IST-004527 ARTIST2
Network of Excellence
on Embedded Systems Design

Activity Progress Report for Year 3

JPIA-Platform

Timing Analysis Platform

Clusters:

Compilers and Timing Analysis

Activity Leader:

Prof. Dr. Reinhard Wilhelm (Saarland University)
http://rw4.cs.uni-sb.de/people/wilhelm.shtml

Policy Objective (abstract)

Combine the best components of existing European Timing-Analysis tools and prototypes in a standard tool architecture with well-defined textual interfaces. Our objective is to integrate European efforts on the Timing Analysis of Real-Time Systems, to preserve the existing lead of European Research and Industry in this important sector.

The resulting platform will be used in teaching the technology all over Europe.

Table of Contents

1. Overview of the Activity

1.1 ARTIST Participants and Roles

Prof. Dr. Reinhard Wilhelm – Saarland University (Germany)

Activity Leader, Compiler design, Static Program Analysis, Timing Analysis

Saarland University has developed much of the timing-analysis technology that is further developed and commercialised by its spin-off company AbsInt. Saarland University is coordinating the integration activity.

Dr. Christian Ferdinand – AbsInt GmbH (Germany)

Leading Tool Supplier

AbsInt provides advanced WCET analysis tools for a wide variety of targets. The work within Artist 2 focuses on the advance of WCET analysis techniques by providing and defining interchange formats for the components of WCET tools like AIR (Artist 2 Intermediate Program Representation for WCET tools).

Dr. Guillem Bernat – University of York (UK)

Tool Supplier

The contribution of University of York is on measurement-based WCET analysis where its focus is on instrumentation methods, coverage analysis, and timing documentation.

Prof. Dr. Björn Lisper – Mälardalen university (Sweden)

Timing analysis tools

Mälardalen University is working on automatic flow analysis, WCET analysis case studies on industrial code, the maintenance of a WCET-benchmark suite, the definition of interface formats for timing analysis, and the use of WCET tools in education.

Dr, Jan Gustafsson – Mälardalen university (Sweden)

Timing analysis research and tools, WCET analysis case studies

Dr. Andreas Ermedahl – Mälardalen university (Sweden)

Timing analysis research and tools, WCET analysis case studies

Prof. Dr. Peter Puschner – TU Vienna (Austria)

Timing-Analysis Tools and temporally predictable HW-SW architectures

Within the Timing-Analysis Platform Activity TU Vienna focuses on the test-data generation for measurement-based WCET analysis, the definition of the interchange representation for WCET-related information, and on time-predictable architectures*.*

Dr. Niklas Holsti – Tidorum Ltd. (Finland)

Timing-Analysis Tools

Tidorum Ltd supplies the timing analysis tool Bound-T. Tidorum takes part in the definition of the architecture of the tool platform and in particular in the definition of the interchange representation, AIR. Later, Tidorum will integrate Bound-T with the platform by adding AIR export and import functions.

Prof. Dr. Peter Marwedel – Dortmund University (Germany)

Architecture-aware compilation, low-power code generation, Development of optimizations for WCET minimization.

Dr. Stephan Thesing, Saarland University (Germany)

Research on Timing Analysis

Dr. Raimund Kirner – TU Vienna (Austria)
Timing-analysis tools and *compilation with support for timing analysis.*

Oleg Parshin, Saarland University, Saarbrücken (Germany)
PhD student, research on the integration of code synthesis, compilers, and timing analysis and on timing predictability of virtual memory systems

Jan Reineke, Saarland University, Saarbrücken (Germany)
PhD student, research on Timing Analysis and Timing Predictability.

Sebastian Altmeyer, Saarland University, Saarbrücken (Germany)
PhD student, research on Timing Analysis and Scheduling.

Gernot Gebhard, Saarland University, Saarbrücken (Germany)
PhD student, research on Timing Analysis for Multi-Core Architectures.

1.2 Affiliated Participants and Roles

Dr. Isabelle Puaut – IRISA (France)
Timing-Analysis Tools

1.3 Starting Date, and Expected Ending Date

September 1st, 2004 until the European timing-analysis platform has been constructed.

1.4 Baseline

Europe is leading this field. The only commercially available WCET tools, aiT[1], Bound-T[2], and RapiTime[3], and most of the academic prototypes, Heptane[4], SymTA/P[5], SWEET[6], and the tools of TU Vienna[7], are of European origin.

The commercial tools and the academic prototypes follow different approaches, namely analytical and measurement-based approaches. Each individual approach has to solve essentially the same set of sub-problems. The differences between the approaches lie in the methods used to solve some of the subproblems. Methods for some subproblems can be combined across approaches.

The timing-analysis problem has been essentially solved. Projects for the development of time-critical systems in industry use tools of ARTIST2 partners. However, there is room for improvement. The usability can be improved by reducing the necessary amount of user interaction. The efficiency of the tools can be improved by improving our knowledge about the underlying processor architecture. The tool realization can be simplified by computer support. The different tools offer different strengths in the areas of different subproblems. The goal is to combine the respective strengths.

[1] http://www.absint.com/ait/

[2] http://www.tidorum.fi/bound-t/

[3] http://www.rapitasystems.com/wcet.html

[4] http://www.irisa.fr/aces/work/heptane-demo/heptane.html

[5] http://www.ida.ing.tu-bs.de/projects/symta/home.e.shtml

[6] http://www.mrtc.mdh.se/projects/wcet/

[7] http://www.wcet.at/

Proposals for a modular framework allowing the integration of prototypical developments existed before the start of the NoE. Documented and supported interfaces existed for the individual tools, but not for any combination of components from different tools.

In an ongoing discussion between the participants about WCET-tool architectures, interfaces, and integration agreement on interfaces has been reached.

Traditionally, timing analysis tools have been designed independently of compilers. It has now turned out that proceeding along this path would result in a duplication of efforts. Flow facts are available in compilers and need to be regenerated in timing analysis tools. Timing information is available in timing analysis tools and would be useful for timing-aware optimizations in compilers. Currently, compilers use very rough approximations of timing, if they use any timing model at all. As a result, the impact of certain transformations on the run-time is frequently not known by the compilers. Hence, the user has to follow a trial-and-error approach, experimenting with different compiler options and figuring out a suitable combination of them. However, even this time-consuming process cannot really minimize the execution time since options which might be good for some part of the code might lead to bad result for some other part of the code. A tight integration of timing models into compilers and their optimizations is urgently needed.

1.5 Problem Tackled in Year 3

The ARTIST2 interface language, AIR, for the exchange of results and components of timing-analysis tools has been further defined, implemented and documented. The various components of a program representation in AIR may carry attributes. To manage these attributes, an attribute data base was established on a web server accessible by all ARTIST2 partners. A group mechanism was introduced. Subsets of attributes can be associated with a group of users, who define and use these attributes.

Transformations between AIR and the internal representations of other tools have been implemented.

Several of the tool providers needed a language for the specification of the semantics of a machine instruction set. The language ALF developed at Mälardalen was identified as the most promising candidate.

An activity was started to identify flow description attributes that could serve as an exchange format among different tool components. As a result within year 3, existing flow description languages have been analysed and described.

