


CerCo

INFORMATION AND COMMUNICATION
TECHNOLOGIES
(ICT)
PROGRAMME

Project FP7-ICT-2009-C-243881 CerCo

Supplement to Deliverable 6.2
CerCo Workshop and Publication Planning

Version 1.0

Main Authors:
I. Stark

Project Acronym: CerCo
Project full title: Certified Complexity
Proposal/Contract no.: FP7-ICT-2009-C-243881 CerCo

1 Summary

In March 2012 the second-year review panel for the CerCo project requested details of specific plans to promote the final results of the project. This document is a response to that, with the following two contributions.

- Plans for the CerCo workshop at the HiPEAC '13 meeting in Berlin (Deliverables 6.4 and 6.5).
- A list of proposed publications reporting CerCo results.

This report supplements the *Plan for the Use and Dissemination of Foreground*, Deliverable 6.2, from August 2010 and addendum *Future Dissemination Plans* of April 2011.

2 CerCo Workshop

We shall hold a CerCo one-day workshop for academic and industrial researchers. Our preferred destination is *HiPEAC '13*, to be held in Berlin at the end of January 2013. This is an international meeting on high-performance and embedded architectures and compilers, organised by the European Network of Excellence in the field, and as such provides an ideal audience for CerCo results. The conference workshops for 2013 have not yet been selected, but we have submitted our CerCo proposal for consideration. In case this is not successful, we are also holding an offer of a workshop day at ETAPS 2013, the European Joint Conferences on Theory and Practice of Software. This too is a forum for a range of academic and industrial researchers, but it takes place in March 2013 and so does not fall within the CerCo timetable.

This CerCo workshop has two objectives:

- To publicise the results of CerCo to members of the relevant industrial and academic research communities.
- To stimulate discussion between the industrial and academic participants about future work building on the current state of the art.

To support these objectives, we propose to send invitations to a dozen industrial and academic research groups working in the field. While the workshop will be open to anyone at HiPEAC who wishes to register, we consider an invitational workshop to be the most direct way to bring together CerCo researchers with potential users of the project results. A list of possible invitees appears below.

The workshop, with these objectives, is designed to meet the requirements of Deliverables 6.4 and 6.5 as set out in the original CerCo proposal (Annex I "Description of Work", Workplan Table 3, Work package 6). In particular, that proposal envisioned overlapping the presentation of CerCo results to invited industrial and scientific audiences at an event co-located with an international conference.

We plan the following draft structure for the workshop:

- Morning: Short talks by CerCo partners presenting the project and its results.
  - Project overview: a trustworthy framework for cost analysis of embedded code.
  - Demonstration of Frama-C plugin: high-level static reasoning about global runtime complexity.
  - Demonstration of Lustre plugin: automatic analysis of component reaction time.
  - The CerCo compiler: what makes a certified compiler trustworthy.
  - The labelling approach to cost analysis: a modular way to prove correctness.

  - Structured traces: tracking high-level structure in low-level execution.
  - Dependent labels: managing refined cost information for richer architectures.
  - Future directions: the latest very-low-power embedded processors; other runtime costs; further languages.

- Afternoon: Short presentations by some of the invited participants on the state of the art in static analysis of execution costs for embedded systems.

- Conclusion: Round-table discussion on potential applications and future challenges.

This structure addresses both of the listed objectives, with the morning dedicated to dissemination of CerCo results and the afternoon to promoting interaction between academic and industrial researchers working in the field.

We have identified the following relevant groups as some initial invitees to the CerCo workshop.

AbsInt aIT *AbsInt* provide tools for validation of safety-critical software, including static analysis for worst-case execution time through abstract interpretation. They have participated in a previous CerCo event, and their insights into the requirements for precise timing analysis are extremely relevant to the domain of CerCo.

Frama-C The *Frama-C* platform for static analysis of C source code provides the context for the CerCo plugin which demonstrates the use of automated cost annotations on source code.

Rapita Systems The *RapiTime* timing analysis tool for real-time embedded systems uses runtime instrumentation of C to identify source-level execution costs and predict worst-cast execution time.

PASTA This research group at the University of Edinburgh are the designers of the *EnCore* family of configurable embedded microprocessors. Their *ArcSim* tool offers cycle-accurate simulation of these low-energy cores, giving significant insights into the practicalities of precise cost analysis.

WCET-Aware Compilation Dr Heiko Falk at TU Dortmund leads this research project investigating the optimizations and compilation techniques appropriate to the requirements of worst-case rather than average-case timing.

