Towards Dynamic Composition of Dependable Embedded Systems

Leandro Batista Ribeiro
lbatistaribeiro@tugraz.at

Institute of Technical Informatics
Embedded Automotive Systems Group
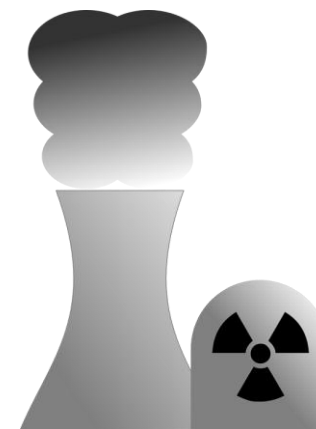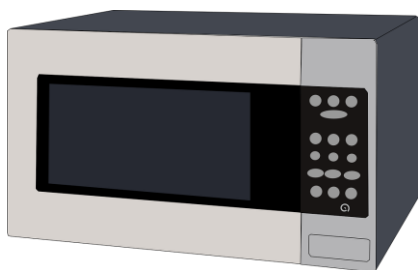Graz University of Technology

Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems
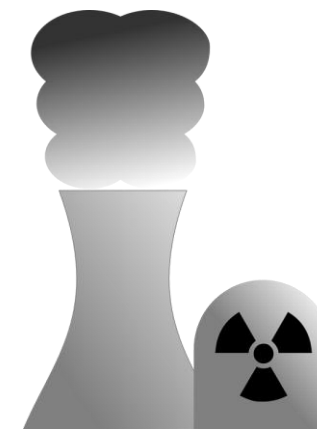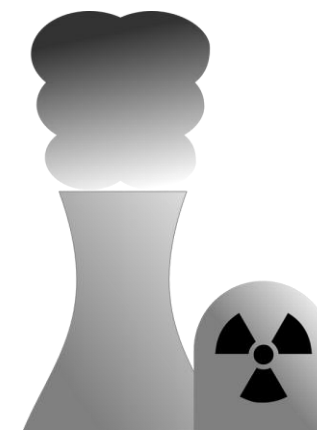
Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems
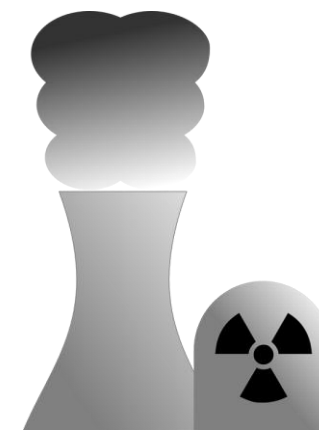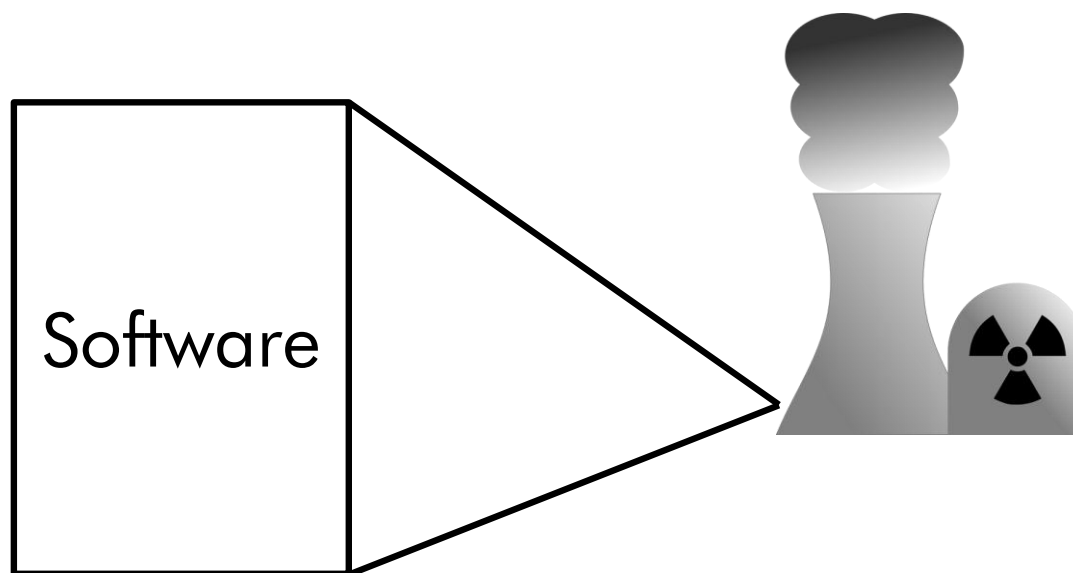
Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems

Towards Dynamic **Composition** of Dependable Embedded Systems

Towards Dynamic Composition of Dependable Embedded Systems

Modules

Software

Towards Dynamic **Composition** of Dependable Embedded Systems

Modules

Build → Software
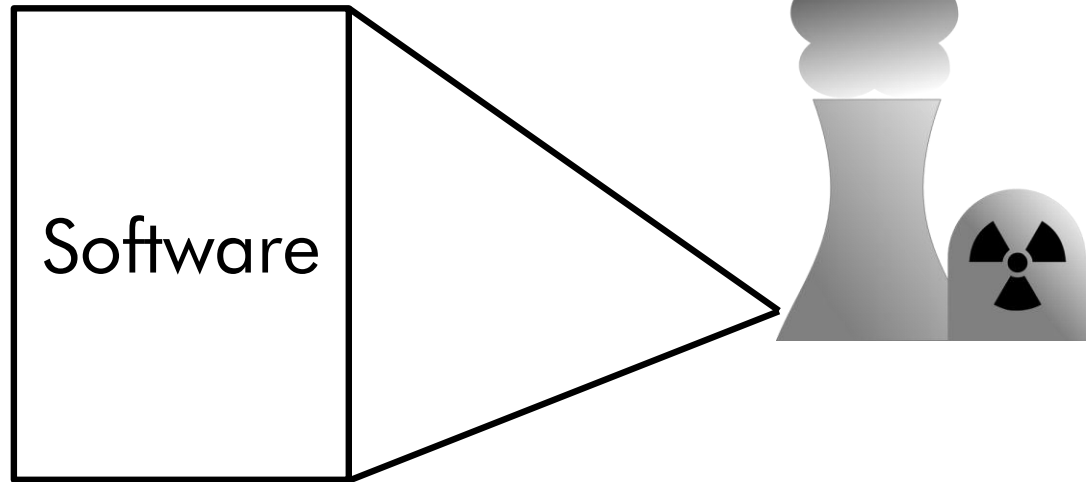
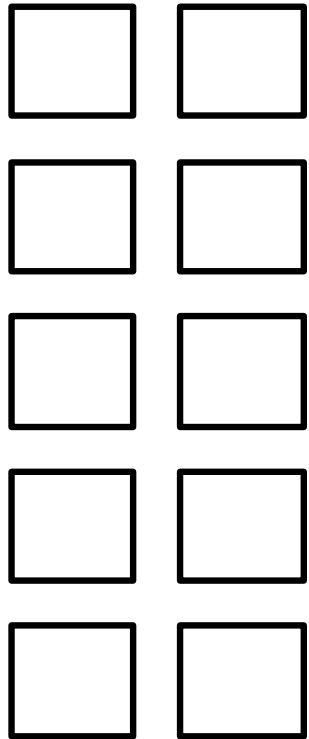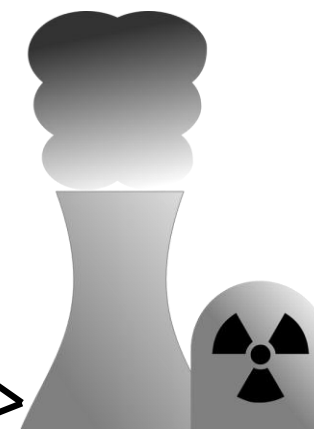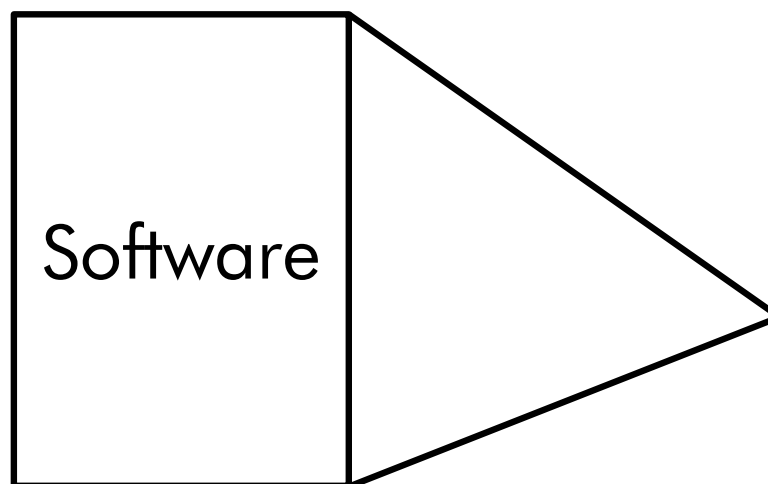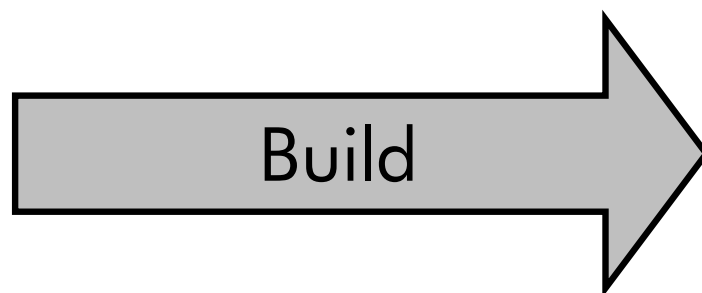# Towards Dynamic **Composition** of Dependable Embedded Systems

