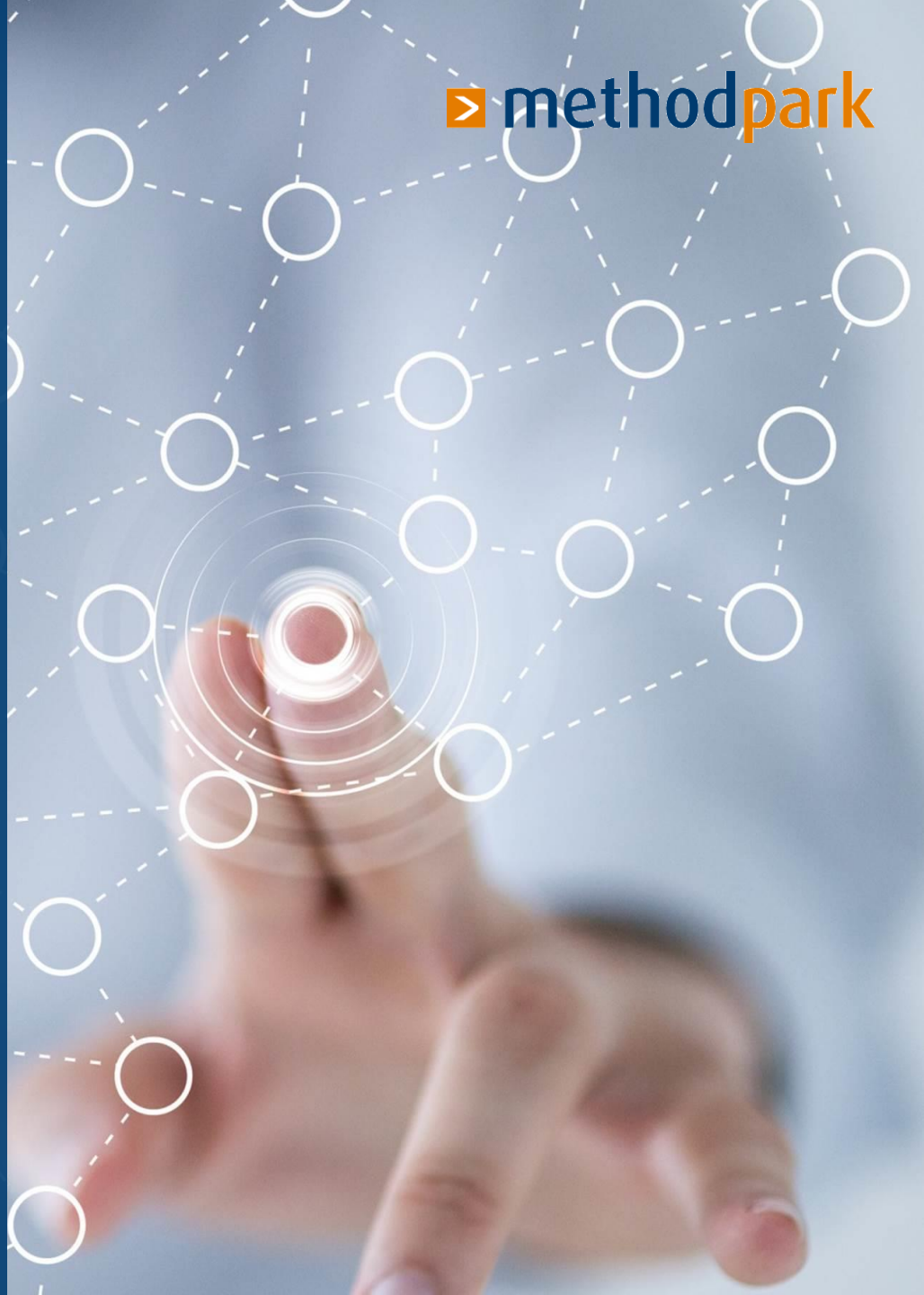
Using Generic Software Components for Safety-Critical Embedded Systems

Felix Bräunling

January, 31st 2020

ERTS 2020



methodpark

aramis II

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

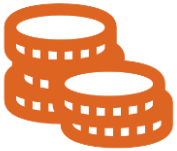
This work was funded by the German Federal Ministry for Education and Research (BMBF) within the project ARAMiS II with the funding ID 01IS16025B. The responsibility for the content remains with the authors.

