IST-004527 ARTIST2
Network of Excellence
on Embedded Systems Design

Activity Progress Report for Year 4

JPIA-Platform

Timing Analysis Platform

Clusters:

Compilers and Timing Analysis

Activity Leader:

Prof. Björn Lisper (Mälardalen University)
http://www.idt.mdh.se/~blr/

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. Björn Lisper – Mälardalen university (Sweden)
Activity Leader, Timing-AnalysisTtools
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. Mälardalen University is coordinating the integration activity.

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

Dr. Christian Ferdinand – AbsInt GmbH (Germany)
Tool Supplier
AbsInt provides advanced WCET analysis tools for a wide variety of targets. The work within ARTIST2 focuses on the advance of WCET analysis techniques by providing and defining interchange formats for the components of WCET tools like AIR (ARTIST2 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.

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 – TU Dortmund (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, definition of Annotation Language for WCET Analysis, Measurement-Based 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

Prof. 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 both follow two 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 solved for tasks executed on monoprocessors without preemption. 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

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/forschung/publikationen/symbolische_laufzeitanalyse_fuer_prozesse_symtap/

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

7 http://www.wcet.at/

can be simplified by automated generation from formal descriptions of the processors and systems to be analysed. The different tools offer different strengths in the areas of different subproblems. The goal is to combine the respective strengths.
The transition of industry to multi-processor and multi-core targets opens up a whole domain of new research problems. Not all envisioned multi-core architectures will allow timing analysis. The timing-analysis community needs to take an influence on the architecture design to make timing analysis feasible. Good compromises have to be found between the goals of good average-case performance, good predictability of the worst case and the necessary effort for the analyses.

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 Problems Tackled in Year 4

The flow analysis format ALF has been finalized. Work has begun to convert the flow analysis of Mälardalen's WCET analysis tool SWEET into analyzing ALF. This work will continue after the end of ARTIST2.

A context-sensitive program flow constraint format "PFF" is being defined. This format will be used for the results of the flow analysis of SWEET, which then can be exported to other tools. The format will replace the current "Flow Facts" format, which has some limitations. Through ALF and PFF, other tools will be able to use the flow analysis of SWEET as a plugin. Also this work will continue after the end of ARTIST2.

The second event in the WCET Challenge series was organized in 2008. Compared to the first Challenge in 2006, the number of participants grew from five to six, four of which are new. One participant, the WCET-aware compiler WCC from TU Dortmund, demonstrates integration of tools by connecting a compiler to a WCET tool (aiT from AbsInt). WCC uses the CRL2 language as the connection channel. Several Challenge participants now supported the same target processor, which for the 2008 Challenge is the ARM7 as implemented in the LPC2138 chip from NXP. The largest benchmark, a real program from the space domain, was now provided with a test suite for the use of measurement-based tools. The analysis tasks now include both WCET questions and control-flow analysis questions which allows the comparison of flow-analysis results even for different target processors. The benchmarks and the analysis tasks are defined in a shared Wiki site which also presents the results. The participants are still refining and extending their results.

Work on a WCET-aware compiler continued. This compiler uses the AIR/CRL2 interface format and aiT technology to determine WCET estimates for various code possibilities and then to select the most efficient of these.

The research on defining common flow attributes was continued by proposing a list of essential ingredients for a WCET annotation language. As one of the new concepts the distinction between timing invariants (i.e., information that can be derived from the considered system layers) and timing fictions (i.e., information that cannot be derived form the considered system layers) was introduced. The timing fictions allow the user to ask “what if” questions to the WCET analysis tool. But since timing fictions do not necessarily used to describe any worst-case behaviour, they have to be labelled as timing fictions. The concept of system layers was also introduced because not all annotations for WCET analysis describe only the control flow of the program. For example, it might be also be required to describe a loop bound that depends on the writing time of a flash memory. We categorized what we consider the essential ingredients of a WCET annotation language into five categories: C1 Annotation Categorization, C2 Program-Specific Annotations, C3 Addressable Units, C4 Control Flow Information, and C5 Hardware-Specific Annotations.

1.6 Comments From Year 3 Review

1.6.1 Reviewers’ Comments

“This is a quality document. No specific remarks.”

1.6.2 How These Have Been Addressed

We have applied the same quality assurance procedures as last year.