Modules

Towards Dynamic Composition of Dependable Embedded Systems
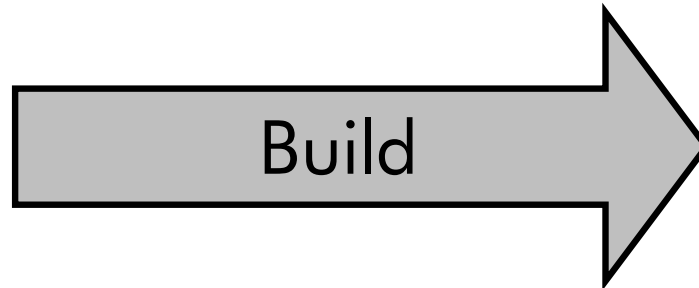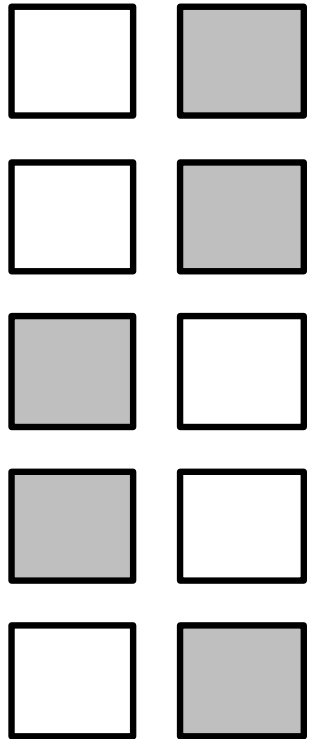
Software
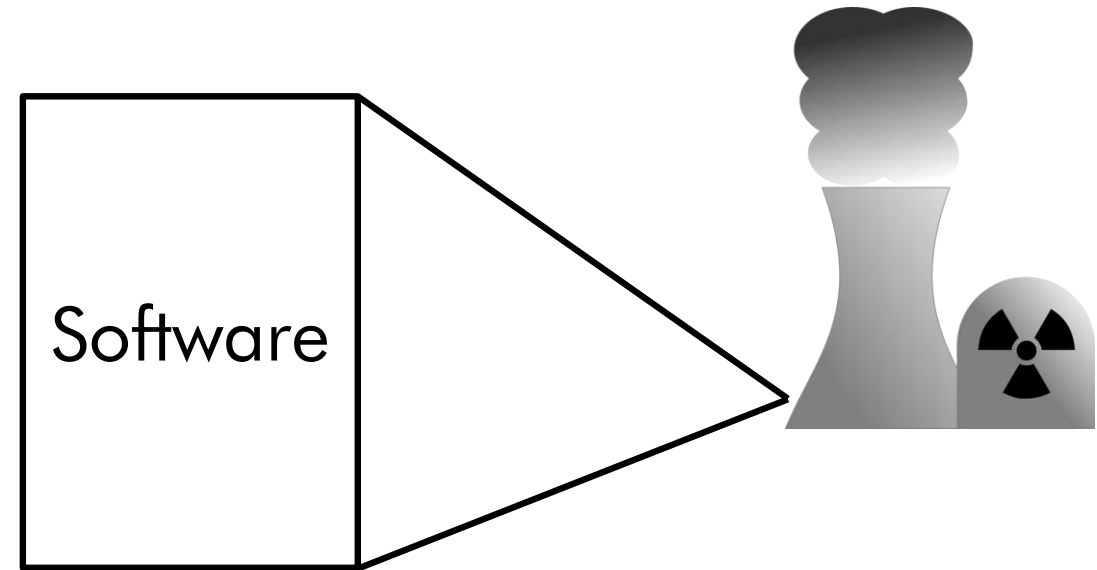
Towards **Dynamic Composition** of Dependable Embedded Systems

