanded at the application level, and this has strong implication on the HW/SW co-design process. Enabling fast prototyping of a board being designed is paramount to enable low time-to-market for applications running on it, and ultimately, for the whole platform: programmers must be provided with accurate hardware models, to support the software development cycle at the very early stages of the design process. Virtual platforms fulfill this need, providing that they can be in turn efficiently developed and tested in a few months timespan. In this position paper we will share our experience in the sphere of the AXIOM project, identifying key properties that virtual platforms modeling next-generation cyber-physical systems should have to quickly enable simulation-based software development for a these platforms.

1 Introduction

As the technological scaling for semiconductors predicted by Moore's law hit the so-called *power wall*, and energy consumption became a primary concern for the market of electronic devices, computing platforms shifted to many-core heterogeneous designs [1, 2, 3, 4]. These platforms are perfectly suited to meet the requirements especially of next-generation cyber-physical systems (CPS), where a huge number of peripherals interacting with the surrounding environment are coupled to a computing board delivering high performance/watt through many-core SMPs and hardware accelerators. Sensors and actuators will be integrated in the design through *ad-hoc* bridges/circuits, or more flexible reprogrammable logic (e.g., FPGAs), composing a system made of several communicating nodes with one or more centralized controllers running on general purpose SMP cores. Hardware accelerators are application-specific circuits which increase the power efficiency of portions (*kernels*) of applications by orders of magnitude. The consequence is that, today, software developers must write code that runs on multiple cores and uses the hardware resources available in the platform, in a productive and effective manner: extracting the tremendous performance/watt potential of such a complex platform essentially becomes also a software development problem. Dependability is also improved when adopting programmable logic: for example, systems based on programmable logic can execute a function in a deterministic way, without the need of a continuous push-pull to/from caches. Most systems based on caches tend to offer a good average performance but may fail to respect a hard deadline in the worst case. Moreover, if the specific architecture fails, reconfiguration of the FPGA can help. Concepts like Data-Flow Threads (DF-Threads) [5, 6] can enable the repetition of the execution of a failed thread.

Virtual platforms are the key to fight this problem, as they enable fast software prototyping at the very early stages of the design cycle of a board, where hardware is not yet 100% available. Computer architects are well aware of this, and in recent years a number of simulator infrastructures have been developed [7, 8, 9], and eventually commercialized, that model a generic or specialized computing fabric with also high accuracy (e.g., cycle-accurate [9, 10]). Unfortunately, correctly modeling the behavior of an hardware platform is time-costly: fully cycle-accurate simulators[1] can be orders of magnitude slower than the corresponding hardware counterparts [11]. For this reason, recently, some virtual platforms were proposed (such as Qemu [11]), for pure functional simulation. They can be successfully adopted in an initial phase to enable functional testing/debugging of the alpha versions of applications, and to quickly exploring the hardware/software partitioning of applications into kernels. Then, software developers might resort to slower and fully cycle-accurate simulators in advanced stages of debugging, until the first prototypes of the board are available.

In this position paper we describe our preliminary analysis on building a virtual platform for simulating cyber-physical systems, in the context of the AXIOM project [12]. AXIOM explores energy-efficient, many-core platforms for next-generation cyber-physical systems. We will briefly describe the guidelines that drive the development of the AXIOM board, in section 2. In section 3 we decompose a simulator for many-core heterogeneous platforms in its basic building blocks, and for each of them we discuss in detail the main issues in simulating it, and how it can (should) be accurately modeled in the quickest way possible. We will do this bringing our expertise on already existing simulation platforms,

[1] cycle accurate virtual platforms mimic the behavior of each component of a system at every clock cycle

both industry [7, 8] and academical solutions [10]. Finally, section 4 draws some conclusions.

2 Requirements for a cyber-physical system: The Axiom project

We are entering the *cyber-physical age*, where both objects and people will become nodes of the same digital network for exchanging information. This vision is also referred to as *"Internet of Things"* (IoT) becauses the general expectation is that "things" or systems will become somewhat smart as people, allowing a tight system-to-human and device-to-environment interaction. As a consequence, we expect that such cyber-physical systems (CPS) will at least react in real-times, consume the least possible energy for a given task, scale up through modularity, and allow for an easy programmability across performance scaling. The whole set of these expectations impose scientific and technological challenges that AXIOM project (Agile, eXtensible, fast I/O Module[12, 13]) tries to address, exploring new hardware/-software architectures for CPSs.

Communities [14, 15, 16] that are using CPSs are devising more and more the need for more powerful embedded platforms that could be: i) easy programmable through an almost standard software toolchain; ii) be customizable with programmable logic (i.e., FPGAs), iii) be extensible to one or more boards (e.g., two robotic arms that need to be closely synchronized toward a single real-time task); iv) provide an easy way to integrate sensors (e.g., through widely available Arduino [15] shields). Current solutions providing enough energy-efficient computational power for fulfilling this needs are starting to rely more and more on multi- and many-core architectures (e.g., UDOO [14] and RaspberryPi2 [16] rely on a quad-core and GPUs) . For example, some current research projects (such as ADEPT [17] or FP7 P-SOCRATES [18]) are already investigating how to join efforts from the high-performance computing (HPC) and the embedded computing domains, which are both focused on high power efficiency, while GPUs and new dataflow platforms such as Maxeler's [19], or in general FPGAs, are claimed as the most energy efficient.