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 ARTIST2 Intermediate Representation. AIR is based on CRL2. CRL2 is a dialect (superset) of AIR as explained in Section 2.2. 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 “computation 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 computation 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 computation 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 Intermediate') 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 Previous Work in Year 3

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 they 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. The WCET Tool Challenge was also presented at the WCET workshop 2007.
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 a 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 ECRTS 2008 paper; see 2.4.2) 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). 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/

2.4 *Final Results*

2.4.1 *Technical Achievements*

ARTIST Interchange Representation and Attribute Database: AIR (AbsInt)

Work on the AIR format continued. The format was extended and adapted to the needs of the partners. The attribute database was extended by new attributes as required.

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 and 4. It was decided that the ALF language from Mälardalen University meets the requirements, and should be used for this purpose. Mälardalen produced a specification of ALF, which finalizes the preliminary specification produced in Year 3.
http://www.mrtc.mdh.se/index.php?choice=publications&id=1351

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

This work will allow SWEET to analyse binaries for the NECV850 processor using AbsInt tools like aiT or "exec2crl". The AIR format for NECV850 is translated into SWEET's internal code representation for low-level representation. The translator is close to final. AbsInt's existing StackAnalyzer for NECV850 was extended by a timing analysis component.

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

Common Flow Description Attributes (TU-Vienna/Real-Time Systems Group, TU-Vienna/Programming Languages Group, Mälardalen University, AbsInt, University of York)
Defining common flow description attributes has shown to include more aspects than initially expected. Besides the flow description it is also necessary to describe other aspects like the memory layout, absolute timings, etc. We decided to postpone the final proposal of common flow description attributes to get more feedback from the community. To get more feedback, we proposed the WCET Annotation Language Challenge at the 7th International Workshop on Worst-Case Execution Time Analysis (WCET 2007) in Pisa, Italy. The website of the WCET Annotation Language Challenge has been set up at the address

http://costa.tuwien.ac.at/languages.html.

Meanwhile we published a paper that presents what we consider essential ingredients. This paper is intended to trigger concrete feedback. It is planned to post feedback from other people also on this web page to reflect the current status.

WCET Challenge 2008 (Tidorum, Mälardalen, USaar)

The structure of the Challenge, as a set of benchmark programs, analysis tasks, and results, was developed from the loose structure of the 2006 Challenge to a more detailed and precise form, shown on the Challenge Wiki site at http://www.mrtc.mdh.se/projects/WCC08/doku.php.

It was agreed that the Challenge should be considered a continuous process, allowing the addition of benchmark programs, analysis tasks, participants, and results at any time, but punctuated by an annual deadline at which a snapshot is taken and becomes the outcome of the Challenge for that year.

Most participants in the 2008 Challenge found a need for a standard, formal language to describe the execution flows assumed in the analysis tasks. This creates an interesting connection to the Annotation Language Challenge of the CoSTA project (http://costa.tuwien.ac.at/languages.html).

In addition to the ARTIST2 partners listed in the heading above, who were active in organizing the 2008 Challenge, several other ARTIST2 partners participated in the Challenge with their tools: TU Dortmund University, TU Vienna Real-Time Systems Group, TU Vienna Programming Languages Group, and University of York (through Rapita Systems).

Timing Analysis and Timing Predictability (USaar and AbsInt)
The notion of predictability of cache architectures has been clarified. This is the first precise notion of predictability found in the literature. It turns out that the cache-replacement strategy is the decisive characteristic for the predictability of architecture. 4 different replacement strategies were compared, and the LRU strategy was found to be optimal.

The PREDATOR project in the 7th Framework Programme attempts to reconcile performance and predictability. It has identified the PROMPT (PRedictability Of Multi-processor Timing) design rules for predictable multi-processor design. The first principles are to avoid interference on shared resources in the architecture and to allow the application designer the mapping of applications to target architecture without the introduction of new interferences that were not present in the application.

Parametric Timing Analysis (USaar and Mälardalen)
Timing analyses require that information such as bounds on the maximum numbers of loop iterations are known statically, i.e., during design time. Parametric timing analysis softens these requirements: it yields symbolic formulas instead of single numeric values representing the upper bound on the task's execution time. So, some input parameters to the program can remain unknown until the final use of the task. The developed analysis determines the parameters of the program, constructs parametric loop bounds, takes processor behaviour into account and attains a formula automatically.