Integration at Runtime

Towards Dynamic Composition of Dependable Embedded Systems
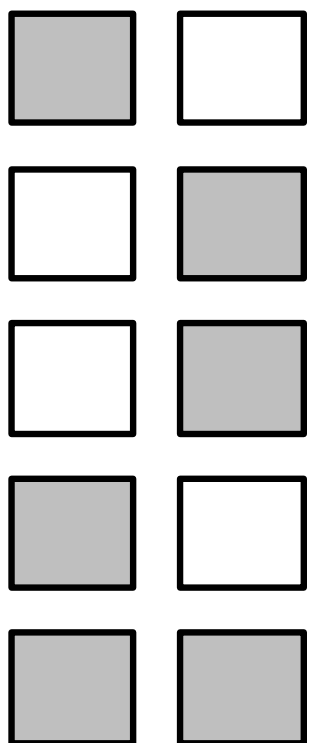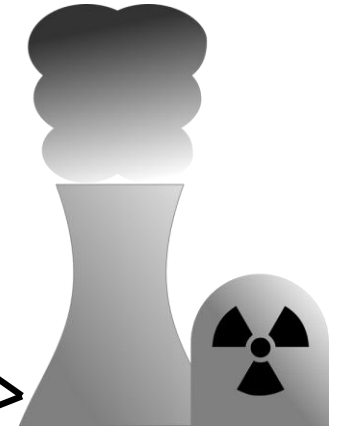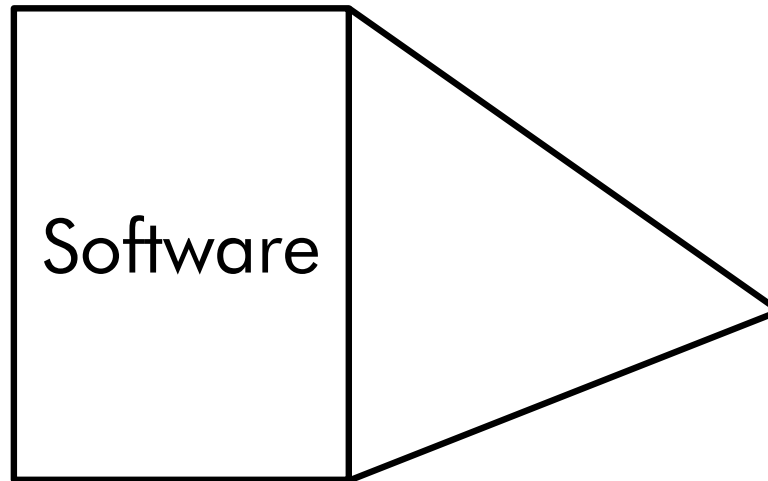