AXIOM research mainly targets designs coupling power-efficient multiple cores, such as ARM ones, and FPGA accelerators on the same die as in the Xilinx Zynq [1], and produce prototypes of single-board computers, similar to UDOO [14], Arduino [15] and RaspberryPi [16]. This architecture includes capability to high-speed board-to-board interconnects and controllers for commodity CPS peripherals such as *Arduino Shields*. AXIOM partners will start the development using a virtual platform: this paper reports the preliminary results of such investigations. At the same time, the tested parts, when ready, are progressively migrated on the FPGA prototype (a Xilinx ZC-706). As a consequence, the AXIOM project requires a virtual platform which simulates general-purpose cores, programmable logic (for accelerators), and peripherals ASIC circuits that integrate sensors and actuators. Figure 1 shows the scheme of a computing platform including two general-purpose cores, FPGA logics and a few peripherals/sensors connected to it.

From the software viewpoint, the AXIOM system will interact with and react to the surrounding environment by properly managing actions in real-time through an operating system (such as Linux), a well-known parallel programming model:

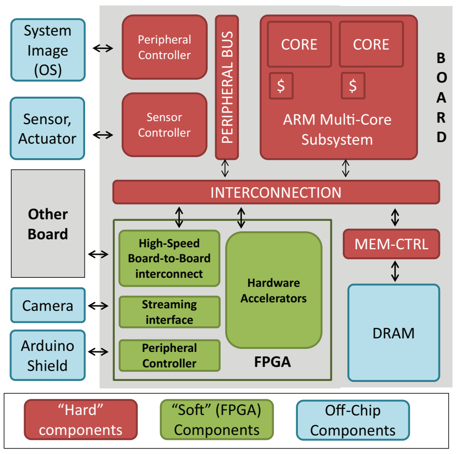


Figure 1: Heterogeneous computing platform with sensors

OmpSs [20]. By using OmpSs, applications will be hardware/software partitioned, i.e., decomposed in parallel tasks that can be mapped on multiple software execution units (OS threads) and/or hardware execution units, e.g. the accelerators in the FPGA. This provides a huge number of options for mapping tasks to resources, considering the device on which a task is mapped, the size of the input data, the data transfer time, or the different speed of the devices in executing the task. Tools and techniques for quickly finding the optimal HW/SW partitioning of applications according to performance and power metrics, and to validate them against real-time constraints, are therefore crucial for the project. The issue is that, when all tasks are mapped either on hardware or software, a complete FPGA synthesis flow for hardware accelerators can spend from hours to days, depending on the size of the computational kernels to process. With virtual platforms, on the contrary, new accelerator elements can be quickly added in the simulation environment, and we can run a timing accurate full system simulation of the applications partitioned on the SMP cores and FPGA accelerators in a matter of minutes to few hours.

3 Virtual Platform requirements

This section describes how to build a virtual platform for a computing system such as the one targeted by AXIOM. Starting from AXIOM specifications, we will first describe its basic building blocks, and discuss how a proper design for each of them will enable fast prototyping of the target board. We will bring our expertise, previously gained using/developing two simulator for heterogeneous systems, namely COTSon [7] in the TERAFLUX project [21, 22], and a prototype built after the open-source academical VirtualSoC [9] by University of Bologna: HC-VSoC [10, 23]. We will also refer to other existing simulator infrastructure of potential interest.

From AXIOM specifications, the simulator must enable quick software prototyping of a system *whose hardware architecture is not 100% defined at early stages of the project*. We identify these four key requirements:

1)immediate availability of at least a first functional version of the simulator, to let the software development cycle start;
2) possibility of defining architectures and their timing model for cycle accurate evaluations, to be selectively used in combination with functional models;

3) the virtual platform must be capable of integrating multiple modules (such as proximity sensors), that *generate/simulate events coming from the surrounding environment*, hence whose behavior must be random, or driven by user/parametrizable;
4) we must be capable of easily putting new hardware models in the design, and to replace (fast and inaccurate) behavioral models of its components, e.g., sensors and actuators, with more timing-accurate (and slower) versions.

Requirements 1) and 2) match the experience of the COTSon [7] simulator, while requirements 3) and 4) match the experience of the HC-VSoC [10, 23] simulator, which models platforms with user-defined hardware accelerators called HWPUs, i.e., whose functionality is defined by the end-user. HC-VSoC architects reduce the problem complexity by specifying a clearly-defined common communication contract and infrastructure for all the blocks modeled in the systems. SMP cores and HWPUs are equipped with a memory shell that supports that communication protocol. Also the COTSon [7] virtual platform, whose primary design requirement was to build a highly scalable architecture, employs a similar solution, providing a well-defined communication API.