Synergy between Code Synthesis and Timing Analysis (USaar and AbsInt)
One of the problems to be solved synergetically by code synthesis compiling, and timing-analysis is to support mode-specific timing analyses. Many embedded control systems have several operating modes with different timing requirements. Some operating modes are not explicitly specified on the model level or in comments on the C level. Saarland University in cooperation with Bosch (within the PREDATOR project), attempts to semi-automatically identify such operating modes. This would allow timing analysis to implement mode-specific execution-time bounds which can be used to improve schedulability of task sets.

WCET Analysis for Systems with Preemptive Scheduling (USaar and Absint)
Derivation of timing guarantees was extended in order to cope with the systems with preemptive scheduling. A new method to compute valid upper bounds on a task's worst case execution time (WCET) under preemption was proposed. This method approximates an optimal memory layout such that the set of possibly evicted cache-entries during preemption is minimized. This set then delivers information to bound the execution time of tasks under preemption in an adopted WCET analysis.

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

Flow information is used to guide the WCET analysis framework in finding the worst-case path. Such flow information may be calculated automatically or given as manual code annotations. Also the automatic calculation of flow information is easier at higher abstraction levels than the machine code. Thus, in both cases, the automatic calculation or the manual annotation a mechanism is required to transform the flow information down to the object code level where the WCET analysis is done. The initial work done by Kirner in his PhD thesis has been extended to the transformation of flow information at source-code level and interprocedural program optimization. The transformation of the flow information has been integrated into the interprecedural program transformation framework SATiRE, which connects other frameworks like ROSE or PAG. Adding the support of transforming flow information at the source-code-level has the advantage that most transformations that change the structure of the code can be already done in a rather compiler-independent way at the source code. Thus, on a compiler that does not support the transformation of flow information, one can deactivate all code optimisations In the compiler that change the structure of the control flow and use instead code transformations at the source-code level. Further information can be found at the web site of the CoSTA project: http://costa.tuwien.ac.at/languages.html.

WCET- and Memory Architecture-aware Compilation (TU Dortmund, AbsInt)
See the activity report for the Compilers Platform Activity.

WCET-aware Procedure Positioning and Cloning (TU Dortmund, AbsInt)
See the activity report for the Compilers Platform Activity.
Using Learning to Support the Development of Embedded Systems (York)
In 2007, the University of York was awarded 3.5 years of funding by EPSRC to investigate how novel techniques could be used as part of WCET analysis. The first part of the funding is for a post-doc to investigate how machine learning can be used to infer models of software and hardware as part of WCET analysis. To date two publications have been produced as part of this grant. The first paper (Bartlett 2008) builds on previous work that considers how Inductive Logic Programming can be used to establish relationships in loop guards to reduce the pessimism in program flow analysis. In particular the latest paper look at how the scalability of the work can be improved. The second paper, (Bate 2008), looks at how models of branch predictors can be determined by assessing program traces and then how this can be combined with static analysis. The second part of the funding is for a PhD student who will investigate which coverage metrics are most appropriate for WCET analysis and then how automated techniques can be used to generate the test cases needed.

Handling Timing Anomalies for Efficient WCET Analysis (USaar)
Abstractions employed for static timing analysis can lead to non-determinism that may require the analyzer to evaluate an exponential number of choices even for straight-line code. Pruning the search space is difficult because of the danger of "Timing Anomalies" where local worst-case choices may not lead to the global worst-case scenario. Earlier work explored the characterization and identification of such anomalies. Starting with the assumption that all reasonable abstractions of modern hardware do exhibit timing anomalies, we explored on the theoretical level, when and how the search space can be safely pruned to enable more efficient, yet safe, WCET analyses. To this end, we propose an approach that uses precomputed information to safely discard states in almost constant time.
The Impact of Timing Anomalies on WCET Analysis (TU Vienna)
There are two different phenomena that are called timing anomaly: the inversion of a local effect at the overall execution time and the amplification of a local effect at the overall execution time. We explored different the effects of timing anomalies at a theoretical level and analysed whether a stepwise timing analysis of hardware effects is still possible. We published first results where we described two different timing composition methods. The interesting result is that each timing composition method is safe if only one type of timing anomaly is present. However, in the case that both types of timing anomalies are present both techniques are not safe. These results provide more insights to the nature of timing anomalies and how they are challenging to WCET analysis.