Conflicting Priorities

methodpark

Safety

Cost



Efficiency

Reusability

Special vs. Platform Development

methodpark

Software Adaptation

methodpark

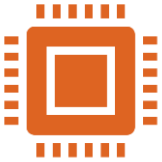
Software Platform

Adapting

Specific Product

Run Time Efficiency

Memory Efficiency



Specialized Hardware

Temporal Isolation

Spatial Isolation

Software Adaptation

methodpark

Software Platform

Adapting

Specific Product

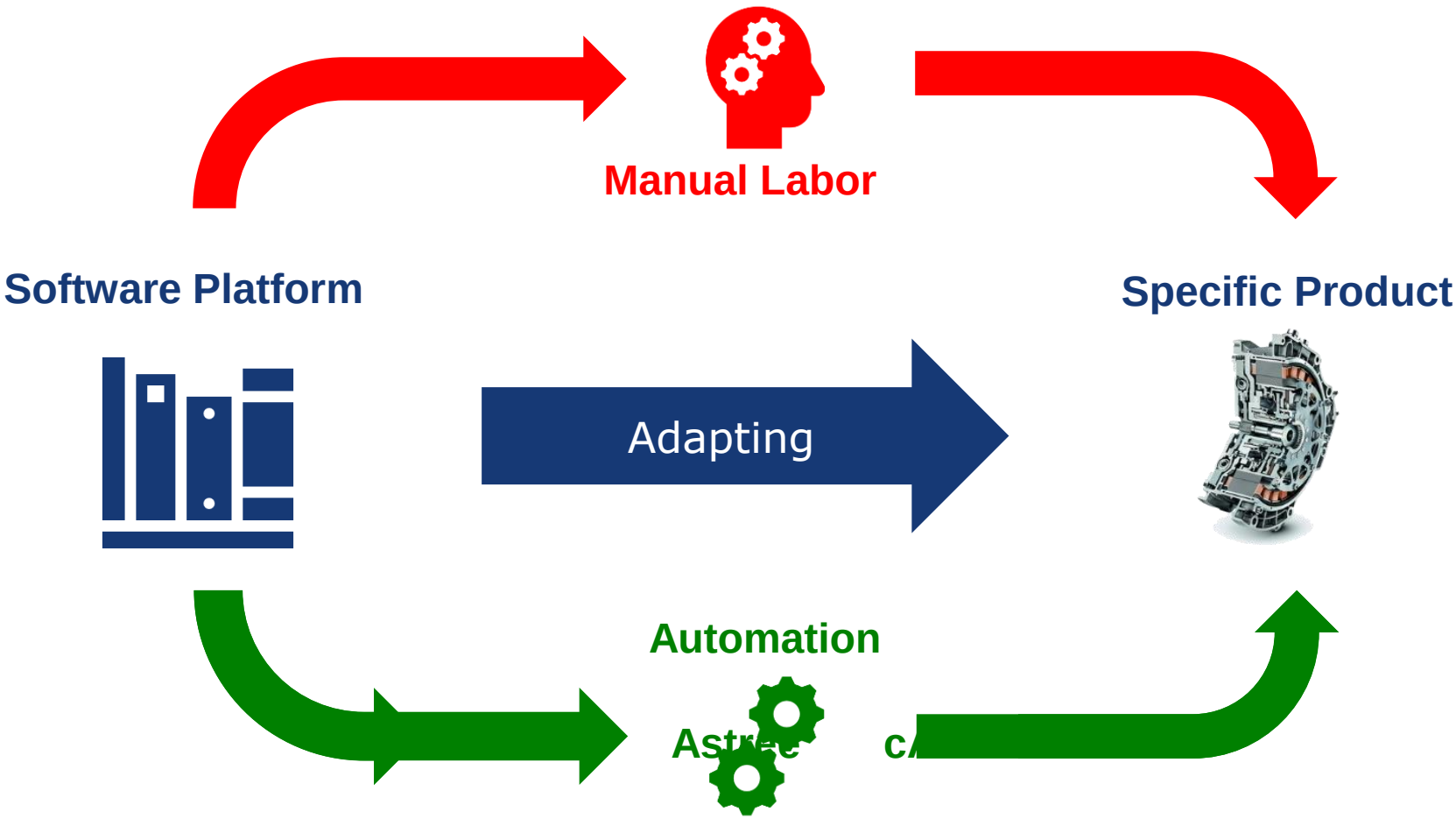
System Architecture

Software Architecture

Software Implementation

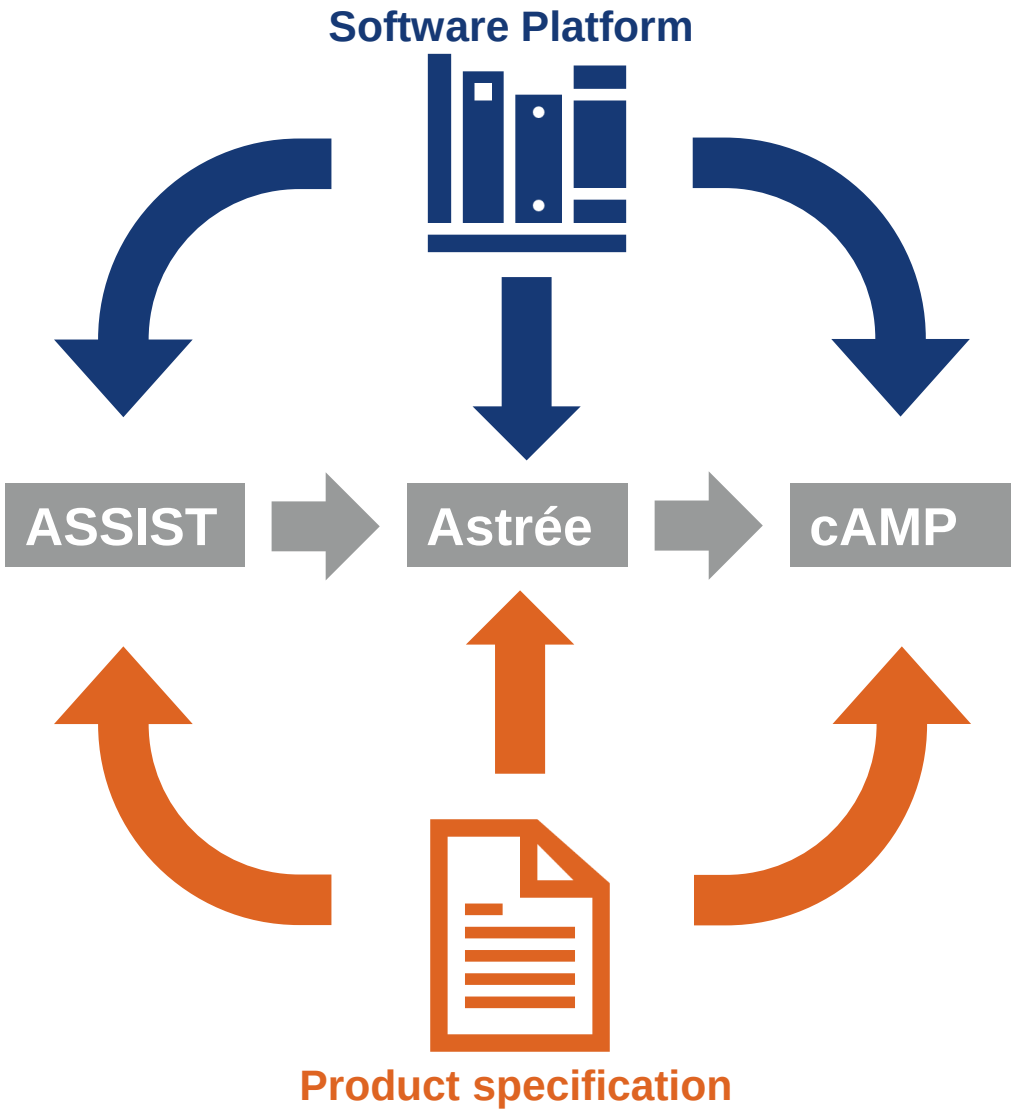
Software Adaptation

methodpark



Adaptation Framework

methodpark



- Deployment
- Scheduling
- Type & Memory Safety
- Data Flow and Access
- Data Partitioning
- Memory Binding