A WCET Tool Challenge has been initiated and performed. All WCET tool providers were invited to participate. The goal was to produce a realistic picture of the state of the art and of the individual strengths and weaknesses of the different tools and to identify further necessary developments. The Challenge has been executed by an external test person. The results have been published. The Challenge will be repeated every two years with improved procedures and more challenging benchmarks and conditions.

Timing analysis needs to be integrated into compilers. Based on the work on a timing-aware compiler infrastructure performed in Year 2 of Artist2, year 3 dealt with studies of standard compiler optimizations developed to reduce the program’s average-case execution time and their influence on the WCET. As an outcome of the current reporting period, it could be shown that the compiler optimization Procedure Cloning allows improved WCET estimations since it makes the code more predictable. Additionally, WCET-aware memory hierarchy exploitation using locked instruction caches was studied. The work on timing-aware compiler optimizations was published on international conferences and workshops of very high quality.

Several industrial case studies were performed to gain more experience with industrial requirements and how they are matched with the current state of the tools.

1.6 Comments From Year 2 Review

1.6.1 Reviewers' Comments

"This document has been internally reviewed which is good. This technique should be generalised for other deliverables."

2. Summary of Activity Progress

2.1 Previous Work in Year 1

2.1.1 Work achieved in the first 6 months

We started with an extensive discussion of potential tool-interface languages. Such a language needs to offer the representation of object programs with an adequate attribute mechanism to store annotations about feasible and infeasible paths for program execution and to present analysis results at the program-source level in sufficient detail to be easily understandable by the tool user.

The prime candidate for such a language was CRL2, the interface language of AbsInt's timing-analysis tool aiT. CRL2 is the result of long evolution of intermediate representations in analysis frameworks of AbsInt and Saarland University.

The commercially-supported CRL2 format was interfaced with several of the components needed for the planned Timing-Analysis platform, i.e., parts of the analysis tool chain made ready to communicate via CRL2. With the selection of CRL2 a robust and generic interface was introduced into the ARTIST2 framework.

CLR2 is a generic and processor independent format usable for static analysis (including WCET analysis), optimisation of machine code and assembly language. It supports the integrated representation of control flow graph and intermediate analysis results. An efficient C/C++ library reads/writes CRL2 interface files in a text-representation format and provides an API to the data structures used by the components of the timing-analysis tool suite.

CRL2 has an interface to the Program-Analyzer Generator developed at Saarland University, such that interprocedural analyses are easily implemented.

For the given reasons, CRL2 was first chosen for the integration of various work groups' analyses. Several partners in the Timing-Analysis cluster started to interface with CRL2. Experiments at IRISA and Tidorum with the CRL2 library of AbsInt showed good results. In particular, preliminary experiments made at IRISA used CRL2 to add a tree structure on top of the control flow graph supported CRL2 library. The experiments showed that the CRL2 library, through its attribute system, is flexible enough to define data structures used by WCET analysis methods different from those implemented in aiT.

Based on the first successful results with CRL2 it was discussed that the interchange format should not only be used for timing analysis but should also serve as a compiler-analyser interface, thus facilitating the WCET-aware compilation of code. The interface language is called AIR, for Artist 2 Intermediate Representation. AIR is based on CRL2. CRL2 is a dialect (superset) of AIR as explained in section 2.2.1. Further, it was found that in order to suit the needs of the partners of the CTA cluster, a text format of the interchange format had to be defined and the documentation should be extended (until then CRL2 was available as a library only). AbsInt offered to provide this format definition and documentation.

Besides the work on the common interchange format for timing analysis, the partners of the CTA cluster accomplished the following:

- Students from Mälardalen University performed a number of industrial WCET analysis case studies in Swedish enterprises, mostly with AbsInt's tool aiT. The case studies showed that for common embedded processors tight WCET bounds could be obtained. However, the effort needed to provide the required program-flow information by hand turned out to be high and time consuming.

- Mälardalen and Vienna developed a prototype editor plug-in that helps the programmer to achieve more predictable timing and aims at simplifying WCET analysis. By highlighting code segments that are executed conditionally, i.e., that are executed only for a subset of all possible inputs to a piece of code, the editor helps the programmer identify possible sources of timing variability. Avoiding such input-dependent conditionals, or at least minimising them in length, leads to code that in general has fewer execution paths, a smaller execution-time jitter, and is thus easier to analyse for its WCET.

2.1.2 Work achieved in months 6-12

The major focus of the activity was on the further development of the common interchange format for WCET analysis:

We identified one important area that was not explicitly represented in CRL2: the “instruction set semantics”, by which we mean the computation that is performed by each instruction in the program. As CRL2 has been used so far, the control-flow graph identifies the instructions and their operands, but an analysis tool must itself have enough knowledge of the target processor to map the instruction identifier and operand identifiers to the computation, for example to understand that the instruction adds two registers and puts the sum in a third register. This computation, connected to the control-flow graph, is the essential information that the tools from Mälardalen and Tidorum use to find constraints on execution flow, such as loop bounds. We therefore decided to define an explicit and general representation of this instruction set semantics as one part of AIR, using the flexible attribute mechanism from CRL2. Work on this extension is under way.

Peter Marwedel's group at Dortmund University used CRL2 to interface their compiler with the aiT tool. Similarly to earlier activities, one of the results was that a better documentation on aiT’s usage of attributes would be useful.

While AbsInt and Saarland University were working on extensions and improvements of CRL2, feedback from Rennes, Mälardalen and Tidorum asked for a formalisation of the control flow graph structure and some further extensions of the interchange format.

In parallel, Vienna and Mälardalen started work on extending the path-annotation support of CRL2. As for the instruction-set semantics, the plan was to use the attribute mechanism of CRL2 to represent these annotations about feasible and infeasible execution paths to facilitate a highly accurate modelling of the possible execution paths for WCET analysis. Inputs from Vienna, Mälardalen, Tidorum, and AbsInt for these path annotations have been collected and summarized in a presentation.

Though each of the activities was very promising by itself, the consortium had to find that the definition of such a complex interface as the WCET interchange format requires and would yet require a lot of further work till its final completion.

Besides the core work on the common format the partners of this activity reported about the following work related to timing analysis:

- Tidorum and Mälardalen cooperated on implementing a version of the Bound-T WCET analysis tool for the Renesas H8/300 processor, which can be found in the popular Lego Mindstorms kit. Mälardalen is now using this tool in real-time systems education.
- Licensing policies and the issue of securing the availability of AbsInt’s CRL2 library had been discussed. We are looking for a full and open definition of the syntax and semantics of AIR so that anyone can build their own AIR libraries.
- Vienna and York started to cooperate on measurement-based timing analysis. The partners had been working individually on measurement-based analysis before. York

uses measurement-based WCET analysis for non safety-critical applications and has a strong focus on code instrumentation and report formats for representing measurement details to the user. In Vienna, the use of measurements is seen as a complement to static analysis for the purpose of validating static-analysis results. In Vienna, the focus is on the automatic test-data generation. The cooperation started within the CTA cluster aims at combining the efforts and exchanging the complementary know-how of the groups. In particular, the partners started to work on the definition of coverage criteria for measurement-based WCET analysis.