York RTS The *Real-Time Systems* research group at the University of York are active in worst-case execution time analysis and the design of time-predictable architectures.

ARM Verification Project Gordon, Fox and Myreen at the University of Cambridge have a machine-checked mathematical model of the ARM microprocessor. With the release of the ARM Cortex-M series of low-power embedded processors, this is a potential application area for CerCo technology in the future.

CompCertTSO Sewell's group, also at the University of Cambridge, have a verified C compiler treating concurrency in a relaxed memory model. Like CerCo, they have relevant experience of extending CompCert-based verification to additional code properties.

Gliwa Embedded Systems The software consultancy *Gliwa GmbH* develop specialist tools for measurement and verification of execution time in embedded systems.

Artemis The European industry association *Advanced Research & Technology for Embedded Intelligence and Systems*. We already have contact with Artemis through the University of Bologna, which is a member of the association.

IMDEA Software Dr Boris Köpf applies quantitative analysis of code execution time to evaluate potential security weaknesses by side-channel attacks.

ADSIG The *ArtistDesign Special Interest Group on Embedded Systems Design* is a large consortium of experts in embedded systems design, arising from the ARTIST Network of Excellence.

We propose to invite each of these groups to send participants and, if they wish, contribute a short talk on their work to the afternoon session. We shall also invite the CerCo project reviewers, who are experts in the area. This is not a closed list — depending on responses, we are open to inviting further participants as appropriate.

3 Publication Planning

The original Deliverable 6.2 and its addendum, prepared earlier in the project, presented a list of conferences and journals suitable for publicising CerCo results. Now that the project has progressed, we have reviewed the technical activities to date and identified a specific list of topics and results for dissemination in individual papers and articles. Some of these have already been submitted, or accepted for publication: in particular, an FMICS 2012 paper on the labelling approach which also references the Frama-C plugin and Lustre case study.

Labelling As presented in Deliverable 2.1, the use of labelling to build cost annotations and prove their correctness is key to the CerCo development. It allows decoupling the certification of machine code execution properties from the verification of the compilation map between C source and assembler.

Dependent labelling The indexing of labels with additional data-dependent information makes is possible to track sources of variation in execution cost: compiler optimizations such as unrolling or loop peeling; or processor features like instruction pipelining and caches.

The CerCo Frama-C plugin This was demonstrated at the second-year review panel, and a paper would describe its operation and effect. The plugin annotates C source code with fine-grained cycle costs according to CerCo analysis. This instrumented code is passed to the *Jessie* plugin which can generate the verification conditions necessary to check the cost specifications of complete procedures. Finally, these conditions can be passed to a range of automatic proof engines. The overall result is verified declarations of the cycle execution time of identified C functions, in terms of input and other parameters, given in the ACSL specification language.

Lustre analysis This was also demonstrated at the review panel. The CerCo tools can perform automated analysis of response time for individual components in the Lustre synchronous control language. Lustre automatically generates C source code for concurrent components, in a sufficiently standard form to allow precise costing of execution paths.

Certifying machine code CerCo extends previous work on compilation by also verifying the final step from assembler to binary object code. In addition, it is essential to prove that cost annotations exactly capture machine execution steps. These are both substantial proofs with novel features.

Correctness of branch selection Many processors provide a range of branch commands at the machine level, typically for different ranges of jump. Generating efficient and

compact assembly code requires appropriate branch selection. This is non-trivial — indeed, finding the optimum selection is in some cases NP-hard — and the complexity of any algorithm here makes it a challenge for proving correctness. CerCo have developed a novel decomposition of branch selection into policy and implementation, to separate concerns and allow for tractable proof.

Structured traces CerCo reports execution costs at the level of C source, where there is an evident code structure of nested call and return; but these costs arise from unstructured individual steps at the machine level. We have developed a notion of structure on machine execution traces which captures precisely the information necessary to correctly relay cost information between these levels. This is essential to the compiler proof, and interesting in itself as a carrier of high-level information in a low-level execution environment.

The CerCo formalisation The large-scale structure of the CerCo formalisation, its components, how they depend upon each other, and what is required of them to complete the correctness proof. Some parts of these are related to analogous constructions in work such as CompCert; many are novel and arise from our particular attention to verifying non-functional runtime properties of code.

A survey of CerCo results All the above papers deal with individual parts of the CerCo project, or the technical details of its operation. A final paper should survey the outcome of the project, its achievements and potential future work to build on these.