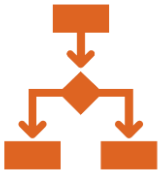
Adaptation Technologies

methodpark

Software Platform

Adapting

Specific Product



Model-based Development

Static Analysis

Code Generation

Adaptation Base

methodpark

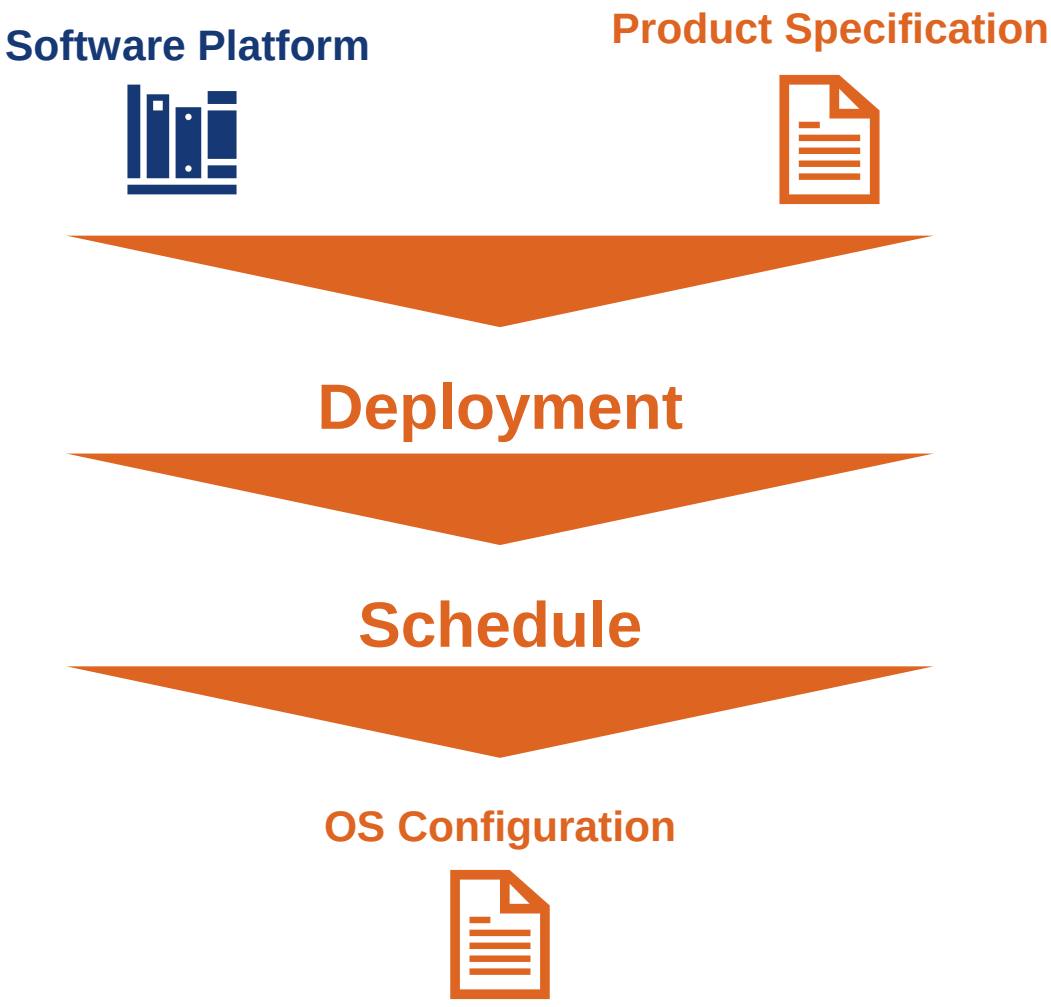
Software Platform

- **Basic Software Library**
- **Model-based Application Library**
- **Platform Target Hardware Description**