- The industrial WCET case studies performed by students from Mälardalen University in Swedish enterprises were continued. The findings essentially confirmed the conclusions from the earlier studies.
- Representatives of AbsInt, Tidorum and Mälardalen all demonstrated their WCET tools and participated in the Real-Time in Sweden (RTiS) conference, held in Skövde, Sweden, Aug 2005. This was a joint effort to introduce the concept of static timing analysis to the Swedish companies that participated in the RTiS conference. See: http://www.snart.org for details about RTiS.

A number of publications have been produced as a result of the cooperation inside the cluster. They are included in the list of publications collected and submitted by the coordinator.

2.2 Previous Work in Year 2

Definition of AIR (Artist2 Intermediate program representation for WCET tools)
In the past few months, a file format specification was developed to define the AIR ('ARTIST2 Interchange') format that may be used by the cluster participants' tools for integration. So AIR is the proposed exchange format of the tools of the groups participating in the ARTIST2 project. The format is based on CRL2, which is the successor of CRL. These formats were originally developed in cooperation by Saarland University and AbsInt Angewandte Informatik GmbH over several years of work.

The idea behind AIR is that an interface is to be defined on the file-format level, in contrast to CRL2, whose interface definition only covers the C++ library interface. Internally in AbsInt tools, a specification of the C++ library interface is preferred over a file format specification, simply because all tools use the library and thus the storage on disk is secondary For ARTIST2, different work groups prefer their own libraries over the usage of proprietary software, so there is a serious demand for a file format specification.

Since CRL2 was not primarily meant to be a file format, much work had to be done before this document could be written. Apart from the mere documentation the file format had to be defined and implemented. In order to get a stable interface on file level CRL2 had to be extended. For example, version numbers and specification IDs had to be added to meet the strict safety criteria of real-time systems analysis. Thus, this document can be viewed as the first step of the final documentation phase in a larger effort towards an exchange file-format for the different WCET tools and tool components used within this ARTIST2 activity.

From the release of the first AIR specification on, the CRL2's file format interface will be a dialect of the AIR file format. CRL2 as well as dialects of other work groups are allowed to feature extensions as long as they are not vital for the operation of the tools. E.g., AbsInt tools will only use the plain AIR file format during normal operation, the extensions of CRL2 are mainly implemented for debugging and diagnosis purposes. In the same way, extensions of other dialects shall never be vital to the operation of the corresponding tools
http://www.absint.com/artist2/doc/crl2/air.pdf

Timing-Analysis Survey paper

The work on the survey paper about Timing-Analysis Methods and Tools has clarified the characteristics, the advantages and disadvantages, and the application domains for the different approaches, e.g. analytical and measurement-based approaches. It has clarified the modularisation of the overall timing-analysis task and the possibility of combining modules for different subtasks across the approaches. The joint authorship of this paper expresses strong and enduring cooperation between the different groups. This paper will represent a landmark publication for the area!

Timing Anomalies Characterisation and Checking

Timing Anomalies in processors produce counter-intuitive timing behaviour, i.e., local worst-case behaviour does not necessarily lead to global worst-case behaviour. The existence of timing anomalies requires complex timing-analysis procedures. Saarland University together with Freiburg University have worked on the clarification of the concept and the origins of timing anomalies. The goal is an automatically checkable definition of timing anomalies, which would allow for a safe reduction of the WCET analysis effort whenever the absence of timing anomalies can be shown for a processor platform. Furthermore, it was found that certain cache-replacement strategies lead to Domino Effects, which are timing anomalies without bounds for their effects. This work is funded by the Transregional Research Centre AVACS (Automatic Verification and Analysis of Complex Systems) of the Deutsche Forschungsgemeinschaft.

Parametric WCET Analysis

The runtime of programs might depend on parameters. In these cases the worst case execution time (WCET) has to be recomputed for each parameter assignment. This can be very time consuming. On the other hand the relation between parameters and WCET cannot be easily identified.

Saarland University together with Mälardalen University have initiated collaboration about this type of parametric WCET analysis. Two M.Sc. students from Saarland visited Mälardalen in early 2006 to learn about the Mälardalen approach.

In the joint approach a parametric WCET analysis based on the aiT-tool chain is performed. It computes a WCET formula instead of a concrete value. Since programs spend most of their runtime in loops, we focus on a parametric loop-bound analysis. Prior to this part the parameters of the executable have to be determined. In the path analysis part, a parametric optimisation method is needed. Afterwards the resulting formula has to be evaluated. Evaluation in this context means visualisation or instantiation of the formula.

WCET analysis benchmark suite

Mälardalen University has collected a suite of benchmark programs for WCET analysis. The suite is maintained on the web by Mälardalen. One of the purposes of this benchmark suite is to be able to evaluate and compare different WCET tools as is done in the initiated WCET Tool Challenge (cf. http://www.mrtc.mdh.se/projects/wcet/benchmarks.html)

Input format for flow analysis

Mälardalen University has initiated work to define a standard input code format “ALF” for flow analysis. The purpose is to facilitate flow analysis of codes in different formats by translating to ALF. ALF will provide an interface to Mälardalen’s flow analysis. A first draft for ALF exists, and it will be disseminated within the cluster before the format is finalised. ALF should also be harmonised with the AIR instruction semantics format.

Synergy between Code Synthesis and Timing Analysis

Saarland University together with AbsInt and ETAS have integrated the ASCET-SD code-synthesis with AbsInt’s timing-analysis tool to improve usability of the timing-analysis tool and precision of the results.

Timing Predictability
First quantitative results have been obtained on the influence of architectural properties on the timing predictability of embedded systems. In particular, four different cache replacement policies and their influence on predictability have been considered at Saarland University. This research is funded by AVACS. On the side of TU Vienna, a model for a time-predictable processing node has been worked out – on this node software timing behaviour can be predicted with the granularity of the CPU clock. This node uses a purely time-triggered input-output interface and relies on single-path code (code that is free from input-data dependent control flow) in both the operating system and the application code. Tasks are only preempted at pre-planned task preemption points and simple clock synchronization keeps the operations of the nodes in synchrony with its real-time environment. The work on the time-predictable node yielded a time-predictable task-preemption model where an instruction counter instead of the CPU clock is used to implement preemptions (It was shown that CPU-clock based pre-emption may lead to unpredictable timing).

Measurement-Based WCET Analysis
As the complexity of WCET analysis varies with the structure of the program to be analysed and the type of target hardware, TU Vienna and York worked out a detailed list of issues for measurement-based WCET analysis. This list of issues is to be used for different purposes: First, it is a check list for the designer of a WCET analysis tool. Second, it gives the system developer clues about relevant hardware and software criteria when designing a system with the goal of simple analysability and predictability. The list is divided into three categories: a) issues that only relate to the software of the system, b) issues that address only the target hardware of the system, and c) issues that are relevant for both, the software and the hardware part of the system. The partners used these results as a starting point for the work on coverage criteria for measurement-based WCET analysis that was initiated in this work period. The goal of this work item is to find meaningful metrics for assessing the timing-related code coverage and the value of input-data sets for the measurement-based analysis.
The results are documented in a technical report, which is available at:
http://www.vmars.tuwien.ac.at/php/pserver/extern/docdetail.php?DID=1975&viewmode=published&year=2007

2.3 Current Results

2.3.1 Technical Achievements

Timing Analysis and Timing Predictability (USaar and AbsInt)
Timing Anomalies are a difficult problem for timing analysis. They complicate the design of tools; they increase the necessary analysis effort, and the decrease the precision of the results. No real understanding of the concept existed, only observational properties were known. A new definition has been produced that helps to understand this phenomenon. It covers both scheduling as well as speculation anomalies.

