Systems Engineering & Design Space Exploration based on the Correctness by Construction Methodology

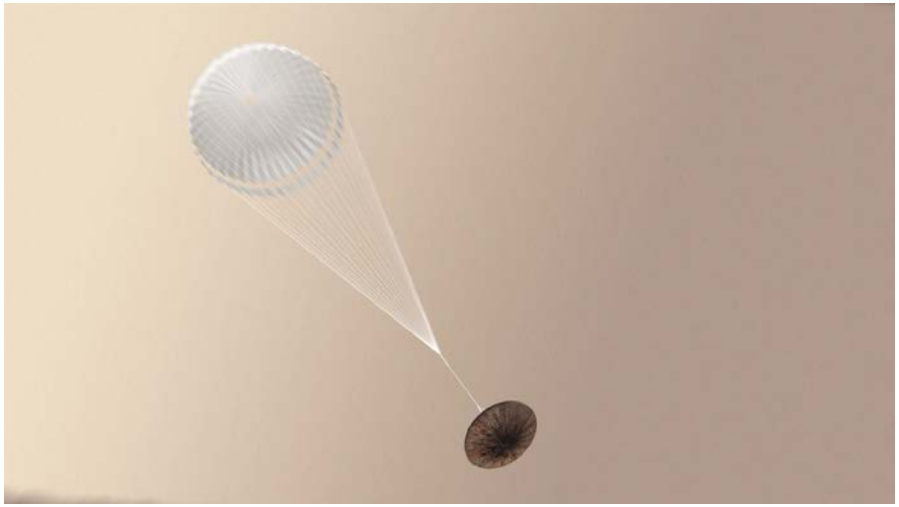
Dr. Robert Hilbrich
German Aerospace Center (DLR)

DLR

German Aerospace Center

Research Institution, Space Agency and Project Management Agency

- Aeronautics
- Space Research and Technology
- Transport
- Energy
- Defense and Security
- Space Administration
- Project Management Agency



Credit: Nonwarit/Fotolia

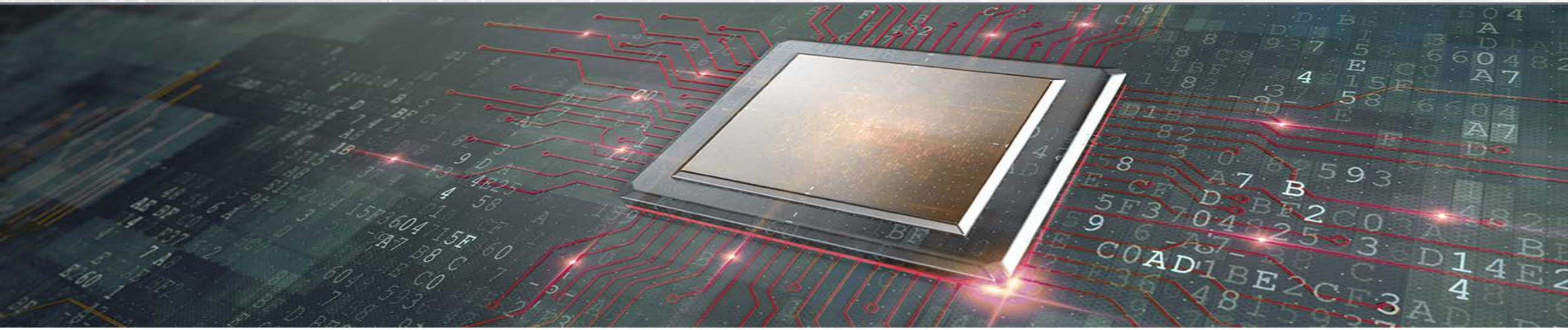
Current Research Project: aramis II

DEVELOPMENT PROCESSES I TOOLS I PLATFORMS
FOR SAFETY-CRITICAL MULTICORE SYSTEMS

STRUCTURED MULTICORE DEVELOPMENT

MULTICORE METHODS AND TOOLS

INDUSTRIAL PLATFORMS FOR MULTICORE SYSTEMS



Current Research Project:

aramis II

DEVELOPMENT PROCESSES I TOOLS I PLATFORMS
FOR SAFETY-CRITICAL MULTICORE SYSTEMS

Automotive

Audi

Continental

DENSO

BOSCH

SCHAEFFLER
LuK INA FAG

Avionics

AIRBUS
AN EADS COMPANY

DIEHL

LIEBHERR
Aerospace.