Integration at Runtime

Module

Transmission

Software

Towards Dynamic Composition of Dependable Embedded Systems

Integration at Runtime

Module

Transmission

Software
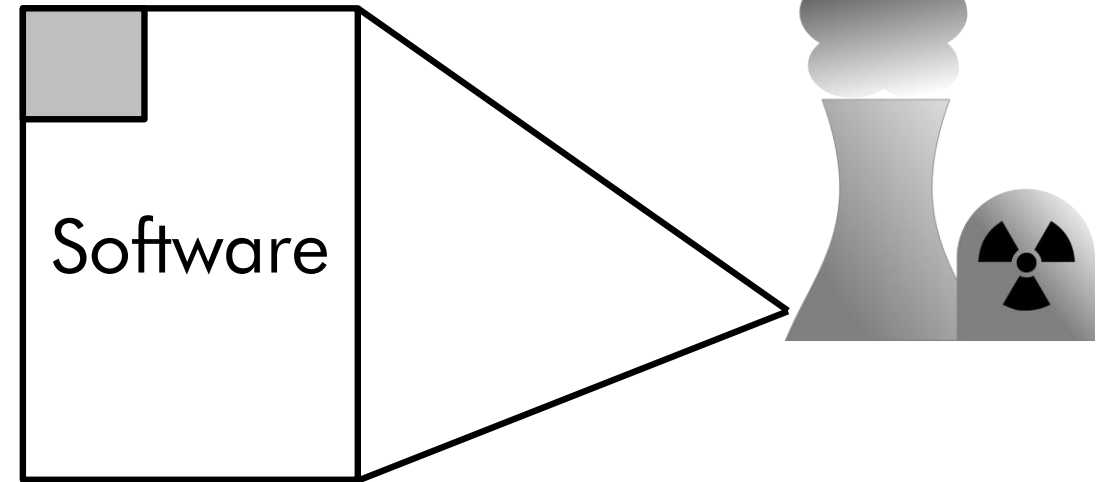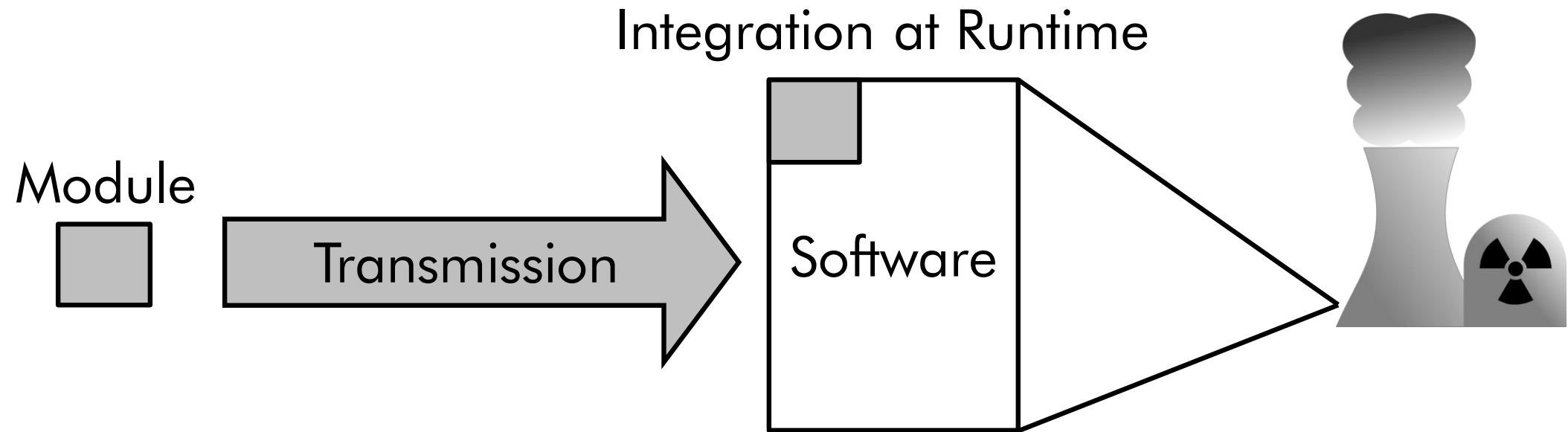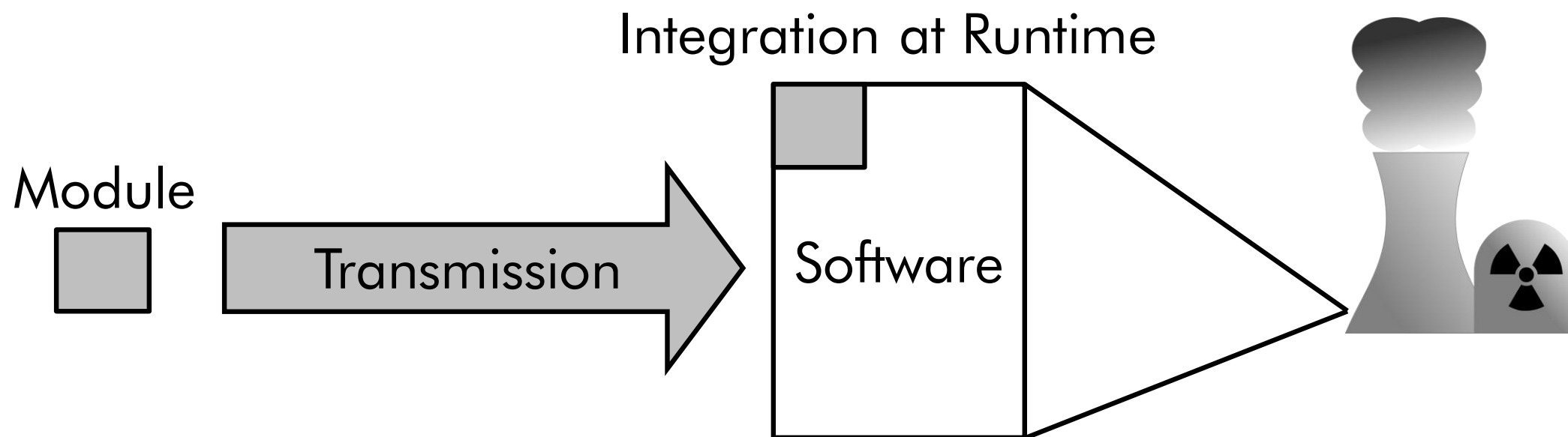
Inherent Problems