This brings us to the first component of a virtual platform, the **simulation engine**, which supports/simulates the interaction between modeled blocks. A number of tools for this exist, both coming from industry and academia, and the most known is probably SystemC [24] by Accelera. The SystemC package is a very flexible macro library (C++ language) and a simulation engine, to simulate the behavior of hardware blocks with different levels of abstraction and accuracy, from RTL- to cycle-accurate, to transactional-level modeling. Higly-scalable infrastructures (such as – as explained before – COTSon [7], or OVP [8]) expose a very simple API to integrate hardware blocks in their engines, and come with a few pre-built architecture models. These are the best solution if the architecture models included in these packages partially or totally match the one being developed.

An second component that must be carefully designed at early stages is the **interconnection**, which emulates on- or off-chip connectivity. Designing an interconnection infrastructure with acceptable good tradeoff of simulation speed, scalability, and timing accuracy is not trivial and it is probably the most time-consuming part in developing virtual platforms. In addition, in AXIOM, the interconnection must enable fast integration of the future versions of the hardware models, to meet requirement 4). An example of scalable communication *medium* is the one connecting multiple COTSon [7] nodes, or the one of HC-VSoC [23], whose protocol is called PINOUT. The difference between them is in the way they are implemented: in the former it is exposed as a pluggable model rigorously decoupled as a functional model and a timing model. Hence the simulated hardware blocks access to the interconnection by directly invoking a simple given API. The latter is itself an instantiated as a SystemC modules with its own model of hardware ports, and a timing accuracy given by design. An amenable property of a simulated interconnection (although not a critical requirement for AXIOM), is that it should be possible to customize its internals and the modeled communication delay should be driven, e.g., *via* simulator parameters or configuration files. An example of configuration files for a simulator is shown in Figure 2. It was developed in the PREDATOR FP7 project [25].

Rows and columns in the figure simulate a hierarchical crossbar by specifying the communication delay among each master port (e.g., SMP cores) and each slave ports (memories and peripherals), respectively. In this example, we are modeling an N-to-M crossbar with user-defined delay for each master-to-slave (core-to-mem bank) path: for instance, we note that the M-th slave implements the controller for an off-chip DDR, because the delay that each master "sees" to get to it (specified in the last column) are significantly higher than for other memories.

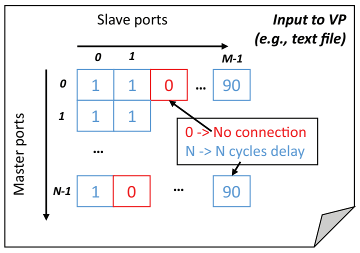


Figure 2: Parametrizable interconnection model

General Purpose cores are the most complex component of a simulator, as they must correctly mimic the functional behavior, e.g., of modern superscalar cores, with branch-prediction units and multiple deep pipelines, or the complex hierarchical shared cache systems and prefetch buffers of next-generation many-core architectures. Luckily, the choice of the instruction-set architecture (ISA) and core model to adopt is usually made at the very beginning of the project, and it does not change in the following. Moreover, most of the simulation infrastructures provide a portfolio of processor models, which is often freely available as a library (see for instance OVP [8]).

The key point in integrating core models in a bigger design is that, in order to support the development of software, *each core model must come with the required toolchain for compiling the code of applications, deploying them on the simulator and – possibly – to support debugging* to do what ultimately is the main purpose of a virtual platform: support software development. In AXIOM, this is reflected in requirement 1).

A few examples can be:

- COTSon [7], which includes x86_64 processor models together with the associated toolchain;
- The HC-VSoC package [10, 23] targets for ARM-based embedded systems, and it comes with a "standard" GNU Compiler Collection (GCC [26]) cross-compiled for it;
- Open Virtual Platforms by Imperas [8] provides a wide portfolio of core models, including ARM (32 and 64 bit), Imagination MIPS (32 and 64 bit), PowerPC, Xilinx Microblaze, and many more.

A project can also adopt a proprietary ISA from a specific provider: they also usually come with a library/software package that simulates a single processing core, using "open" simulation engines (e.g., SystemC), or again with in-house engines or define ISEs (ISA Extensions).

Due to its complexity, the processor model is usually the component from which the development of a virtual platform starts, together with the simulating engine. For this reason, the preliminary version of the platform provided to programmers typically embeds only one or multiple SMP cores, the interconnection model, and a few memories, with limited set of peripherals. Using this, software developers for an heterogeneous platform (such as AXIOM's) can immediately compile, deploy and test the "host/SMP part" of their code.

Programmable logic and peripherals (and sensors/actuators). The platform template targeted by AXIOM embeds on-chip programmable logic, as well as a number of peripherals controllers to interact, e.g., with sensors or Arduino shields. Once the communication infrastructure has been set, and a scalable model of the on-chip interconnection implemented as explained before, it is extremely easy to include in the simulator in-house customized models for peripherals and hardware accelerators. For instance, the PINOUT interface in [23] is implemented in the so-called COMU of HC-VSoC HWPUs. Internally, each of HWPU model can be implemented with a different simulation speed/timing accuracy tradeoff, as required by project specification.