Software & Tool Vendors

EB

AbsInt

ACCEMIC

Timing Architects

VECTOR

SYMTA VISION

SYSGO
EMBEDDING INNOVATIONS

SILEXICA

OPENSYNERGY

Industry Automation

HIRSCHMANN
A BELDEN BRAND

KSB

GE

SIEMENS

Research-organizations

DLR

TECHNISCHE UNIVERSITÄT
CAROLO-WILHELMINA
ZU BRAUNSCHWEIG

fortiss
innovation in software and systems

C A U
Christian-Albrechts-Universität zu Kiel

TUM
TECHNISCHE UNIVERSITÄT MÜNCHEN

UNIA
Universität Augsburg University

OFFIS

Fraunhofer

KIT
Karlsruher Institut für Technologie

Technische Universität Kaiserslautern

UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR SOFTWARETECHNIK
UND PROGRAMMIERSPRACHEN

DLR

Correctness by Construction

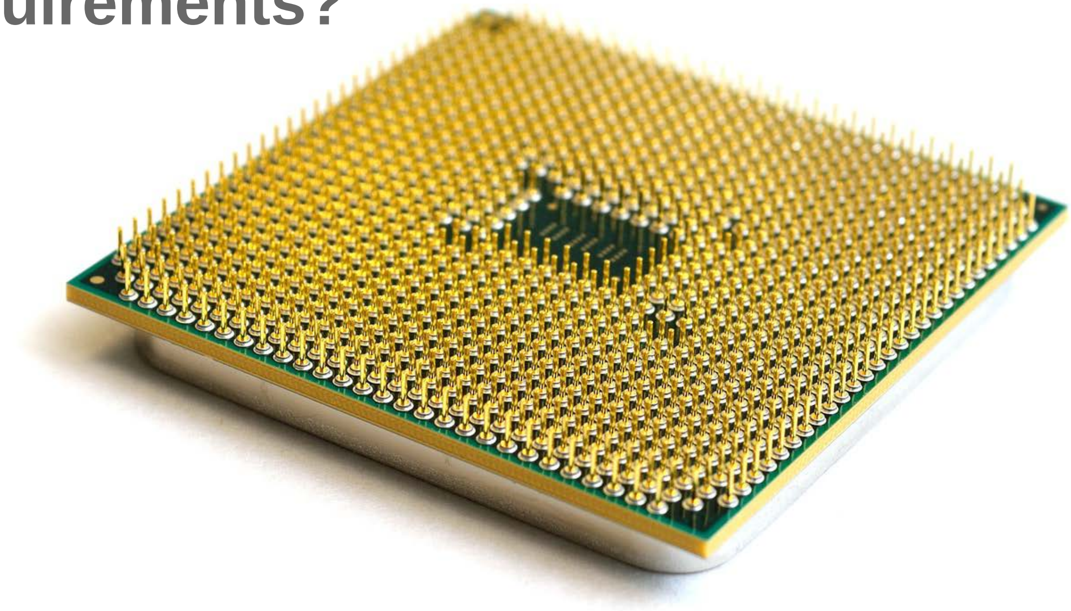
an engineering paradigm for complex safety-critical systems

Trends and Challenges in Safety-Critical Systems

- More (safety-related) functions in software
  - The average (embedded) device now has one million lines of code, and that number is doubling every two years.
  - A modern passenger jet, such as a Boeing 777, depends on 4 million lines of code. Older planes such as a Boeing 747 had only 400,000 lines of code.
- Demand for less power consumption, less weight, less space
- Shorter development cycles