Inherent Problems
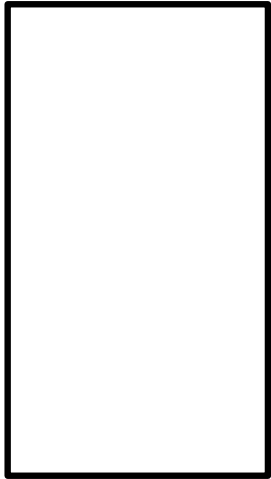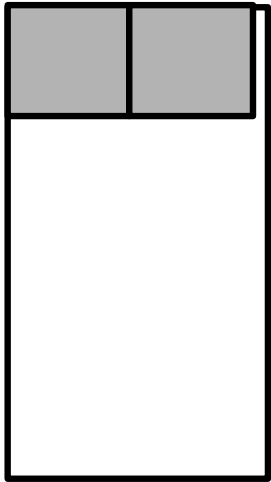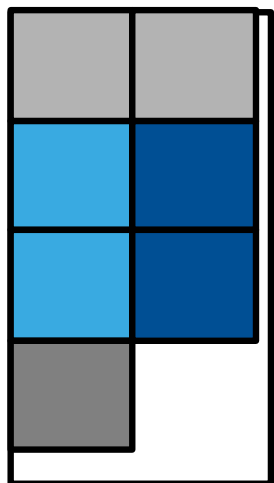
Memory