2.4.2 Individual Publications Resulting from these Achievements

Saarland University

- Jan Reineke, Daniel Grund, Christoph Berg, and Reinhard Wilhelm. Timing Predictability of Cache Replacement Policies. *Real-Time Systems*, 37(2):99-122, November 2007.
- Jan Reineke and Daniel Grund. Relative Competitive Analysis of Cache Replacement Policies. In *LCTES '08: Proceedings of the 2008 ACM SIGPLAN-SIGBED conference on Languages, compilers, and tools for embedded systems*, pages 51-60, New York, NY, USA, June 2008. ACM.

AbsInt

- Christian Ferdinand and Reinhold Heckmann: Static Memory and Execution Time Analysis of Embedded Code. *Transactions Journal of Passenger Cars - Electronic and Electrical Systems* 9, 2007.
- Philippe Baufreton, Reinhold Heckmann: Reliable and Precise WCET and Stack Size Determination for a Real-life Embedded Application. *Workshop on Leveraging Applications of Formal Methods, Validation and Verification*, 12-14 December 2007, ENSMA, Poitiers-Futuroscope. France.
- C. Ferdinand, R. Heckmann, M. Jersak, F. Martin, K. Richter: Integrating System-Level and Code-Level Timing Analysis for Dependable System Development. *4th European Congress ERTS - Embedded Real Time Software*, January 29, 30, 31, February 1, 2008, Toulouse, France.
- C. Ferdinand, R. Heckmann, T. Le Sergent, D. Lopes, B. Martin, X. Fornari, F. Martin: Combining a High-Level Design Tool for Safety-Critical Systems with a Tool for WCET Analysis on Executables. *4th European Congress ERTS - Embedded Real Time Software*, January 29, 30, 31, February 1, 2008, Toulouse, France.

- Christian Ferdinand, Reinhold Heckmann, Bärbel Franzen: Static Program Analysis with Tools Based on Abstract Interpretation. *ECE Information & Know-how for Embedded Engineers* 02/08.
- Kästner D.: Validierung des Zeitverhaltens von Echtzeitsystemen. *Fourth SafeTRANS Industrial Day*, Fraunhofer-Forum, Berlin, April 30, 2008.
- Christian Ferdinand and Reinhold Heckmann: Worst-Case Execution Time - A Tool Provider's Perspective*. 11th IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing ISORC 2008*, Orlando, Florida, USA, 5-7 May 2008.

Tidorum

- Niklas Holsti. Computing Time as a Program Variable: A Way Around Infeasible Paths,, *Proc. 8th International Workshop on Worst-Case Execution Time Analysis (WCET'08)*, Prague, Czech Republic, July, 2008

Vienna University of Technology

- Raimund Kirner and Peter Puschner. *Obstacles in Worst-Case Execution Time Analysis. Proc. 11th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing*, p. 333-339, 2008.
- Raimund Kirner, Albrecht Kadlec, Peter Puschner, Adrian Prantl, Markus Schordan and Jens Knoop. Towards a Common WCET Annotation Language: Essential Ingredients*.* In *Proc. 8th Euromicro Workshop on WCET Analysis*, 2008.
- Raimund Kirner. *Compiler Support for Timing Analysis of Optimized Code: Precise Timing Analysis of Machine Code with Convenient Annotation of Source Code*. VDM Verlag, 2008.

Mälardalen University

- Dani Barkah, Andreas Ermedahl, Jan Gustafsson, Björn Lisper and Christer Sandberg. Evaluation of Automatic Flow Analysis for WCET Calculation on Industrial Real-Time System Code, *Proc. 20th Euromicro Conference of Real-Time Systems (ECRTS'08),* Prague, Czech Republic, July, 2008
- Jan Gustafsson and Andreas Ermedahl. Merging Techniques for Faster Derivation of WCET Flow Information using Abstract Execution, *Proc. 8th International Workshop on Worst-Case Execution Time Analysis (WCET'08),* Prague, Czech Republic, July, 2008

University of York

- M. Bartlett and I. Bate and D. Kazakov. Challenges in Relational Learning for Real-Time Systems Applications, *Proc. 18th International Conference on Inductive Logic Programming*, Sep. 2008.
- Iain Bate and Dimitar Kazakov. New Directions in Worst-Case Execution Time Analysis, *Proc. IEEE Congress on Evolutionary Computation (IEEE CEC 2008)*, 2008

2.4.3 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, continued from Year 3.