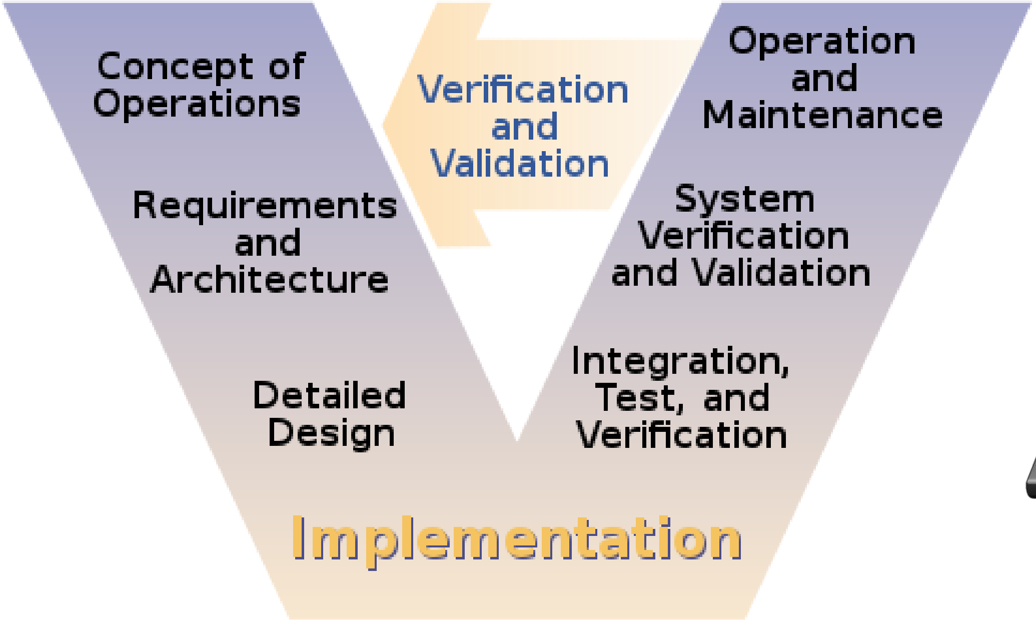
- Solution:
  - Powerful hardware components: multicore processors
  - “Active” migration necessary
  - Increasing “function density”
  - Increased probability for errors and fault propagation

Are our engineering foundations adequate to handle the increasing complexity AND increasing safety requirements?



Engineering Tasks and Responsibilities

Synthesis


Concept of Operations
Requirements and Architecture
Detailed Design
Implementation
Integration, Test, and Verification
System Verification and Validation
Operation and Maintenance
Verification and Validation

Analysis

V-Model for Systems Engineering

FHWA-JPO-05-072, Federal Highway Administration (FHWA), 2005

Limits of Automated Analysis Tools

- Validation requirements are increasing
- Multicore and Parallelism increase complexity
- Space of possible system states is huge(!)

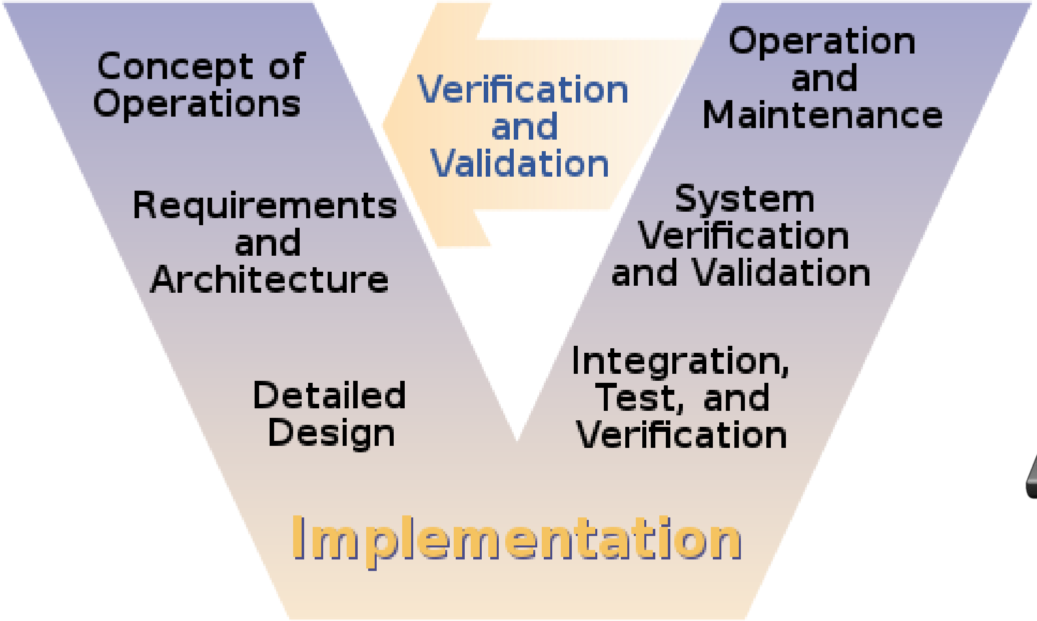
- Current analysis tools are not powerful enough to reach sufficient state coverage

- Idea:

  Argue the correctness of a system based on its **construction process** and not (primarily) on an analysis of its behavior!