Integration with external components. In the AXIOM project, peripheral components will either interact with the surrounding environment, or connect the board to COTS components or 3rd party subsystems such as the Arduino Shields, and the virtual platform will simulate these behaviors. In the first case, we can employ parameters or proper input files for the simulator that mimic the surrounding environment. For instance, the behavior of temperature sensors can be easily defined *via* simple input text files describing the variation of the temperature in time. The second scenario, in turn, has a great impact on the simulation infrastructure, and raises a potential problem. Simulator developers might need to integrate pre-existing models of the two platforms (e.g., the core model running on SystemC, and the model of an Arduino running on a proprietary simulation engine), which are potentially not designed to communicate each other, or can even be written in different programming language. This possible incompatibility in the communication between simulator models, may require to implement stub functions to transform the information between formats understood by the two components.

Memories. In current virtual platforms, typically memories are implemented as "wrappers" that simply add a delay for accessing big arrays of data modeling the memory banks. For this reason, it is not uncommon that virtual platform developers create their in-house simulation models of memories, when possible. More complex or "fancy" memory models, such as smart memories, can be easily implemented starting from these components.

Support software libraries Applications running on the simulator might employ specific standard libraries, such as `libc` and `libsdtc++`, or custom runtimes, such as `nanos++` [27], or `libhwpu` (in HC-VSoC) to do their work. This is also the case of AXIOM. In this case, the simulator infrastructure must support the same set of required APIs as the "real" board, to ensure code portability.

4 Conclusions

We presented in this paper the approach used by the AXIOM project for flexibly simulating a realistic Cyber-Physical System, soon to be implemented as single board computer. Mainly, besides a FPGA prototype, we developed the preliminary steps through virtual platforms. In particular two platforms had been selected: COTSon and HC-VSoC as they can provide the best support for our design needs. In particular, the inclusion of FPGA in the simulation toolchain provides support for exploring dependability options.

5 Acknowledgment

This work is supported by the AXIOM project, funded by EU H2020 program (grant ICT-01-2014 GA 645496), the Spanish Government, through the Severo Ochoa program (grant SEV-2011-00067) the Spanish Ministry of Science and Technology (TIN2012-34557) and the Generalitat de Catalunya (MPEXPAR, 2014-SGR-1051). Authors would also like to thank the anonymous reviewers for their precious feedback.

References

[1] Xilinx Inc., *Zynq Series*.

[2] G. Kyriazis (2012), *Heterogeneous System Architecture: A Technical Review*.

[3] AMD, *The AMD Fusion Family of APUs*.

[4] R. Giorgi (2015), *Scalable embedded systems: Towards the convergence of high-performance and embedded computing*, in Proceedings of the 13th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing.

[5] R. Giorgi and P. Faraboschi (2014), *An Introduction to DF-Threads and their Execution Model*, in IEEE Proceedings of MPP-2014, (Paris, France), pp. 60–65.

[6] S. Weis, A. Garbade, B. Fechner, A. Mendelson, R. Giorgi, and T. Ungerer (2014), *Architectural support for fault tolerance in a teradevice dataflow system*, Springer International Journal of Parallel Programming, pp. 1–25.

[7] E. Argollo, A. Falcón, P. Faraboschi, M. Monchiero, and D. Ortega (2009), *COTSon: Infrastructure for Full System Simulation*, SIGOPS Oper. Syst. Rev., vol. 43, pp. 52–61.

[8] Imperas Software, *OVP – Open Virtual Platforms*.

[9] D. Bortolotti, C. Pinto, A. Marongiu, M. Ruggiero, and L. Benini (2013), *VirtualSoC: A Full-System Simulation Environment for Massively Parallel Heterogeneous System-on-Chip*, in Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International, pp. 2182–2187.

[10] P. Burgio, A. Marongiu, D. Heller, C. Chavet, P. Coussy, and L. Benini (2012), *OpenMP-based Synergistic Parallelization and HW Acceleration for On-Chip Shared-Memory Clusters*, in 15th Euromicro Conference on Digital System Design, DSD 2012, Cesme, Izmir, Turkey, pp. 751–758.

[11] F. Bellard (2005), *QEMU, a Fast and Portable Dynamic Translator*, in Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC '05, (Berkeley, CA, USA), pp. 41–41, USENIX Association.

[12] D. Theodoropoulos *et al.* (2015), *The AXIOM project (agile, extensible, fast i/o module)*, in IEEE Proceedings of the 15th International Conference on Embedded Computer Systems: Architecture, MOdeling and Simulation.

[13] C. Alvarez, E. Ayguade, J. Bueno, A. Filgueras, D. Jimenez-Gonzalez, X. Martorell, N. Navarro, D. Theodoropoulos, D. N. Pnevmatikatos, C. Scordino, P. Gai, C. Segura, C. Fernandez, D. Oro, J. R. Saeta, P. Passera, A. Pomella, A. Rizzo, and R. Giorgi (2015), *The AXIOM software layers*, IEEE Proceedings of the 18th EUROMICRO-DSD, pp. 117–124.

[14] UDOO, *Android Linux Arduino in a tiny single-board computer*.

[15] M. Banzi (2008), *Getting Started with Arduino*. Sebastopol, CA: Make Books - Imprint of: O'Reilly Media.