- Work has started to make Bound-T export an ALF program representation, with the aim to let Bound-T use the abstract-execution analysis from SWEET as a component.
- WCET analysis in teaching. Mälardalen uses Tidorum's Bound-T tool in laboratory exercises in the real-time programming course.

Interaction between Saarland and Mälardalen

- The joint work on parametric WCET analysis has continued through the writing of a joint publication (RTCSA 08), see the list of joint publications below.

Interaction between AbsInt, TU Vienna, York, and Mälardalen

- AbsInt, TU Vienna, Mälardalen, and York spinoff Rapita Systems are partners in the FP7 STREP ALL-TIMES (Integrating European Timing Analysis Technology). The partners have worked together on the definition of open interfaces between timing analysis tools, for subsequent integration.

Interaction between Tidorum, Mälardalen, and York

- These three partners organized the WCET Tool Challenge 2008 and jointly prepared the Challenge Wiki site, which is hosted at Mälardalen. A new Wiki for the next Challenge(s) will also be hosted at Mälardalen.

Interaction between Tidorum and York

- Tidorum (and partially York through Rapita Systems) is engaged in a project for the European Space Agency to study the timing and verification aspects of cache memories in space systems. The main PEAL project ended in February 2007. An extension focusing on cache-aware code layout was started in the fall of 2007 and continues to 2008. The project uses tools from Rapita Systems with Tidorum's Bound-T as a front-end. The main partner from the aerospace domain is Thales Alenia Space, France, and the fourth partner is the University of Padua

Interaction between AbsInt and Dortmund

- AbsInt is supporting Dortmund in its effort to build an WCET-aware compiler by integrating aiT technology into the compiler framework.

Interaction between AbsInt and Saarland

- AbsInt and Saarland are cooperating on the integration of code synthesis, compilers, and timing analysis, on timing predictability, on scheduling, and on analysis of multi-core architectures.

Interaction between TU Vienna and Dortmund

- TU Vienna and Dortmund are cooperating on the transformation of flow facts during code transformation. The TU Vienna had integrated the flow-facts transformation into a source-to-source program transformation framework. Dortmund had integrated the flow-facts transformation into their research compiler.

The joint publications in Year 4, related to this activity.

- Sebastian Altmeyer, Christian Hümbert, Björn Lisper, and Reinhard Wilhelm: Parametric timing analysis for complex architectures. *Proc. 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 08)*, Kaohsiung, Taiwan, Aug. 2008

- Sebastian Altmeyer and Gernot Gebhard. WCET Analysis for Preemptive Systems. In Raimund Kirner, editor, *Proc. 8th International Workshop on Worst-Case Execution Time (WCET) Analysis*, pages 105-112, Prague, Czech Republic, July 2008. OCG.
- Heiko Falk, Sascha Plazar, and Henrik Theiling: Compile-Time Decided Instruction Cache Locking Using Worst-Case Execution Paths. *CODES+ISSS*, Salzburg, Austria, 2007.
- Niklas Holsti, Jan Gustafsson, Guillem Bernat (eds.), Clément Ballabriga, Armelle Bonenfant, Roman Bourgade, Hugues Cassé, Daniel Cordes, Albrecht Kadlec, Raimund Kirner, Jens Knoop, Paul Lokuciejewski, Nicholas Merriam, Marianne de Michiel, Adrian Prantl, Bernhard Rieder, Christine Rochange, Pascal Sainrat, Markus Schordan: WCET Tool Challenge 2008: Report. *Proc. 8th International Workshop on Worst-Case Execution Time Analysis (WCET'08),* Prague, Czech Republic, July, 2008
- Enrico Mezzetti, Niklas Holsti, Antoine Colin, Guillem Bernat, Tullio Vardanega: Attacking the Sources of Unpredictability in the Instruction Cache Behavior. Accepted for presentation at the 16th International Conference on Real-Time and Network Systems, Rennes, France, October 16-17, 2008.
- Daniel Kästner, Reinhard Wilhelm, Reinhold Heckmann, Marc Schlickling, Markus Pister, Marek Jersak, Kai Richter, Christian Ferdinand: Timing Validation of Automotive Software. *Proc. 3rd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISOLA)*, Kassandra, Greece, 2008.
- Christian Ferdinand, Florian Martin, Christoph Cullmann, Marc Schlickling, Ingmar Stein, Stephan Thesing, and Reinhold Heckmann: New Developments in WCET Analysis. In Thomas Reps, Mooly Sagiv, and Jörg Bauer, editors, *Program Analysis and Compilation, Theory and Practice. Essays Dedicated to Reinhard Wilhelm on the Occasion of His 60th Birthday (Lecture Notes in Computer Science 4444)*, pages 12-52. Berlin, Springer, 2007.
- Stephan Wilhelm and Björn Wachter: Towards Symbolic State Traversal for Efficient WCET Analysis of Abstract Pipeline and Cache Models. In Christine Rochange, editor, *Proc. 7th International Workshop on Worst-Case Execution Time (WCET) Analysis*, Pisa, Italy, July 3, 2007.