Product Specification

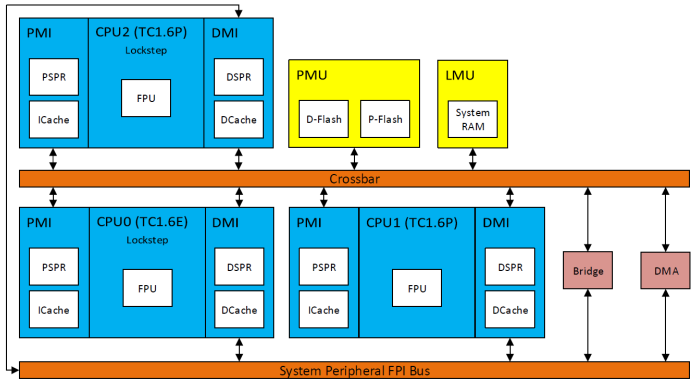
- **Safety Requirements**
- **Functional Architecture**
- **Non-functional Requirements**
- **Specific Hardware**

methodpark

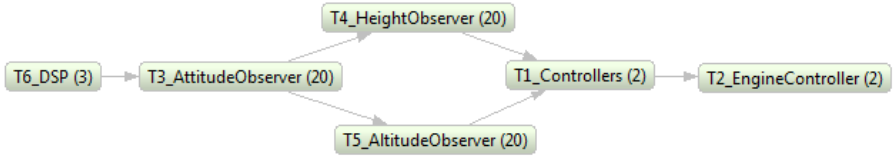

Software Platform
Product Specification
Deployment
Schedule
OS Configuration

ASSIST

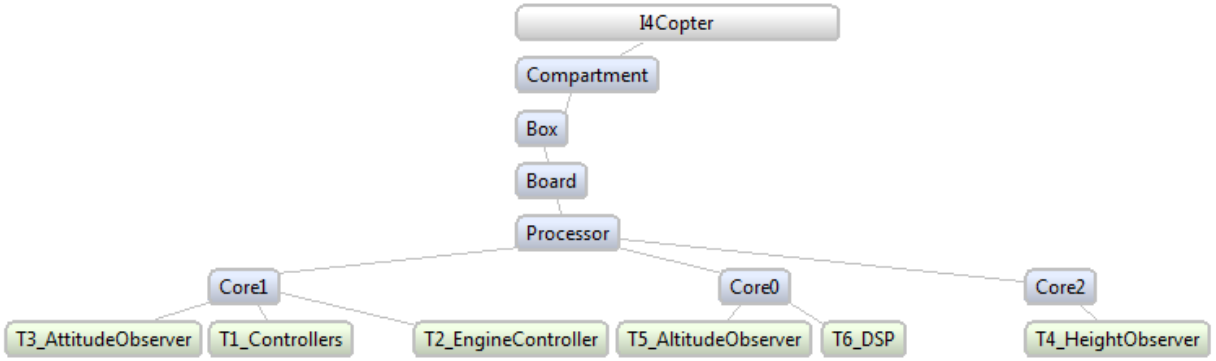
methodpark