The large state space to be traversed during the pipeline analysis largely determines the necessary analysis effort. An approach has been followed to use symbolic representations of this state space to increase efficiency of the analysis.

The work on predictability has continued and first hard analytical results about predictability of architectural features have been obtained, in this case cache replacement strategies. These show that the replacement strategy has a strong influence on the precision of any type of cache analysis.

The formal derivation of abstract processor timing models has been mostly implemented. This process starts from a specification of the hardware architecture in VHDL and proceeds by a series of analyses and transformations. Analyses of such models for several kinds of properties will be possible once formally derived abstract architectural models are available.

Preemptive scheduling of hard real-time tasks requires precise estimations of context-switch costs. These are largely dependent on the cache-refill costs caused by pre-empting tasks. An approach has been developed and implemented that estimates and even minimizes the cache interference of tasks. The latter optimization uses the memory allocation to define the cache mapping.

Synergy between Code Synthesis and Timing Analysis (USaar and AbsInt)

An integration of AbsInt's aiT timing-analysis tool with the ASCET specification and synthesis tool of ETAS has been realized, and experimental results about the effect have been produced (ISoLA 2006 Paper, see 2.3.4).

ARTIST Interchange Representation and Attribute Database: AIR (AbsInt)

The work on AIR has continued. The syntax specification produced in Year 2 is now accompanied by a data-base that defines the current set of "attributes" that carry much of the inputs and outputs of the analysis as decoration on the extended control-flow graph. A data base for managing these attributes was designed and implemented. The data base specifies attribute names and types, their location (e.g. at routines or at instructions) and their access rights (which tools may introduce/read/write which attributes). The attribute data base was established on a web server and is accessible by all ARTIST2 partners. The database interface is designed to be open to allow the definition of further attributes by other ARTIST participants.
http://www.theiling.de/absint/attrdb.fcgi

Computation Semantics Representation: ALF (USaar, Tidorum, Mälardalen, AbsInt)

To support the flow analysis (loop bounds and infeasible paths) AIR must be able to represent the computations executed by the instructions in the program under analysis. CRL2, the basis of AIR, does not have an explicit, portable computation semantics representation. Work on such a representation started in Year 2 and continued in Year 3. The most advanced candidate is the ALF language from Mälardalen University. Mälardalen produced a preliminary specification of the ALF syntax and semantics
http://www.mrtc.mdh.se/index.php?choice=publications&id=1351

WCET Challenge 2006 (Mälardalen, USaar, AbsInt, Tidorum, National University of Singapore, TU Vienna together with Christian-Albrechts University Kiel, DaimlerChrysler Research Ulm, University Duisburg-Essen, and University Stuttgart)

The purpose of the WCET Tool Challenge was to be able to study, compare and discuss the properties of different WCET tools and approaches, to define common metrics, and to enhance the existing WCET benchmarks. Four WCET tools (two commercial, two research prototypes) completed the Challenge. The WCET Tool Challenge was performed during the autumn of 2006 and resulted in two papers at ISoLA 2006 (one of them mentioned under 2.3.2, the other written by an external test person). The WCET Tool Challenge was also presented at the WCET workshop 2007 (see 2.3.2).

Based on our experience with the WCET Challenge 2006 we plan to hold future challenges bi-annually with results presented at the International WCET Workshop. The next challenge will be run in the fall of 2007 and spring of 2008, with a more formal and rigorous structure and hopefully more participants. The results will be presented at WCET-2008.
http://www.idt.mdh.se/personal/jgn/challenge/

Transformation of Flow Information during Compilation (TU Vienna/Real-Time Systems Group + TUV/Programming Languages Group)

Flow information has to be used in general to guide the worst-case execution time analysis. When given such extra flow information manually, it is most convenient to give them at source code level. However, when compiling the code the compiler may change the control-flow structure of the program, thus rendering the original flow information as invalid. Thus, it is

necessary to update the flow information in parallel to the compiling of the code. The Real-Time Systems Group of TU-Vienna participating in the Timing Analysis platform and the Programming Languages Group participating in the Compiler platform started a cooperation on developing such a flow information transformation framework. The research is funded by the Austrian Science Fund (Fonds zur Förderung der wissenschaftlichen Forschung) within the research project “Compiler-Support for Timing Analysis" (COSTA). COSTA started in July 2006 and has duration of three years. A first prototype that is performing source-to-source transformation of flow information has been already implemented.
http://ti.tuwien.ac.at/rts/research/projects/COSTA/

Common Flow Description Attributes (TU-Vienna/Real-Time Systems Group, TU-Vienna/Programming Languages Group, Mälardalen University, AbsInt, University of York)
To compute the WCET of program in general, additional flow information is needed to guide the WCET computation. Due to different research approaches and framework architectures, there exist many different flow description languages for WCET analysis, making it hard to connect different components of WCET analysis frameworks together. Within ARTIST2 the partners TU-Vienna and Mälardalen University together with the industrial partners started an effort with the ambitious goal to define flow description attributes that serve as an exchange format among different tool components. As a result within year 3, existing flow description languages have been analysed and described by TU-Vienna. The next activity will be to set up a web page as an information exchange portal for a “challenge on flow description languages”. The research is partially funded at the TU Vienna by the Austrian Science Fund (Fonds zur Förderung der wissenschaftlichen Forschung) within the research project “Compiler-Support for Timing Analysis" (COSTA). COSTA started in July 2006 and has duration of three years. The next steps of the ARTIST2 partners working on the flow description attributes will be to collect feedback received from this challenge on flow description languages, and to define a common set of flow information. Based on this common set of flow information a set of properties to be embedded as attributes into the AIR format will be defined.
http://ti.tuwien.ac.at/rts/research/projects/COSTA/

Four partners of the team (Vienna, Mälardalen, Tidorum, and AbsInt) continued to work on path description attributes for AIR to arrive at a uniform notation. This work led to the conclusion that a more detailed study and comparison of the possible formats is necessary. This study and comparison was done by TU Vienna. The results of this study have been documented and published. As a next step, it is planned to set up a challenge of flow-description languages to get a broad feedback about proper solutions for that problem domain. (*Due to the need for further investigations, the integration of new path description attributes into AIR has been moved beyond the scope of ARTIST2*).