2.4.4 Keynotes, Workshops, Tutorials

Workshop : 8th Int'l Workshop on Worst-Case Execution Time Analysis (WCET'08)
Prague, Czech Republic – July 1st, 2008

The 8th International Workshop on Worst-Case Execution Time Analysis (WCET 2008) was held as a satellite event to the 20th Euromicro Conference on Real-Time Systems (ECRTS 2008) in July 1st 2008 in Prague. The goal of the workshop is to bring together people from academia, tool vendors and users in industry that are interested in all aspects of timing analysis for real-time systems. The workshop fosters a highly interactive format with ample time for in-depth discussions. It provides a relaxed forum to present and discuss new ideas, new research directions, and to review current trends in this area. The presentations are kept short to leave plenty of time for interaction of attendees.

The WCET 2008 event of this workshop has gained on popularity, it had 45 registered participants. This tendency may be interpreted that the real-time community becomes increasingly aware of the importance of WCET analysis. The workshop program included four regular sessions with 13 talks on WCET analysis. Additionally an invited talk on timing analysis at system level had been given by Prof. Rolf Ernst and the current results of the WCET Tool Challenge 2008 had been presented by its organizer, Niklas Holsti.

http://www.artist-embedded.org/artist/WCET-08.html

Tutorial Session: Derivation of Tight Execution-Time Bounds
11th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing
Orlando, Florida, USA – May 5-7, 2008

This tutorial session was organized by Peter Puschner (TU Vienna) and consisted of three presentations, given by members of the ARTIST2 consortium, that presented the very different aspects of and views on the topic. The first presentation, by Raimund Kirner (TU Vienna), described the state of the art and open problems in WCET research. The second talk, by Christian Ferdinand (AbsInt), presented the industrial view and findings from a tool provider's perspective. In the final presentation, Jan Gustafsson (Mälardalen University) reported on the 2006 WCET challenge and its results. The presentations were followed by numerous questions and intensive discussions that underlined the relevance of the topic and the interest that the audience in the problem area and the diverse solution strategies. The topics covered ranged from different analysis techniques (static analysis versus measurement, complexity problems of WCET analysis, alternative architectures to that help to make WCET analysis easier, etc.).

3. Milestones, and Future Evolution Beyond the NoE

3.1 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 50% (AIR defined and implemented, ALF defined, AIR-to-SWEET connection 95% completed).***
  - **the definition and call for the flow description language challenge shall be completed *(100% completed, web site for the challenge has been set up).***

3.2 Indicators for Integration

During the course of the 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.

- 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.
- Tidorum and York (through Rapita Systems Ltd) continue to work for the European Space Agency, now focusing on cache-aware code layout.
- 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.
- The WCET Tool Challenge has been held two times. The organization of this event, as well as the participation, has involved more or less all of the partners.
- Several partners continue the ARTIST2 Timing Analysis work in FP7 projects, such as ALL-TIMES and PREDATOR.
- A substantial number of joint publications have been written.