Inherent Problems

Memory

Inherent Problems

Memory

Inherent Problems
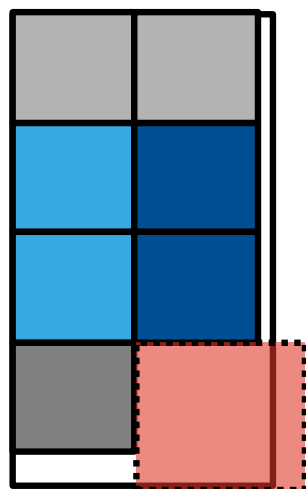
Memory

Inherent Problems
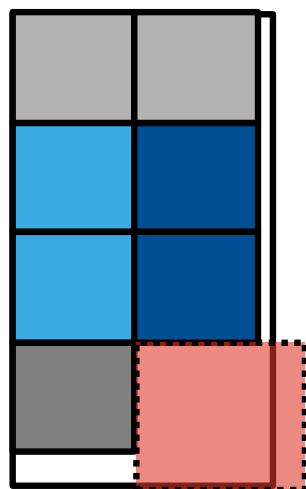
Memory

Dependencies

Inherent Problems
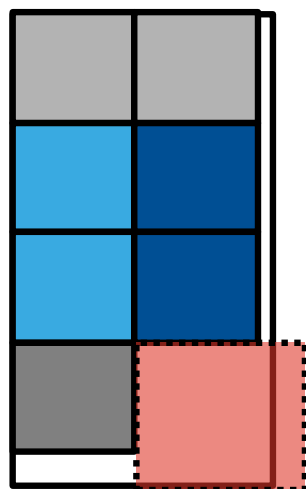
Memory

Dependencies

Inherent Problems
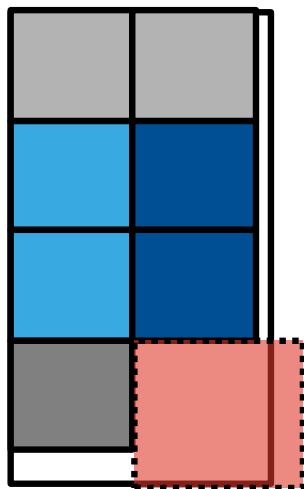
Memory

Dependencies

Behavior

Inherent Problems
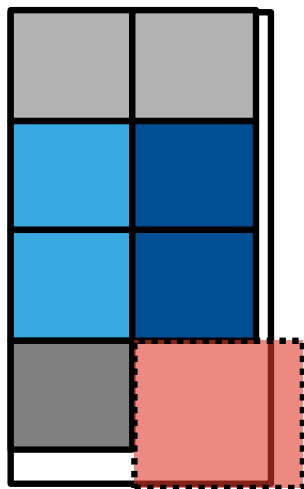
Memory

Dependencies

Behavior

Functional Requirements

Inherent Problems

Memory
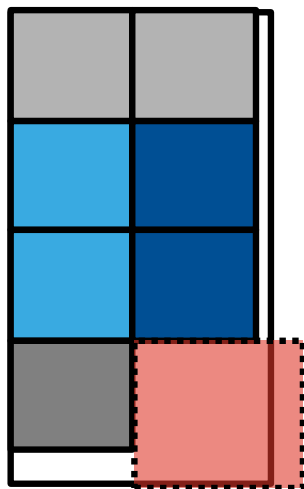
Dependencies

Behavior

(Non-)Functional Requirements

Research Question

How to partially update embedded systems and guarantee the correctness of the resulting software composition?

Research Question

How to **partially update** embedded systems **and guarantee the correctness** of the resulting software composition?