[16] The Raspberry Pi Foundation., *The Raspberry Pi Board*.

[17] The ADEPT Consortium, *ADEPT – Addressing Energy in Parallel Tehcnologies*. [Online]. Available: http://www.adept-project.eu/.

[18] L. M. Pinho, V. Nélis, P. M. Yomsi, E. Quiñones, M. Bertogna, P. Burgio, A. Marongiu, C. Scordino, P. Gai, M. Ramponi, and M. Mardiak (2015), *P-SOCRATES: a Parallel Software Framework for Time-Critical Many-Core Systems*, Microprocess. Microsyst., vol. 39, no. 8, pp. 1190–1203. [Online]. Available: http://dx.doi.org/10.1016/j.micpro.2015.06.004.

[19] Maxeler Technologies, *MPT Hardware*.

[20] J. Bueno, L. Martinell, A. Duran, M. Farreras, X. Martorell, R. M. Badia, E. Ayguade, and J. Labarta (2011), *Productive Cluster Programming with OmpSs*, in Proceedings of the 17th International Conference on Parallel Processing - Volume Part I, Euro-Par'11, (Berlin, Heidelberg), pp. 555–566, Springer-Verlag.

[21] R. Giorgi *et al.* (2014), *TERAFLUX: Harnessing dataflow in next generation teradevices*, Microprocessors and Microsystems, vol. 38, no. 8, Part B, pp. 976 – 990.

[22] R. Giorgi and A. Scionti (2015), *A scalable thread scheduling co-processor based on data-flow principles*, ELSEVIER Future Generation Computer Systems, pp. 1–10.

[23] P. Burgio, A. Marongiu, P. Coussy, and L. Benini (2014), *A HLS-Based Toolflow to Design Next-Generation Heterogeneous Many-Core Platforms with Shared Memory*, in 12th IEEE International Conference on Embedded and Ubiquitous Computing, EUC 2014, Milano, Italy, pp. 130–137.

[24] Accelerat Systems Initatives, *SystemC*.

[25] P. Burgio, M. Ruggiero, and L. Benini (2010), *Simulating Future Automotive Systems*, tech. rep., DEIS - University of Bologna.

[26] The Free Software Foundation, *GCC – The Gnu Compiler Collection*.

[27] A. Duran, E. Ayguadé, R. M. Badia, J. Labarta, L. Martinell, X. Martorell, and J. Planas (2011), *OmpSs: A proposal for programming heterogeneous multi-core architectures*, Parallel Processing Letters, vol. 21, pp. 173–193.

The CONCERTO Project: an Open Source Methodology for Designing, Deploying, and Operating Reliable and Safe CPS Systems

Silvia Mazzini

Intecs, via Umberto Forti, 5 Montacchiello – I-56121 Pisa, Italy; Tel: +39 050 9657 513; email: silvia.mazzini@intecs.it

(on behalf of the CONCERTO project partners)

Abstract

The principal goal of the CONCERTO project is to provide an open source methodology for designing, deploying, and operating reliable and safe cyber physical systems (CPS). Building on the results from the CHESS project, CONCERTO pursues this objective by identifying cross-cutting concerns, which are relevant for multiple application domains, notably avionics, space, telecommunications, telecare, petroleum, and automotive, and proposes solutions to address them in a uniform way.

Keywords: model-based, real-time, safety, component-based, modeling, correctness-by-construction, separation of concerns, CPS, multicore.

1 Introduction

As technology advances, we become increasingly dependent on automated systems, which serve and assist several aspects of our daily activities. The reliability and safety of such systems must be carefully evaluated before putting them to use. Indeed, many application domains, from telecommunications to medical, from petroleum to automotive, have to account for the provision of these properties all along the development process.

The approach of ensuring reliability and safety through testing and corrections is resource-consuming and not exhaustive, and may easily become unpractical with the increase of systems complexity and heterogeneity.

Presently, multiple heterogeneous techniques are used to this end, as a reflection of the diversity of the application domains and of the involved actors. In spite of the considerable maturity reached by the state of the art, not all the aspects of interest are fully covered yet: the complexity of modern systems keeps increasing, with new challenges being presented, responded by new standards (e.g., ISO 26262 [5], functional safety of road vehicles).

2 Objectives

The principal goal of the CONCERTO project (http://www.concerto-project.org) is to provide an open source methodology for designing, deploying, and operating reliable and safe systems. We pursue this objective by identifying cross-cutting concerns, which are relevant for multiple application domains, notably avionics, space, telecommunications, telecare, petroleum, and automotive, and propose solutions to address them in a uniform way.

Building on the results from the CHESS project (ARTEMIS-2008-1-100022), currently hosted in the Eclipse PolarSys open source ecosystem (https://www.polarsys.org/chess/), the CONCERTO[3] project (ARTEMIS JU Call 2012) consolidates and extends the CHESS component-based language, methodology and related tool support for the modeling and development of energy efficient and high-integrity multi-core systems.

Particular emphasis is devoted to the experimentation in real-life use cases from different domains to validate the methodology and measure its suitability to address concrete industrial issues.