Evaluation of SWEET and aiT at Volvo CE (Mälardalen, Volvo CE and AbsInt)
This evaluation was performed by Dani Barkah, student at KTH (The Royal Institute of Technology, Stockholm), as a Master’s Thesis work. Researchers from Mälardalen University supervised Barkah. The purpose of the work was to test the automatic flow analysis (as described in the RTSS 2006 paper; see 2.3.2) of the prototype WCET tool SWEET (SWEdish Execution time Tool) from Mälardalen University on real industrial code. A second purpose was to compare the automatic flow analysis of SWEET to manual flow analysis made in an earlier Master’s Thesis work (described in Daniel Sehlberg, et al. Static WCET Analysis of Real-Time Task-Oriented Code in Vehicle Control Systems. the ISoLA-06, Cyprus, Nov. 2006) paper; see 2.3.4). Yet a purpose was to run the WCET tool aiT using the results from SWEET. The title of Barkah’s Master’s Thesis is “Evaluating program flow analysis for WCET calculations at Volvo CE”. The work has been finished in August 2007.
Dani Barkah and Nils-Erik Bånkestad (head of the software development at Volvo CE in Eskilstuna, Sweden) visited AbsInt in April 2007 as a part of this work.
http://www.mrtc.mdh.se/index.php?choice=publications&id=1341

WCET Analysis and Certification of Automatically Generated Code (Mälardalen, CC-Systems AB and Tidorum)

This work was performed by the Mälardalen student Elie Assaf as a Master's Thesis work. The commercial static WCET Tool Bound-T was used to conduct a WCET analysis on parts of a C++ code that was automatically generated from the component-based tool Rhapsody. The C++ code was generated for a hard real-time application that runs on a bridge control panel installed in a Rolls Royce marine vessel. Researchers from Mälardalen University as well as Niklas Holsti from Tidorum AB, Finland, supervised Assaf. The title of Assaf's Master's Thesis is "WCET Analysis and Certification of Automatically Generated Code for CC-Systems AB". The work finished in August 2007.

http://www.mdh.se/ide/eng/msc/index.php?choice=show&id=0628

Conversion of the AbsInt AIR format to SWEET format (Mälardalen and AbsInt)

This work is performed by the Mälardalen student Per Wolde as a Master's Thesis work. Researchers from Mälardalen University are supervising Per Wolde. This work will allow SWEET to analyse binaries for the NECV850 processor using AbsInt tools like AiT or "exec2crl". The chain of files and tools is: C-source → NEC_gnu_cross_compiler → exec2crl → crlreader → SWEET. Per Wolde (together with a group of MSc and PhD students from Mälardalen University) visited AbsInt in the winter of 2006/2007 as a part of this work. This is on-going work.

http://www.mdh.se/ide/eng/msc/index.php?choice=show&id=0536

Transformation of Flow Facts within Optimizations of a WCET-aware Compiler (Dortmund University)

Timing analysis relies on the presence of highly precise flow facts. Flow facts represent information about the possible flow of control through a program under analysis – e.g. iteration counts of loops. Usually, such information is provided by the designer who is fully responsible for its correctness. In the context of the WCET-aware compiler developed at Dortmund in the past years, flow facts can now be entered into the compiler within the source code to be processed by the compiler. These flow facts are analyzed and kept semantically correct during each transformation performed by the compiler. In particular, all flow facts are maintained and adjusted during all compiler optimizations, even if they heavily restructure the code. These automatically transformed flow facts are finally passed to the WCET analyzer aiT provided by AbsInt, in order to perform the actual WCET analysis. This achievement has taken away the burden from the designer to specify flow facts at the assembly code level. Now, the designer can annotate the source code, invoke the compiler, perform optimizations, and still obtains valid WCET results.

http://ls12-www.cs.uni-dortmund.de/

Compile-Time Decided Instruction Cache Locking Using Worst-Case Execution Paths (Dortmund University, AbsInt)

Caches are notorious for their unpredictability. It is difficult or even impossible to predict if a memory access results in a definite cache hit or miss. This unpredictability is highly undesired for real-time systems. The Worst-Case Execution Time (WCET) of software running on an embedded processor is one of the most important metrics during real-time system design. The WCET depends to a large extent on the total amount of time spent for memory accesses. In the presence of caches, WCET analysis must always assume a memory access to be a cache miss if it can not be guaranteed that it is a hit. Hence, WCETs for cached systems are imprecise due to the overestimation caused by the caches.

Modern caches can be controlled by software. The software can load parts of its code or of its data into the cache and lock the cache afterwards. Cache locking prevents the cache's contents from being flushed by deactivating the replacement. A locked cache is highly predictable and leads to very precise WCET estimates, because the uncertainty caused by the replacement strategy is eliminated completely.

In year 3 of Artist2, the lockdown of instruction caches at compile-time to minimize WCETs was

explored. In contrast to the current state of the art in the area of cache locking, our techniques explicitly take the worst-case execution path into account during each step of the optimization procedure. This way, we can make sure that those parts of the code are locked in the I-cache that leads to the highest WCET reduction. The results demonstrate that WCET bound reductions from 54% up to 73% can be achieved with an acceptable amount of overhead required for the optimization and WCET analyses themselves.
http://ls12-www.cs.uni-dortmund.de/

Influence of Procedure Cloning on WCET Prediction (Dortmund University, AbsInt)
For the worst-case execution time analysis loops are an inherent source of unpredictability and loss of precision. This is caused by the difficulty to obtain safe and tight bounds on the number of iterations executed by a loop in the worst case. In particular, data-dependent loops whose iteration counts depend on function parameters are extremely difficult to analyze precisely. Procedure Cloning helps by making such data-dependent loops explicit within the source code, thus making them accessible for high-precision WCET analyses.
In year 3 of Artist2, we studied the influence of standard optimizations found in ordinary compilers on the program's WCET. We present the effect of Procedure Cloning applied at the source-code level on worst-case execution time. The optimization generates specialized versions of functions being called with constant values as arguments. In standard literature, it is used to enable further optimizations like constant propagation within functions and to reduce calling overhead .
Our work shows that Procedure Cloning for WCET minimization leads to significant improvements. Reductions of the computed WCET bounds from 12% up to 95% were measured for real-life benchmarks. These results demonstrate that Procedure Cloning improves analyzability and predictability of real-time applications dramatically. In contrast, average-case performance as the criterion Procedure Cloning was developed for is reduced by only 3% at most. Our results also show that these WCET reductions only implied small overhead during WCET analysis.
http://ls12-www.cs.uni-dortmund.de/

2.3.2 *Individual Publications Resulting from these Achievements*

Saarland University

- Stephan Thesing: Modelling a system controller for timing analysis. EMSOFT 2006: 292-300, ACM Press
- Marc Schlickling, Markus Pister*:* *A Framework for Static Analysis of VHDL Code*, WCET Workshop, 2007
- Stephan Wilhelm, Björn Wachter: *Towards Symbolic State Traversal for Efficient WCET Analysis of Abstract Pipeline and Cache Models, WCET Workshop, 2007-08-28*
- Jan Reineke, Daniel Grund, Christoph Berg, and Reinhard Wilhelm. Timing Predictability of Cache Replacement Policies. *Real-Time Systems*, Springer 2007.
- Sebastian Altmeyer and Gernot Gebhard: *Optimal Task Placement to Improve Cache Performance, EMSOFT 2007, Salzburg.*

AbsInt:

- C. Ferdinand, R. Heckmann, and B. Franzen: Static Memory and Timing Analysis of Embedded Systems Code. In: Perry Groot (Ed.), Proceedings of VVSS2007 - 3rd European Symposium on Verification and Validation of Software Systems, 23rd of March 2007, Eindhoven, The Netherlands, TUE Computer Science Reports 07-04, pages 153-163, 2007.