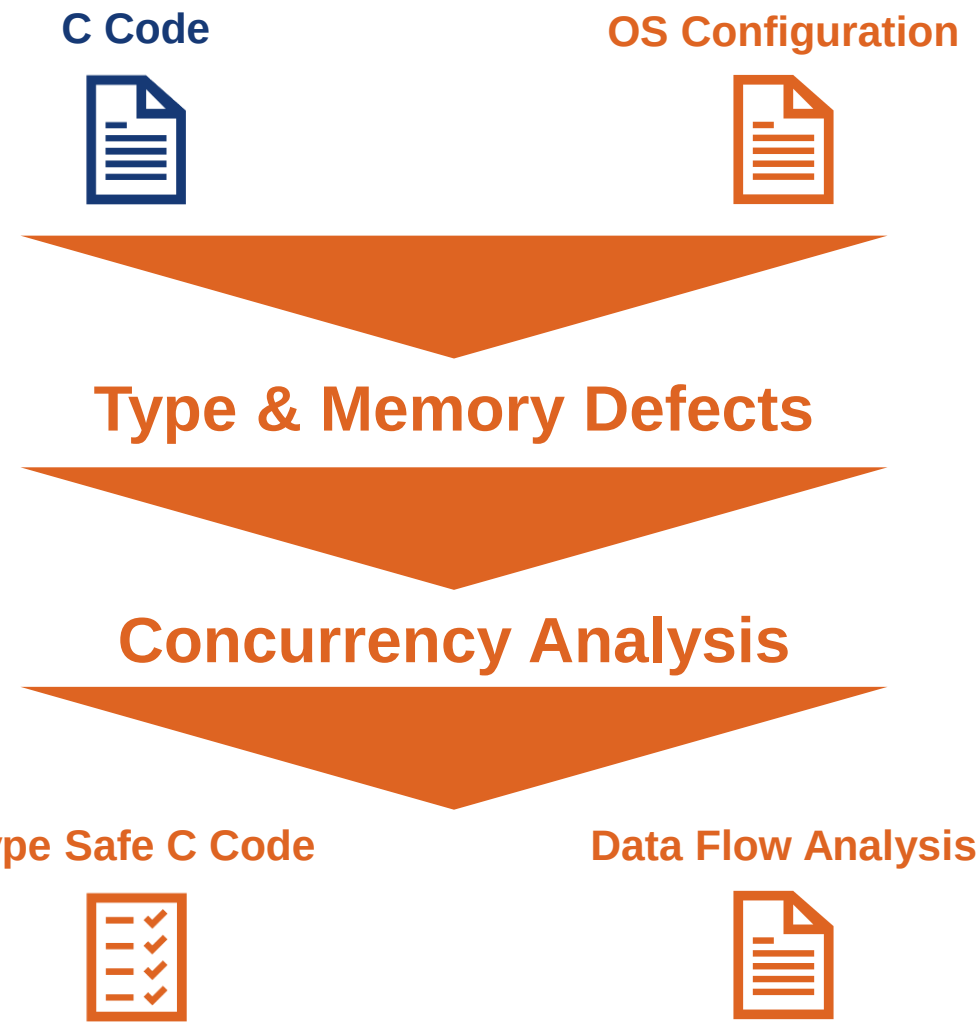
- Offers Floating Point Unit
- Offers Lock-Step Core
- Offers DMA
- Provides 4MB of Flash
- Provides 128kB of SRAM



- Requires MPU
- Requires FPU
- Takes 12kB of Flash
- Takes 960B of SRAM



methodpark

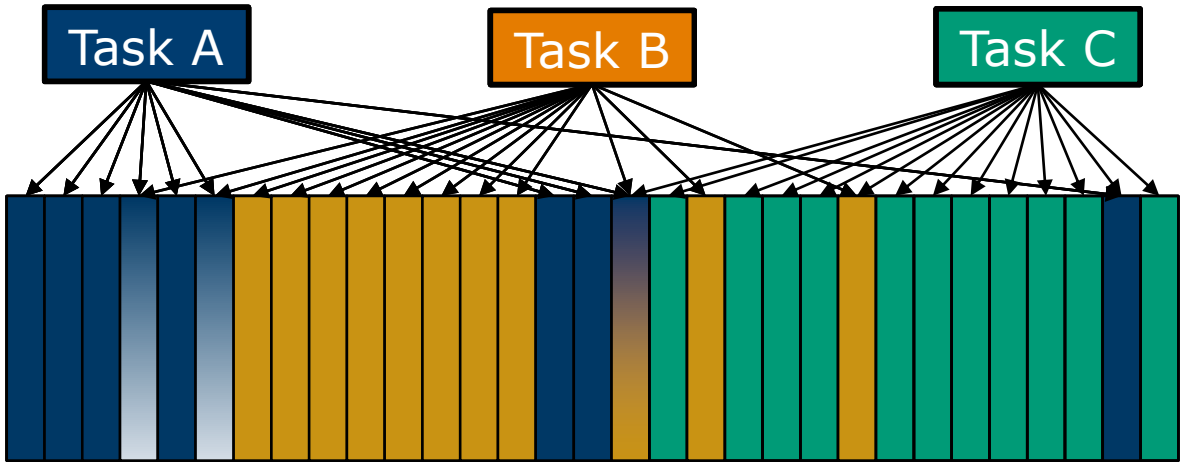

C Code
OS Configuration
Type & Memory Defects
Concurrency Analysis
Type Safe C Code
Data Flow Analysis