Engineering Tasks and Responsibilities

Synthesis


Concept of Operations
Requirements and Architecture
Detailed Design
Implementation
Integration, Test, and Verification
System Verification and Validation
Operation and Maintenance
Verification and Validation

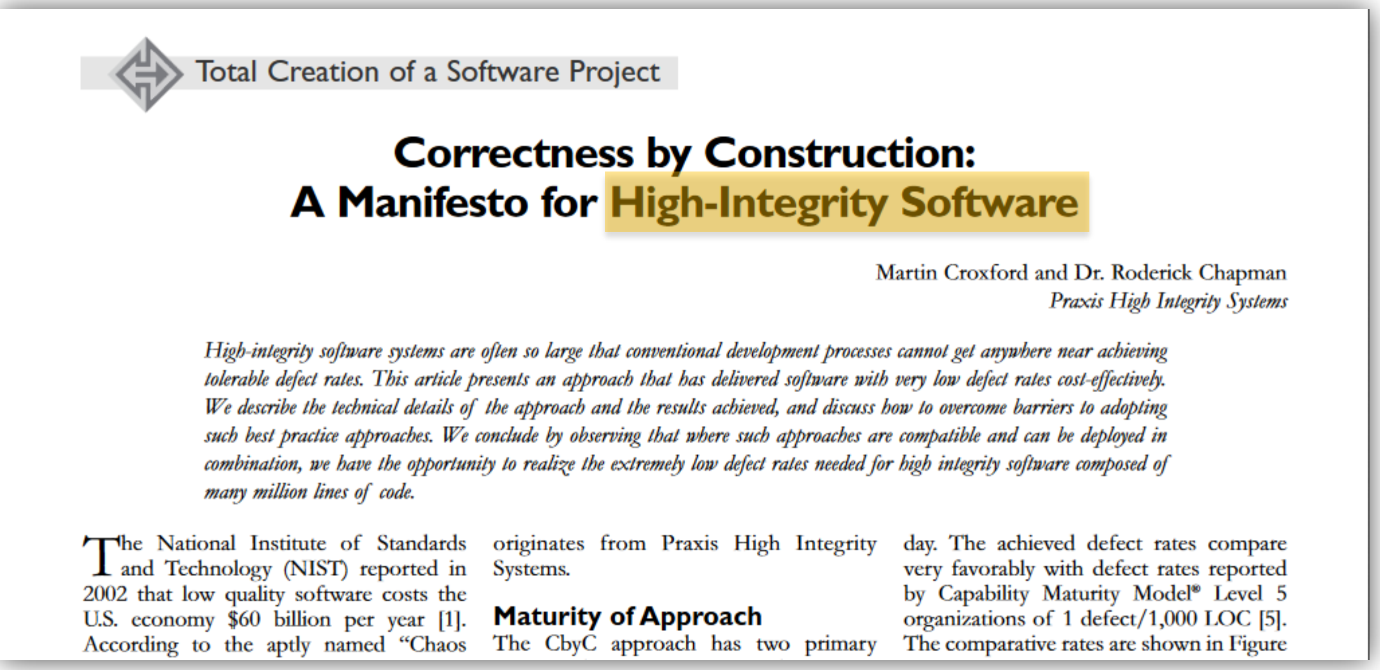
Analysis

V-Model for Systems Engineering

FHWA-JPO-05-072, Federal Highway Administration (FHWA), 2005

Correctness by Construction (CbyC)

Where does it come from?



Total Creation of a Software Project

Correctness by Construction: A Manifesto for High-Integrity Software

Martin Croxford and Dr. Roderick Chapman
Praxis High Integrity Systems

High-integrity software systems are often so large that conventional development processes cannot get anywhere near achieving tolerable defect rates. This article presents an approach that has delivered software with very low defect rates cost-effectively. We describe the technical details of the approach and the results achieved, and discuss how to overcome barriers to adopting such best practice approaches. We conclude by observing that where such approaches are compatible and can be deployed in combination, we have the opportunity to realize the extremely low defect rates needed for high integrity software composed of many million lines of code.

The National Institute of Standards and Technology (NIST) reported in 2002 that low quality software costs the U.S. economy $60 billion per year [1]. According to the aptly named "Chaos

originates from Praxis High Integrity Systems.

Maturity of Approach

The CbyC approach has two primary

day. The achieved defect rates compare very favorably with defect rates reported by Capability Maturity Model® Level 5 organizations of 1 defect/1,000 LOC [5]. The comparative rates are shown in Figure

Martin Croxford and Dr. Roderick Chapman, 2005 (see: http://dl.acm.org/citation.cfm?id=1151820)

Why did I not hear about CbyC before?