- Christoph Cullmann, Florian Martin: Data-Flow Based Detection of Loop Bounds, in Proceedings of 7th International Workshop on Worst-Case Execution Time Analysis, Pisa, Italy, July 3, 2007.
- Ingmar Stein, Florian Martin: Analysis of path exclusion at the machine code level, in Proceedings of 7th International Workshop on Worst-Case Execution Time Analysis, Pisa, Italy, July 3, 2007.

Tidorum

- N. Holsti: "Analysing Switch-Case Tables by Partial Evaluation". In Proceedings of 7th International Workshop on Worst-Case Execution Time (WCET) Analysis, Pisa, Italy, July 3, 2007.
  http://www.irit.fr/wcet2007/finalpapers/holsti.pdf

Vienna University of Technology

- Raimund Kirner, *“SCCP/x - A Compilation Profile to Support Testing and Verification of Optimized Code”*, Proc. ACM Int. Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES'07), Sep/Oct. 2007, Salzburg, Austria.
- Raimund Kirner, Peter Puschner, “Time-Predictable Task Preemption for Real-Time Systems with Direct-Mapped Instruction Cache”, Proc. 10th IEEE International Symposium on Object-oriented Real-time distributed Computing May 2007, Santorini Island, Greece.
- Raimund Kirner, Jens Knoop, Adrian Prantl, Markus Schordan, and Ingomar Wenzel, *”WCET Analysis: The Annotation Language Challenge”*, Proc. 7th International Workshop on Worst-Case Execution Time Analysis, July 2005, Pisa, Italy.

Mälardalen University

- Jan Gustafsson, Andreas Ermedahl, Christer Sandberg, Björn Lisper: Automatic Derivation of Loop Bounds and Infeasible Paths for WCET Analysis using Abstract Execution. The 27th IEEE Real-Time Systems Symposium (RTSS 2006), Rio de Janeiro, Brazil.
- Jan Gustafsson, Andreas Ermedahl, Christer Sandberg, Björn Lisper: Automatic Derivation of Loop Bounds and Infeasible Paths for WCET Analysis using Abstract Execution. Real-Time in Sweden (RTiS) 2007, Västerås, Sweden.
- Jan Gustafsson: The WCET Tool Challenge 2006. 2nd International Symposium on Leveraging Applications of Formal Methods (ISOLA'06), p 248-249, Paphos, Cyprus, Editor(s): Tiziana Margaris, Anna Philippou, Bernhard Steffen.
- Jan Gustafsson: The WCET Tool Challenge. WCET workshop 2007, July 2007, Pisa, Italy.
- Jan Gustafsson, Andreas Ermedahl: Experiences from Applying WCET Analysis in Industrial Settings. The 10th IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC2007), Santorini Island, Greece.
- Jan Gustafsson, Andreas Ermedahl: Experiences from Applying WCET Analysis in Industrial Settings. Real-Time in Sweden (RTiS) 2007, Västerås, Sweden.

Dortmund University

- Paul Lokuciejewski, Heiko Falk, Martin Schwarzer and Peter Marwedel. *Tighter WCET Estimates by Procedure Cloning*. In *Proceedings of “The 7th International Workshop on Worst-Case Execution Time Analysis” (WCET)*, Pisa, Italy, July 2007.

- Heiko Falk and Peter Marwedel (Editors). *Proceedings of the 10th International Workshop on Software & Compilers for Embedded Systems (SCOPES)*, Nice, France, April 2007.

2.3.3 Interaction and Building Excellence between Partners

Manifold interactions have taken place between the partners.

Interaction between Tidorum and Mälardalen:

- The ALF (ARTIST2 Language for Flow Analysis) is a language intended to be used for flow analysis in conjunction with WCET (Worst Case Execution Time) analysis. ALF is designed to be possible to generate from a rich set of sources: linked binaries, source code, compiler intermediate formats, and possibly more. The specification has been developed by Andreas Ermedahl, Jan Gustafsson, and Björn Lisper and has been reviewed by Niklas Holsti at Tidorum, Finland. On-going work.
- Industrial case studies of WCET analysis. Tidorum supported Mr Elie Assaf's WCET analysis of an automotive application implemented in C++ and automatically generated from an UML model with the Rhapsody tool. The analysis used Tidorum's Bound-T tool for the ATMEL AVR processor.
- WCET analysis in teaching. Mälardalen uses Tidorum's Bound-T tool in laboratory exercises in the real-time programming course.

Interaction between Tidorum and York:

- Integration between Bound-T and the commercial variant of York measurement-based tool, RapiTime. The tools were used together in the PEAL project, funded by the European Space Agency. The main PEAL project finished in February 2007. A small extension will be executed in the fall of 2007.

Interaction between AbsInt and Mälardalen:

- The WCET tools SWEET and aiT has been used towards the same code at Volvo CE. First the automatic flow analysis of SWEET was used to produced flow facts; as a next step, aiT was run using the results from SWEET.
- There is ongoing work to convert the AbsInt AIR format to SWEET format. As a first result, this will allow SWEET to analyse binaries for the NECV850 processor using the aiT binary reader.

2.3.4 Joint Publications Resulting from these Achievements

- Reinhard Wilhelm, Jakob Engblom, Andreas Ermedahl, Niklas Holsti, Stephan Thesing, David Whalley, Guillem Bernat, Christian Ferdinand, Reinhold Heckmann, Frank Mueller, Isabelle Puaut, Peter Puschner, Jan Staschulat, , and Per Stenström. The Determination of Worst-Case Execution Times-Overview of the Methods and Survey of Tools. Accepted for ACM Transactions on Embedded Computing Systems (TECS), 2007.
- Christian Ferdinand (AbsInt), Reinhold Heckmann (AbsInt), Hans-Joerg Wolff, Christian Renz, Oleg Parshin (USaar), and Reinhard Wilhelm (USaar). Towards Model-Driven Development of Hard Real-Time Systems - Integrating ASCET-MD and aiT/StackAnalyzer. In *Proceedings of Automotive Software Workshop in San Diego*, 2006