Astrée

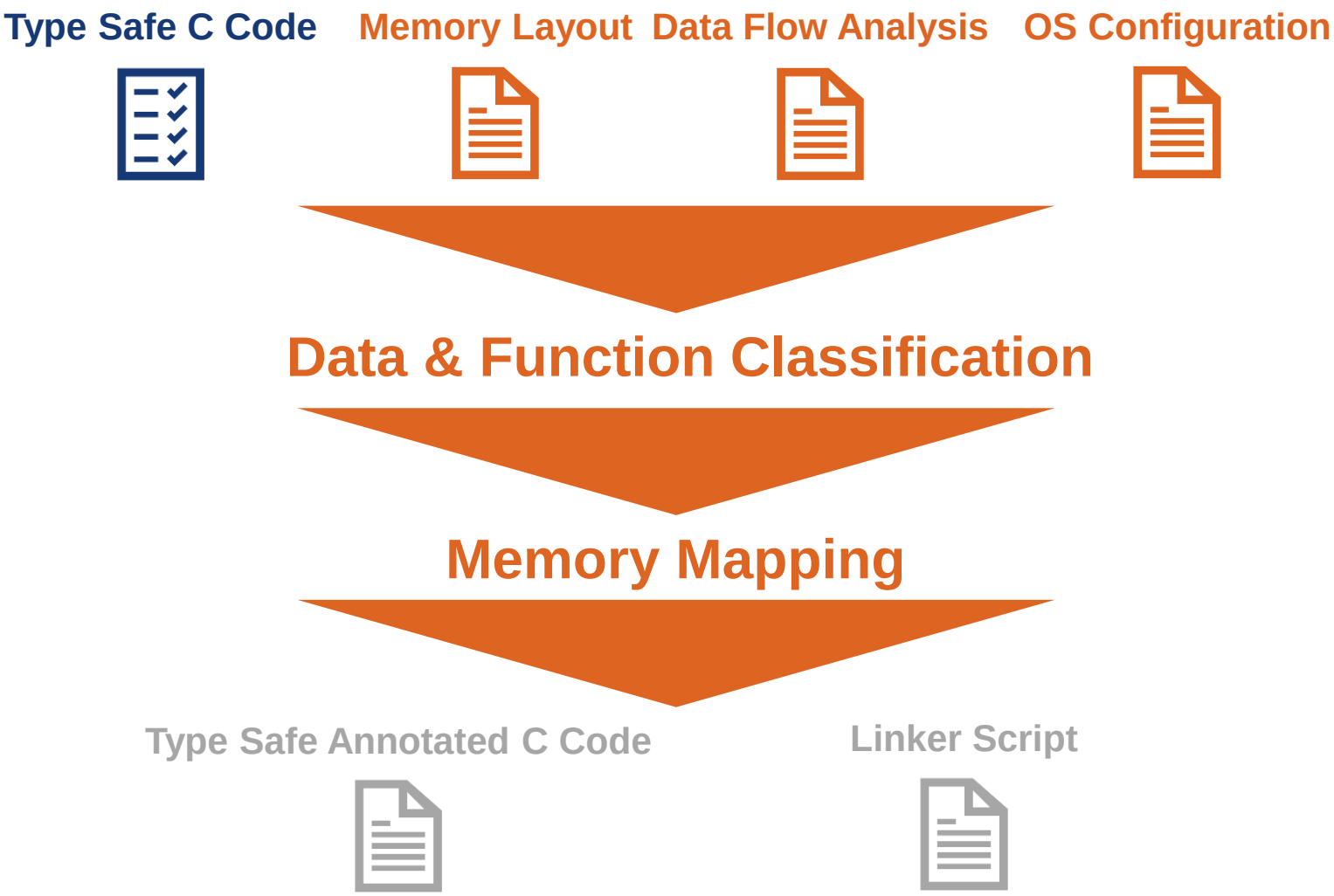
methodpark

```
int a[ARRAY_LEN];
int x = -10;
for(int i = 0; i <= ARRAY_LEN; i++) {
   a[i] = 0;
}
```