The identification of a cross-domain common baseline is pursued to the furthest possible extent, beyond which domain-specific solutions are followed in order to promote a systematic domain-independent approach without however sacrificing the usability and effectiveness of the proposed methodology and tool support.

Spanning across the application areas of space, avionics, petroleum, medical, telecommunications and automotive, the CONCERTO use cases cover a wide spectrum of safety and reliability requirements.

3 The approach

The CHESS Project main aim is to improve Model Driven Engineering practices and technologies to better address safety, reliability, performance, robustness and other non-functional concerns, while guaranteeing correctness of component development and composition for embedded systems. System development costs could be reduced through extensive use of provable automation and model transformation engines specifically for high-integrity applications and through the identification of feasible

[3] "Guaranteed Component Assembly with Round Trip Analysis for Energy Efficient High-integrity Multi-core Systems"

solutions to complex system challenges earlier in the development process. Among the principal cornerstones around which the CHESS methodology was developed is the concept of *correctness-by-construction* and the principle of *separation of concerns*.

The CONCERTO project extends the CHESS results focusing on the development of multi-core systems: an area where complexity makes it difficult for a human being to grasp the pros and cons of different solutions and the implications of components integration. The urge for the definition of a rigorous modeling methodology that enables a property-preserving development process is evident.

The CHESS methodology and tool support is also extended in CONCERTO to provide advanced hardware modeling capabilities, and an enhanced multi-domain component model.

In the following we report on the principal core cross-domain extensions of the CHESS Methodology developed in CONCERTO, and the most relevant design flow support provided for specific industrial domains.

3.1 Multi-core deployment, scheduling and real-time analysis

For a software system that targets a multicore processor using a partitioned scheduling policy, each individual task should be statically assigned for execution to one given core. The partitioning choices have a direct influence on system performance, and may cause inefficient utilization of the system resources if inadequate assignment decisions are taken.

In a model-based development flow, free manual allocation of tasks to cores should be discouraged and instead be positively assisted so that poor decisions are avoided proactively. For this reason, CONCERTO supports an explicit design step that provides the user with guidance on recommended task-to-core allocations, which achieve adequate system utilization. The end user may either accept the proposed allocation, or modify it according to needs or preferences.

The CONCERTO solution to this challenge is addressed in the modeling environment in two ways: (1) an allocation dialog, and (2) an automated procedure. The allocation dialog allows the user to visually assign components to cores or to modify the proposed allocation. This visual tool is convenient for the user and for the CONCERTO implementer. The automated procedure computes a task-to-core allocation that has sufficiently good quality (for feasibility and resource utilization). This allocation step uses a sequence of different algorithms to find a feasible allocation. If none of them are successful, then the user is advised to relax the system load or to increase the number of CPUs and/or cores.

Addressing multi-core systems scheduling and timing analysis introduces the need to provide several new modeling capabilities in CONCERTO, such as the possibility to model multicore processors, and the ability to model and analyze hierarchically defined systems such as those respecting the ARINC [3] specification.

Moreover, in terms of analysis, new scheduling algorithms and shared resource protocols for multicore architectures are needed. The execution of scheduling and timing analysis for multi-core systems relies on an extended version of MAST, a scheduling and timing analysis tool developed and maintained by the Universidad de Cantabria [1].

3.2 Run-time monitoring

CONCERTO enables the modelling and analysis of complex systems spanning over various industrial domains. Yet, the modelling and analyses are based on a set of assumptions that must be verified and/or enforced at run-time after deployment of the application on its target platform. This is even more important on multi-core platforms where tasks are competing for hardware shared resources thereby generating large variations on the timing properties of the application. For these reasons, CONCERTO provides a set of solutions for the analysis of run-time traces as well as mechanisms for the run-time verification of the deployed application.

Such solutions include: (i) simple static analysis of the run-time traces that were actually monitored at run-time; (ii) statistical analysis on the monitored values, providing a way to survey the (possible) presence of statistical fluctuations in behavioral data captured at run-time; (iii) worst-case execution time bounds computation for the tasks running on a multicore platform: this solution is based on some of the results of the PROXIMA[4] project ; (iv) run-time verification of functional or extra-functional properties of the deployed application to verify at run-time whether the system specifications are respected or not, allowing to trigger alarms or counter-measures in case of system misbehavior. This run-time verification framework is specifically designed for safety critical and mixed-criticality applications by enforcing timing (and potentially space) isolation between the “monitors” and the monitored application.

3.3 Modeling dependability

3.3.1 The CONCERTO dependability profile

The Dependability profile of CHESS supports different techniques for dependability analysis, in particular techniques based on failure logic analysis, and state-based analysis techniques.

The Dependability Profile is extended in CONCERTO to address socio-technical systems by incorporating modelling capabilities related to Human factors [6]. More specifically, via the extended profile, not only technology (hardware and software) is taken into consideration. Organizations and humans are also considered and are interpreted as composite components, constituted of sensor- and actuator-

[4] http://www.proxima-project.eu/

like subcomponents, and their nominal as well as failure behavior can be specified and then used to calculate the failure behavior emerging at system level.