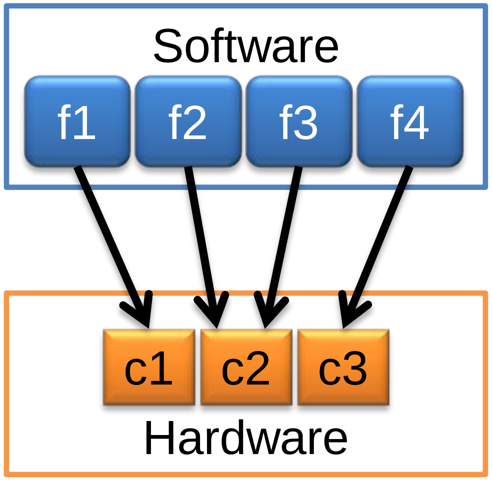
Our Research Goal

1. Apply CbyC to Systems Engineering & Design Space Exploration
2. Bridge the gap between formal methods & engineering practice

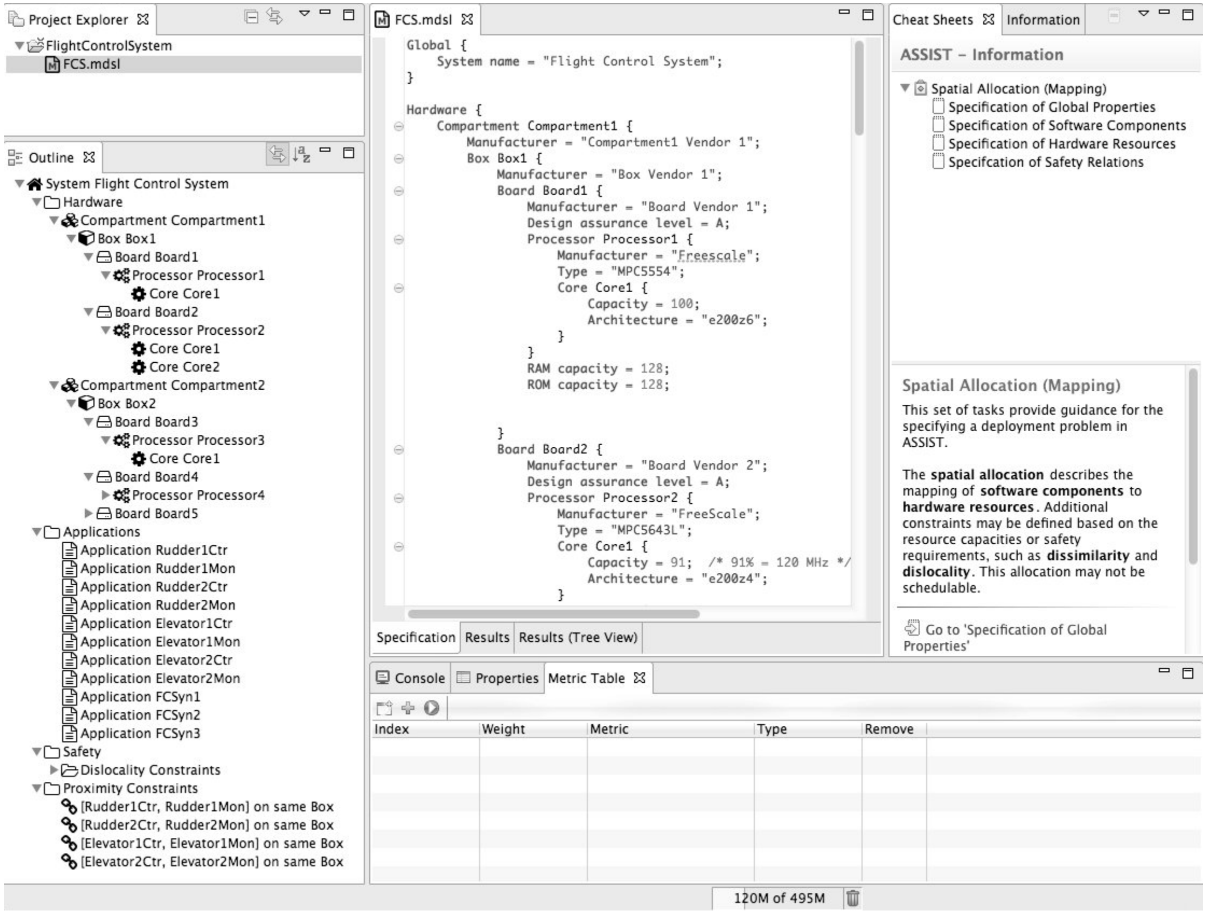
Starting Point:

– Automation of a complex and error prone synthesis task with safety requirements to satisfy

– Deployment of Software Components to Processing Cores (“**Mapping**”)



Architecture Synthesis for Safety Critical Systems (ASSIST)



```
FlightControlSystem
  FCS.mdsl
```

```
System Flight Control System
  Hardware
    Compartment Compartment1
      Box Box1
        Board Board1
          Processor Processor1
            Core Core1
        Board Board2
          Processor Processor2
            Core Core1
            Core Core2
    Compartment Compartment2
      Box Box2
        Board Board3
          Processor Processor3
            Core Core1
        Board Board4
          Processor Processor4
        Board Board5
  Applications
    Application Rudder1Ctr
    Application Rudder1Mon
    Application Rudder2Ctr
    Application Rudder2Mon
    Application Elevator1Ctr
    Application Elevator1Mon
    Application Elevator2Ctr
    Application Elevator2Mon
    Application FCSyn1
    Application FCSyn2
    Application FCSyn3
  Safety
    Dislocality Constraints
  Proximity Constraints
    [Rudder1Ctr, Rudder1Mon] on same Box
    [Rudder2Ctr, Rudder2Mon] on same Box
    [Elevator1Ctr, Elevator1Mon] on same Box
    [Elevator2Ctr, Elevator2Mon] on same Box
```

```
Global {
    System name = "Flight Control System";
}

Hardware {
    Compartment Compartment1 {
        Manufacturer = "Compartment1 Vendor 1";
        Box Box1 {
            Manufacturer = "Box Vendor 1";
            Board Board1 {
                Manufacturer = "Board Vendor 1";
                Design assurance level = A;
                Processor Processor1 {
                    Manufacturer = "Freescale";
                    Type = "MPC5554";
                    Core Core1 {
                        Capacity = 100;
                        Architecture = "e200z6";
                    }
                }
                RAM capacity = 128;
                ROM capacity = 128;


            }
            Board Board2 {
                Manufacturer = "Board Vendor 2";
                Design assurance level = A;
                Processor Processor2 {
                    Manufacturer = "FreeScale";
                    Type = "MPC5643L";
                    Core Core1 {
                        Capacity = 91;  /* 91% = 120 MHz */
                        Architecture = "e200z4";
                    }
```

ASSIST – Information

- Spatial Allocation (Mapping)
  - □ Specification of Global Properties
  - □ Specification of Software Components
  - □ Specification of Hardware Resources
  - □ Specifcation of Safety Relations

Spatial Allocation (Mapping)

This set of tasks provide guidance for the specifying a deployment problem in ASSIST.

The **spatial allocation** describes the mapping of **software components** to **hardware resources**. Additional constraints may be defined based on the resource capacities or safety requirements, such as **dissimilarity** and **dislocality**. This allocation may not be schedulable.

Go to 'Specification of Global Properties'

Index	Weight	Metric	Type	Remove

- Software Tool
- Specify Mapping Problem in **Domain Specific Language**
- Automatic Solution Search & Optimization

Architecture Synthesis for Safety Critical Systems (ASSIST)

- How?
  - **Automatic** transformation of the specification into a “Constraint Satisfaction Problem (CSP)”
  - Uses CHOCO Solver (Open Source!)
  - Solutions can be formally proven to be correct(!)
  - Very efficient modeling!
  - Quickly modified for special requirements

- Technical Requirements of ASSIST:
  - Runs on Linux, Windows, OSX
  - Works on regular Laptop
  - Open Source License (soon: Eclipse Foundation)
  - Url: http://assist.hilbri.ch

Architecture Synthesis for Safety Critical Systems (ASSIST)

- Experiences in practice
  - Textual input was very welcome!
    (precise, efficient, Excel export, …)
  - Significant reduction of engineering effort
    (3 – 12 person months → 10 mins with ASSIST)
  - Very valuable for Systems Engineer to have several solutions and quickly explore alternatives

RESEARCH

Robert Hilbrich

Platzierung von Software-komponenten auf Mehrkernprozessoren

Automatisierte Konstruktion und Analyse für funktionssichere Systeme

Springer Vieweg

Summary

- Correctness by Construction –
  an engineering paradigm to help you
  when analysis tools reach the limit of their capability
- Complex synthesis tasks can be automated
- Tool: Architecture Synthesis for Safety-Critical Systems (ASSIST)
- Successful application in real-world use-cases

Robert.Hilbrich@dlr.de