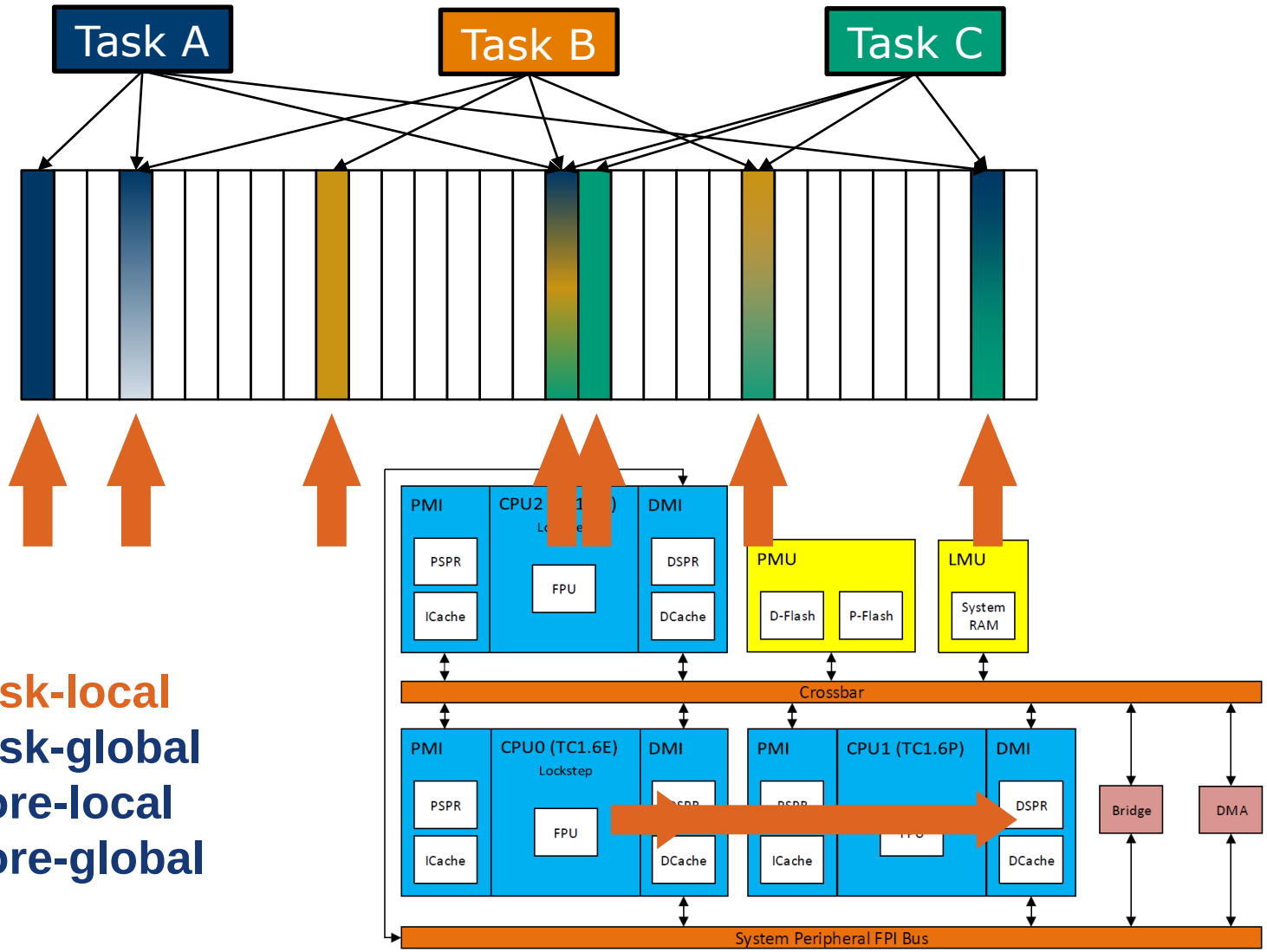
Out-of-bounds Array Access



methodpark


Type Safe C Code
Memory Layout
Data Flow Analysis
OS Configuration
Data & Function Classification
Memory Mapping
Type Safe Annotated C Code
Linker Script

methodpark



- **Task-local**
- **Task-global**
- **Core-local**
- **Core-global**

Summary

methodpark

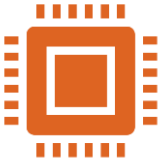
Software Platform

Adapting

Specific Product

Run Time Efficiency

Memory Efficiency



Specialized Hardware

Temporal Isolation

Spatial Isolation

Felix Bräunling
felix.braeunling@methodpark.de

Robert Hilbrich
robert.hilbrich@dlr.de

Simon Wegener
swegener@absint.com

Isabella Stilkerich
isabella.stilkerich@schaeffler.com

Daniel Kästner
kaestner@absint.com