- Christian Ferdinand (AbsInt), Reinhold Heckmann (AbsInt), Hans-Joerg Wolff, Christian Renz, Manabendra Gupta, Oleg Parshin (USaar): Towards an Integration of Low-Level Timing Analysis and Model-Based Code Generation. In: Tiziana Margaria, Anna Philippou, Bernhard Steffen (Eds.) IEEE-ISoLA 2006 - 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, 15-19 November 2006, Paphos, Cyprus, pages 235-241.
- C. Ferdinand (AbsInt), F. Martin (AbsInt), C. Cullmann (AbsInt), M. Schlickling (USaar), I. Stein (AbsInt), S. Thesing (USaar), R. Heckmann (AbsInt): New Developments in WCET Analysis. In: Thomas Reps, Mooly Sagiv, Jörg Bauer (Eds.), Program Analysis and Compilation, Theory and Practice. Lecture Notes in Computer Science 4444, pages 12-52, Springer-Verlag, 2007.
- Daniel Sehlberg, Andreas Ermedahl, Jan Gustafsson, Björn Lisper, Steffen Wiegratz (AbsInt Angewandte Informatik GmbH, Saarbrucken, Germany): Static WCET Analysis of Real-Time Task-Oriented Code in Vehicle Control Systems. 2nd International Symposium on Leveraging Applications of Formal Methods (ISOLA'06), Paphos, Cyprus.
- The Worst-Case Execution Time Problem - Overview of Methods and Survey of Tools, Reinhard Wilhelm, Jakob Engblom (Virtutech, Sweden), Andreas Ermedahl, Niklas Holsti (Tidorum LTD), Stephan Thesing, David Whalley, Guillem Bernat, Christian Ferdinand, Reinhold Heckmann, Tulika Mitra, Frank Muller, Isabelle Puaut, Peter Puschner, Jan Staschulat, Per Stenström, MRTC report ISSN 1404-3041 ISRN MDH-MRTC-209/2007-1-SE, Mälardalen Real-Time Research Centre, Mälardalen University, March, 2007
- Heiko Falk, Sascha Plazar and Henrik Theiling. Compile-Time Decided Instruction Cache Locking Using Worst-Case Execution Paths. In Proceedings of “The International Conference on Hardware/Software Codesign and System Synthesis” (CODES+ISSS), Salzburg, Austria, October 2007.
- Paul Lokuciejewski, Heiko Falk, Martin Schwarzer, Peter Marwedel and Henrik Theiling. Influence of Procedure Cloning on WCET Prediction. In Proceedings of “The International Conference on Hardware/Software Codesign and System Synthesis” (CODES+ISSS), Salzburg, Austria, October 2007.

2.3.5 Keynotes, Workshops, Tutorials

Cluster partners have been very active in the dissemination of results.

Workshop: Reinhard Wilhelm: Timing Predictability - A Must for Avionics Systems

Conference name National Workshop on Aviation Software Systems: Design for Certifiably Dependable Systems, A Workshop on Research Directions and State of Practice of High Confidence Software Systems, October 4-5, 2006, Alexandria, VA, USA.

This workshop was sponsored to bring together the Practice Community with the Research Community in avionics to define the Intellectual Agenda in Software for Critical Aviation Systems. The goals, among others, include:

- Define Current State of the Art
- Identify Key Issues and Needs
- Identify Promising Research Approaches
- Define Educational Needs and Approaches

http://chess.eecs.berkeley.edu/hcssas/

Tutorial: Reinhard Wilhelm: Timing Analysis
MOTIVES ARTIST2 Winter School, Trento, Italy, February 19-23 2007, organized by Kim Guldstrand Larsen, Bengt Jonsson, Reinhard Wilhelm.

This 5-day winter school was for young researchers working or wanting to work in modelling, validation, synthesis and performance analysis of embedded systems, as well as engineers from industry with a practical background in design, control and testing of embedded systems.
http://www.artist-embedded.org/artist/Overview,577.html

Workshop: Reinhard Wilhelm: Design for Timing Predictability

Conference name: Dagstuhl Seminar on Quantitative Aspects of Embedded Systems, Schloss Dagstuhl, 04.03.07 - 09.03.07

Organizers: Boudewijn Haverkort (University of Twente, NL), Joost-Pieter Katoen (RWTH Aachen, D), Lothar Thiele (ETH Zürich, CH)

Despite the importance of the quantitative constraints for the well-operation of embedded systems, the proper assessment of cost, resources, performance, dependability, robustness, etc., often comes as an afterthought. It is rather common for embedded software to be fully designed and functionally tested before any attempt is undertaken to determine its performance, dependability or resource-usage characteristics. One of the main reasons for this situation is that well-developed and rigorous evaluation techniques for non-functional, i.e., quantitative system aspects have not become an integral part of standard software engineering practice. This undesirable situation has led to the increased interest by embedded software researchers to extend the usual functional specification and properties with a set of “performance indices", e.g., stated in terms of costs, timeliness, speed and the like, and constraints on these indices. Also in industry, a growing interest in assessing non-functional aspects of embedded systems as early as possible in the system design life cycle can be witnessed.
http://www.dagstuhl.de/en/program/calendar/semhp/?semnr=2007101

Workshop: Software & Compilers for Embedded Systems (SCOPES) 2007
Nice, France – April 20, 2007

The influence of embedded systems is constantly growing. Increasingly powerful and versatile devices are developed and put on the market at a fast pace. The number of features is increasing, and so are the constraints on the systems concerning size, performance, energy dissipation and timing predictability. Since most systems today use a processor to execute an application program rather than using dedicated hardware, the requirements can not be fulfilled by hardware architects alone: Hardware and software have to work together to meet the tight constraints put on modern devices.

One of the key characteristics of embedded software is that it heavily depends on the underlying hardware. The reason of the dependency is that embedded software needs to be designed in an application specific way. To reduce the system design cost, e.g. code size, energy consumption etc., embedded software needs to be optimized exploiting the characteristics of the underlying hardware.

SCOPES focuses on the software generation process for modern embedded systems. Topics of interest include all aspects of the compilation process, starting with suitable modeling and specification techniques and programming languages for embedded systems. The emphasis of the workshop lies on code generation techniques for embedded processors. The exploitation of specialized instruction set characteristics is as important as the development of new optimizations for embedded application domains. Cost criteria for the entire code generation and optimization process include runtime, timing predictability, energy dissipation, code size and others. Since today's embedded devices frequently consist of a multi-processor system-on-chip, the scope of this workshop is not limited to single-processor systems but particularly covers compilation techniques for MPSoC architectures.

In addition, this workshop intends to put a spotlight on the interactions between compilers and other components in the embedded system design process. This includes compiler support for e.g. architecture exploration during HW/SW codesign or interactions between operating systems and compilation techniques. Finally, techniques for compiler aided profiling, measurement, debugging and validation of embedded software are also covered by this workshop, because stability of embedded software is mandatory.

SCOPES 2007 is the 10th workshop in a series of workshops initially called "International Workshop on Code Generation for Embedded Processors". The name SCOPES has been used since the 4th workshop. The scope of the workshop remains software for embedded systems with emphasis on code generation (compilers) for embedded processors.

SCOPES 2007 was organized by Heiko Falk and Peter Marwedel from Dortmund University and was held as DATE Friday Workshop.

http://www.scopesconf.org/scopes-07/

3. Future Work and Evolution

3.1 Problem to be Tackled over the next 12 months (Sept 2007 – Aug 2008)

The activity will continue its work towards improving the state of the art both in strengthening the tools individually and in combination and in identifying means to take constructive influence on system design to improve the analyzability of systems

Integration of components

Continue integration of tool components.. Finalise the definition of the ALF language as an input language for the Mälardalen flow analysis, thus as a possible vehicle for integrating this analysis with other tools.

Define an embedding of ALF in AIR, to form the "computation semantics" representation in AIR. Generate ALF so that the Mälardalen flow analysis can be used.

Flow description format

Continue research on flow description languages and formats. Define the flow description challenge and set up a call for this challenge. (The results of the challenge will later be used to define a common set of flow information that can be included in languages for WCET description, e.g. AIR).

WCET Tool Challenge

The WCET Tool Challenge will be repeated 2008 with improved procedures and benchmarks.