The Dependability Profile in CONCERTO is augmented to model failure modes and reason about criticality apportionment and typical measures of the confidence in the function performed by the system by incorporating modelling capabilities that capture criticality attributes.

Due to the different existing classification of criticality levels (e.g., SILs, ASILs, DALs [2]), the modeling language allows hierarchies of criticality levels to be specified and put in relation with each other. This approach allows: (i) safety models from different domains to be (at least partially) reused, and (ii) to have both domain-specific and cross-domain criticality levels defined in the same model.

3.3.2 Cross fertilization

Safety analysis of complex systems is a challenging task, which requires the combination of different techniques during the analysis process. Through the modeling language designed in CONCERTO, two very different kinds of analysis are supported: FLA (failure logic analysis), which is traditionally deterministic, and SBA (state-based analysis), which is inherently probabilistic.

Being able to apply both techniques together and to switch from one to another is a contribution to smoother model-based safety analysis. This explains the main motivation behind creating a harmonized modeling language for dependability: the existing analysis techniques can work in the same semantic space, and input and output information can be exchanged between them, promoting cross-fertilization. Cross-fertilization can occur in two directions, which basically reflect the top-down and bottom-up system design approaches.

3.3.3 Failure Logic Analysis

The Failure Logic Analysis techniques supported in CHESS are extended in CONCERTO with the ConcertoFLA [6]. With the ConcertoFLA, failure propagation phenomena can be analyzed automatically for the system of interest. ConcertoFLA builds on top of Failure Propagation and Transformation Calculus, which enables deductive as well as inductive hazards analysis towards semi-automatic generation of artefacts, necessary for arguing about HARA (Hazards Analysis and Risk Assessment).

ConcertoFLA provides valuable support to safety engineers in achieving compliance with safety standards in the specification and verification of certification-aware systems. Deductive and inductive HARA is relevant to virtually all certification-aware domains.

3.3.4 State Based Analysis

CONCERTO supports the automated evaluation of quantitative dependability metrics using stochastic approaches (State-Based Analysis). Properties added using the CONCERTO Dependability Profile are analyzed to obtain the probability of occurrence of specific failure modes and other quantitative metrics involving reliability, availability, and safety.

State-Based Analysis in CONCERTO supports safety engineers in the management of Reliability, Availability, Maintainability and Safety (RAMS) properties, and in the assessment of hazard rate threshold associated with safety integrity levels (e.g. SILs, ASILs, DALs).

A new feature introduced in CONCERTO state-based analysis is the ability to set initial (health) conditions for designated system components. With this feature CONCERTO supports safety engineers in the evaluation of specific configurations (e.g., overdue maintenance) which are deemed crucial for the fulfilment of safety requirements.

3.3.5 Software FMEA

CONCERTO provides the Software Failure Modes and Effect Analysis (SW FMEA) functionality to support the execution/simulation of user models, and the collection of analyzable execution traces [7]. Fault injection is supported at the model level. A set of predetermined faults can be injected on the interfaces of designated components, and the effect evaluated by executing/simulating the model and collecting the related execution traces. The effectiveness of safety mechanisms could be checked by either comparing the execution traces with the nominal ones, or by checking for the violation of specific constraints.

SW FMEA in CONCERTO provides a useful support to safety engineers in the verification of the software architecture design and in performing SW FMEA, through fault injection at model level. These are important steps of any software safety certification process.

3.4 Addressing domain-specific needs

3.4.1 Petroleum domain

The petroleum use case offered the opportunity to pinpoint and validate the CONCERTO dependability modelling language.

Safety properties in the petroleum domain are those properties that are relevant for assessing risks to personnel, the environment or equipment on a petroleum installation. The state of safety barriers such as hydrocarbon leakage prevention and ignition prevention is of particular interest. Such barriers may include technical components, such as gas detectors and programmable logic controllers, as well as human and organizational components.

Safety analysis techniques supported by CONCERTO were experimented on the petroleum use case. The analysis results are made available to an external decision support system, which uses them to provide advice and information to decision makers on the petroleum installation.

Avionics domain

Experimenting the CONCERTO methodology and toolset in the avionics domain demands conformance with the IMA [4] principles: for this reason CONCERTO provides specialized support for the modeling of partitions, expressed in a model as either top-down or bottom-up artifacts.

Assistance to the user is also provided to determine a partition schedule that conforms with ARINC-653, while requiring as little per-partition specification information as possible (where, for example, a time budget requirement is preferred to a requirement that postulates a specific scheduling algorithm).

In this perspective, CONCERTO also provides the automatic generation of containers and connectors for partitions and processes, which fit the API of an ARINC-653 compliant operating system.

3.4.3 Automotive domain

Regarding the automotive domain, one goal of CONCERTO is to provide modeling and analysis support compliant to the AUTOSAR development flow.

The CONCERTO framework provides support to the modeling of the AUTOSAR application layer, together with the target hardware platform and the software to hardware deployment details.

The SW implementation can be modeled using Synchronous Block Diagrams (SBD), e.g. by using the Simulink commercial tool. The SBD model of the SW implementation can be imported into CONCERTO.