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

	USaar	AbsInt	York	MDH	TUW	Tidorum	IRISA	UDo
Definition of common interface format AIR	+	+	+	+	+	+	+	
Extension for Computation 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 MDH's flow analysis into aiT		+		+				
Integration MDH's flow analysis into Bound-T				+		+		
Benchmark programs for timing analysis tools	+		+	+		+		

	USaar	AbsInt	York	MDH	TUW	Tidorum	IRISA	UDo
Parametric WCET analysis	+			+				
WCET-oriented IDE				+	+			
WCET Challenge	+	+	+	+	+	+	+	+

3.3 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).
- Mälardalen, AbsInt, York (through Rapita Systems Ltd) , and TU Vienna have funding from the FP7 STREP ALL-TIMES (ICT-215068, duration Dec. 2007 – Feb. 2009).
- Work at York has been funded by BaE systems and by NextTTA EU project IST-2001-32111.
- Tidorum's work is funded only by the ARTIST2 contribution and a minor role in the continuation of the ESA PEAL project.
- 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.
  - ITEA project ES_PASS (started April 1, 2007). Its goal is to establish a product-based assurance for embedded systems. Saarland University is among the partners.
  - EU FP7 project ALL-TIMES (ICT-215068, duration Dec. 2007 – Feb. 2009) aims at combining currently available timing tools, enabling interoperability of tools from SMEs and universities, and developing integrated tool chains using open tool frameworks and interfaces.

- EU FP7 project INTERESTED (Reference 214889) aiming at overcoming the current lack of integration and interoperability of tools for developing Embedded Systems software. The project started 1.1.2008 and will last for 3 years.
- EU FP7 project PREDATOR (Reference 216008) aims at reconciling predictability and efficiency. Saarland and Dortmund are among the partners. The project started 1.2.2008 and will last for 3 years.

3.4 Future Evolution Beyond the ARTIST2 NoE

Most Timing Analysis partners in ARTIST2 will continue to cooperate within the ARTIST-DESIGN FP7 NoE. The Timing Analysis activity wthin ARTIST-DESIGN will be different from ARTIST2: rather than platform integration, this activity will deal with timing analysis for Multicore/MPSoC systems, and the timing predictability problems that arise for such systems. Since this is basically unchartered land, the focus of the activity will be on identifying research issues, and initiating research in the area. Platform integration activities will continue in the ALL-TIMES project, see below.

Several ARTIST2 activity partners participate in the FP7 project ALL-TIMES (Integrating European Timing Analysis Technology), which will run until Feb. 2009. A major theme in ALL-TIMES is the definition and implementation of open interfaces between timing analysis tools, followed by an integration where different tool combinations are investigated. For code level timing analysis (WCET analysis), the work is very much in line with the ARTIST2 platform integration work and paves the way for its future exploitation. ALL-TIMES also considers the integration with system level tools, such as Symta/S from Symtavision. This provides another line of development from the platform integration work performed within ARTIST2.

Furthermore, several ARTIST2 partners partake in the FP7 project PREDATOR (Design for Predictability and Efficiency). The project started in February 2008 and will run for three years. The main topic of PREDATOR is to define future architectures which permit both high performance and good predictability. To this end, it is within the scope of PREDATOR to identify the obstacles of high predictability both on software and on hardware level and to propose solutions to overcome these obstacles while still maintaining high performance. The main goals are to improve the design and development methods for safety-critical embedded systems, to develop tools supporting these development methods, and to develop architectural concepts supporting the derivation of timing guarantees for hard real-time systems and even at providing architectural platforms exhibiting the desired predictability properties. PREDATOR explicitly considers the problem of integrating all parts of the development tool chain into a synergetic process in order to achieve a higher predictability. By considering model based design, code generation, compilation and timing analysis together, new tools will emerge achieving a higher precision of analysis. The applicability of the solutions developed in PREDATOR is ensured by the collaboration with the industrial partners Bosch and Airbus.

Many of the technical objectives mentioned in Section 2.4.1 such as Timing Predictability, Synergetic Implementation, WCET Analysis for Systems with preemptive scheduling, and the design of a WCET aware compiler, have been developed or co-developed by partners involved in PREDATOR. This work will be continued in the PREDATOR project, and it will be the foundation for the future developments in that area.

4. Internal Reviewers for this Deliverable

Niklas Holsti, Tidorum
Thomas Nolte, Mälardalen