Timing Predictability

Design for Timing-Predictability will be continued in an ICT project, Predator, in a consortium mostly consisting of ARTIST2 partners, together with the two technology consumers Airbus and Bosch.

Measurement-based Timing Analysis

Continue work on the framework for measurement-based analysis , and conduct experiments with measurement-based analysis . The emphasis will be on transferring the current monolithic framework into a modular tool platform in which components can be exchanged and added more easily.

WCET-Aware Compilation

Further studies on the influence of standard compiler optimizations on the program’s WCET will be continued and the development of novel WCET-aware compiler optimizations will be performed by Dortmund University. On the one hand, this will include optimizations exclusively focussing on WCET as objective function, like e.g. exploitation of memory hierarchies for WCET minimization. On the other hand, the mechanisms provided by Dortmund’s WCET-aware compiler developed during ARTIST Year2 for multi-objective optimization (e.g. trading off WCET vs. code size) will be used. It is intended to set up a cooperation with ARTIST2 core partners of the timing analysis platform (Mälardalen) in order to integrate flow analysis techniques into Dortmund’s compiler.

3.2 Current and Future Milestones

- Year2: Standard tool architecture and interfaces

  - The tool architecture has been further clarified. In the course of writing the joint survey paper, modularity of the architecture has been further extended (*100% completed*).
  - An agreement on the ARTIST2 common interface language, AIR, has been reached. The syntax has been defined, the semantics is accepted, but has not been formally written down (*60% completed by writing up the syntax for the agreed semantics*).
  - The extension of AIR for the computation semantics has been conceived. Basic operations have, yet, to be defined (*50% completed by specifying the approach to the specification of computation semantics*)
  - The chosen interface language, AIR, is being extended by Saarland University and by AbsInt to suit the needs of other partners (syntax 95%, Semantics 50%)
  - Four partners of the team (Vienna, Mälardalen, Tidorum, AbsInt) will continue to work on path description attributes for AIR to arrive at a uniform notation (20% completed)

- Year3: Initial integration of existing components
  - **The chosen interface language, AIR, is being extended by Saarland University and by AbsInt to suit the needs of other partners *(90% completed)*.**
  - **Four partners of the team (Vienna, Mälardalen, Tidorum, AbsInt) will continue to work on path description attributes for AIR to arrive at a uniform notation.**
  - Mälardalen will wrap up its flow analysis into a component with well-defined interfaces, which will be integrated with the aiT tool of AbsInt and the Bound-T tool of Tidorum *(achieved 30% - ALF preliminary definition)*
- Year4: Version 2 Integration of existing components
  - Achieved 10% (AIR definition status)
  - the definition and call for the flow description language challenge shall be completed

3.3 Indicators for Integration

In the first two years of the project considerable integration work has taken place:

- The partners have agreed on standard tool architecture and a set of textual interfaces.
- Experiments with interfacing through these textual interfaces have been successfully performed.
- Several professional components of aiT and Bound-T are being offered by the industrial partners to be used by the academic partners.
- AbsInt’s aiT tool and Tidorum's Bound-T tool have been used in industrial case studies by the Mälardalen team (see publications).
- The Bound-T version for Renesas H8/300 is used in real-time education at Mälardalen.
- Tidorum and York (through Rapita Systems Ltd) have completed a project for the European Space Agency, to adapt their timing analysis tools to the LEON2 (SPARC V8) processor and to evaluate the tools on real on-board satellite software. This includes integration between the tools: Tidorum's Bound-T tool can now be used as a

front-end for York's measurement-based analysis as implemented in the RapiTime tool from Rapita Systems.

- TU Vienna and University of York worked on issues of how systematic measurements can be used for WCET analysis.
- aiT and the compiler from Dortmund University have been tightly integrated. This combination of tools is used daily at Dortmund University. It provides the basis for numerous optimizations enabled by this integration. AbsInt is able to access the combined tools as well.

Indicators for integration in the future:

- Working integration of Mälardalen’s flow analysis with the aiT tool of AbsInt and the Bound-T tool of Tidorum.
- Development and documentation of path description attributes for AIR.
- Successful embedding of ALF in AIR. Successful use of ALF/AIR to integrate the Mälardalen flow analysis with some (other) WCET tool.
- Joint publications about Timing Analysis.

The following table shows an overview of the cooperation on the timing analysis platform.

	USaar	**AbsInt**	**York**	**MdU**	**TUW**	**Tidorum**	**IRISA**
Definition of common interface format AIR	+	+	+	+	+	+	+
Extension for Instruction Set Semantics	+	+		+		+	
Extension for Path Annotation		+	+	+	+	+	
CRL2 usage experiments	+	+				+	+
Industrial case studies with aiT		+		+			
Industrial case studies with Bound-T				+		+	
Bound-T for the H8/300 and Real-time systems teaching				+		+	
Measurement based timing analysis			+		+		
Bound-T as a front-end for RapiTime			+			+	
Integrating the MdU’s flow analysis into aiT		+		+			
Integration the MdU’s flow analysis into Bound-T				+		+	
Benchmark programs for timing analysis tools	+		+	+			
Parametric WCET analysis	+			+			
WCET-oriented IDE				+	+		

3.4 Main Funding

Main sources of funding are:

- TU Vienna's work is funded mainly from the DECOS Integrated Project and the MoDECS project (funded by the Austrian FIT-IT programme; duration: Sept. 2003-Aug. 2005), and the Te-DES Project (funded by FIT-IT; duration: Apr. 2005-Mar. 2007) and the COSTA project (funded by the Austrian science funding agency FWF; duration: July 2006-June 2009, and the FORTAS project (funded by the Austrian science funding agency FWF; duration: Jan. 2007-Dec. 2009).
- Saarland University's funding comes from the national project AVACS (DFG).
- Mälardalen has funding from the KK-foundation (grant 2005/0271, duration Jan 2006 – Dec. 2008), and the Foundation for Strategic Research (PROGRESS Strategic Research Centre, duration Jan. 2006 – Dec. 2010).
- Work at York has been funded by BaE systems and by NextTTA EU project IST-2001-32111.
- Tidorum has essentially no external funding for this work, other than the ARTIST2 contribution
- AbsInt currently receives funding from the following projects:
  - Integrated project "Verisoft – Beweisen als Ingenieurwissenschaft" (funded by the German ministry of research (BMBF)) on the formal verification of software and hardware with applications from telecommunication, security and automotive areas.
  - Project "SuReal – Sicherheitsgarantien unter Realzeitanforderungen" (funded by the German ministry of research (BMBF)) on the creation of a pervasive development process in which safety aspects are continuously taken into account and can be formally verified at any time.
  - EU Framework VI Specific Targeted Research Project IST-2004-510255 "EmBounded" whose aims are to identify, to quantify and to certify resource-bounded code in a domain-specific high-level programming language for real-time embedded systems.
  - EU Framework VI Specific Targeted Research Project IST-033661 "INTEREST" aiming at overcoming the current lack of integration and interoperability of tools for developing Embedded Systems software.
  - An ITEA project, ES_PASS started recently. Its goal is to establish a product-based assurance for embedded systems.

4. Internal Reviewers for this Deliverable

Raimund Kirner, (TU Vienna)
Jan Reineke (Saarland University)