Extra-functional properties can be added to the model; then the CONCERTO toolset can be used for validation of these properties, for instance to calculate the constraints to be applied to threads and tasks based on the end-to-end response time analysis on the modeled SW and HW.

Analysis results can then be used to tune the model's extra-functional properties.

The corresponding AUTOSAR representation of the aforementioned information modelled in CONCERTO can be automatically represented in the AUTOSAR exchange format, i.e., the ARXML (AUTOSAR XML) file will be automatically derived by model transformation. The information represented in ARXML can then be used for automatic generation (not in the frame of the CONCERTO project) of the AUTOSAR run-time environment (RTE) to be executed on top of the target AUTOSAR platform, together with the application layer.

To address compliance with the ISO 26262 functional safety standard, CONCERTO provides support for ASIL association and inheritance, as well as the possibility to verify the correctness of redundant ASIL decomposition.

3.4.4 Telecare domain

Within the telecare demonstrator, we developed a complex source code and configuration generation transformation-chain that is able to synthetize both the gateway and the server layer of single home telecare systems.

Conclusions

The experience with the adaptation of the CHESS methodology and toolset to the industrial use cases in CONCERTO has shown that the original concepts were flexible enough, so that they could be adapted to fit the user needs without giving up their strengths.

A significant delta was developed with respect to the dependability modeling and analysis framework available in CHESS. In addition innovative solutions have been developed for the construction (i.e. design, analysis, implementation and monitoring) of predictable parallel solutions with a good cross-domain coverage.

The project is today approaching the final phase, devoted to the final evaluation of the applicability of the CONCERTO solutions by the industrial partners through the elaboration of several industrial use cases across multiple domains. This activity will receive the support of technological partners, receiving feedbacks necessary for consolidation and use of CONCERTO within the lifetime of the project.

The final step will be the delivery of the CONCERTO enhancements and new developments as open source under the Eclipse PolarSys umbrella.

Acknowledgements

This work is based on the partner's contributions and results of the CONCERTO ARTEMIS-JU project.

References

[1] Universidad de Cantabria, Mast: Modeling and Analysis Suite for Real-Time Applications. http://mast.unican.es/

[2] E. Verhulst, J.L. de la Vara, B.H. Sputh, V. de Florio (2013), *ARRL: A Criterion for Composable Safety and Systems Engineering*, SAFECOMP 2013 Workshops - SASSUR'13.

[3] Aeronautical Radio, Inc. (2003), *ARINC Specification 653-1: Avionics Application Software Standard Interface*.

[4] Radio Technical Commission for Aeronautics (2005), *Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations*.

[5] International Organization for Standardization (2011), *ISO 26262-10. Road vehicles — Functional safety — Part 10: Guideline on ISO 26262*.

[6] B. Gallina, E. Sefer, and A. Refsdal (2014), *Towards Safety Risk Assessment of Socio-technical Systems via Failure Logic Analysis*, 2nd IEEE International Workshop on Risk Assessment and Risk-driven Testing (RISK), pp.287-292, Napoles, Italy.

[7] V. Bonfiglio, L. Montecchi, F. Rossi, P. Lollini, A. Pataricza, A. Bondavalli (2015), *Executable Models to Support Automated Software FMEA*, IEEE 16th Intl. Symposium. on High Assurance Systems Engineering, pp.189-196.

National Ada Organizations

Ada-Belgium

attn. Dirk Craeynest
c/o KU Leuven
Dept. of Computer Science
Celestijnenlaan 200-A
B-3001 Leuven (Heverlee)
Belgium
Email: Dirk.Craeynest@cs.kuleuven.be
URL: www.cs.kuleuven.be/~dirk/ada-belgium

Ada in Denmark

attn. Jørgen Bundgaard
Email: Info@Ada-DK.org
URL: Ada-DK.org

Ada-Deutschland

Dr. Hubert B. Keller
Karlsruher Institut für Technologie (KIT)
Institut für Angewandte Informatik (IAI)
Campus Nord, Gebäude 445, Raum 243
Postfach 3640
76021 Karlsruhe
Germany
Email: Hubert.Keller@kit.edu
URL: ada-deutschland.de

Ada-France

attn: J-P Rosen
115, avenue du Maine
75014 Paris
France
URL: www.ada-france.org

Ada-Spain

attn. Sergio Sáez
DISCA-ETSINF-Edificio 1G
Universitat Politècnica de València
Camino de Vera s/n
E46022 Valencia
Spain
Phone: +34-963-877-007, Ext. 75741
Email: ssaez@disca.upv.es
URL: www.adaspain.org

Ada in Sweden

attn. Rei Stråhle
Rimbogatan 18
SE-753 24 Uppsala
Sweden
Phone: +46 73 253 7998
Email: rei@ada-sweden.org
URL: www.ada-sweden.org

Ada-Switzerland

c/o Ahlan Marriott
Altweg 5
8450 Andelfingen
Switzerland
Phone: +41 52 624 2939
e-mail: president@ada-switzerland.ch
URL: www.ada-switzerland.ch