Partial Updates

SmartOS and Update Protocol

Memory
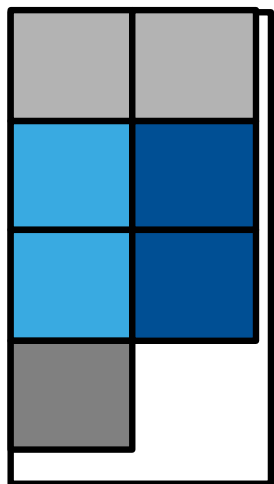
Dependencies

Partial Updates

SmartOS and Update Protocol
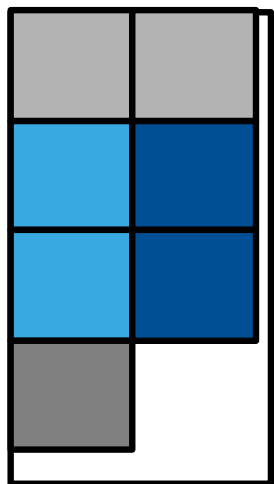
EWSN'20

Memory

Dependencies

SmartOS

Sustainable modular adaptive real-time Operating System

Partial Updates

SmartOS and Update Protocol

EWSN'20

Memory

Dependencies

SmartOS

Sustainable modular adaptive real-time Operating System

Server

Device
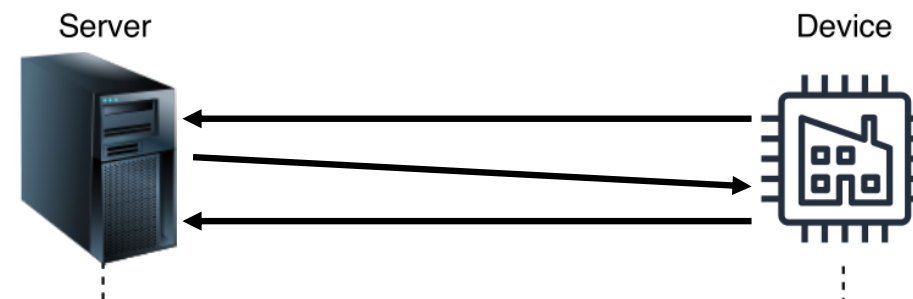
Partial Updates

Maintainability
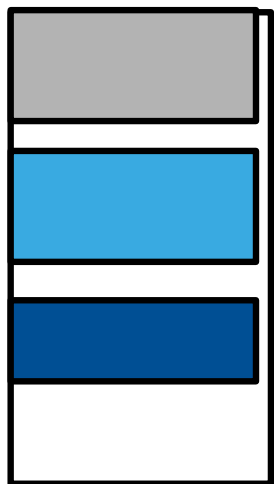
Memory

Partial Updates

Maintainability
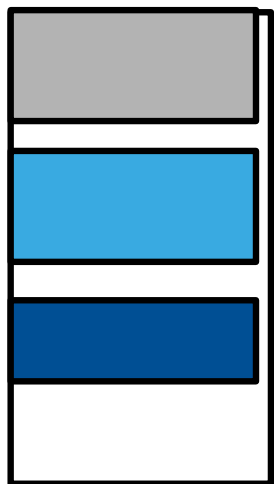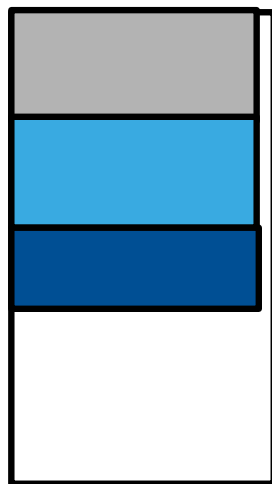
Memory   Defrag.

Partial Updates

Maintainability
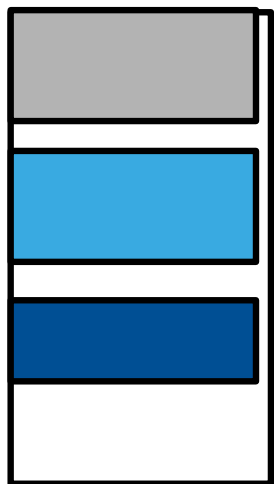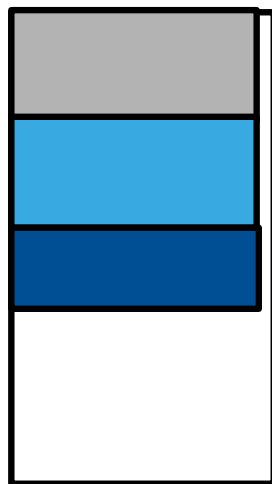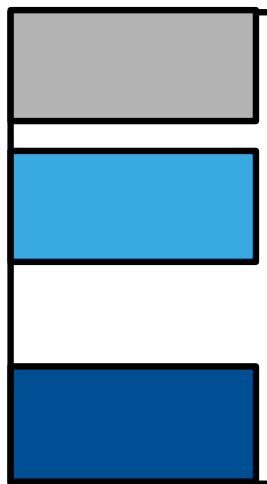
Memory   Defrag.   or   Update

Partial Updates

Maintainability

Memory   Defrag.   or   Update

Partial Updates

Maintainability

Memory   Defrag.   or   Update

Towards Dynamic Composition of Dependable Embedded Systems
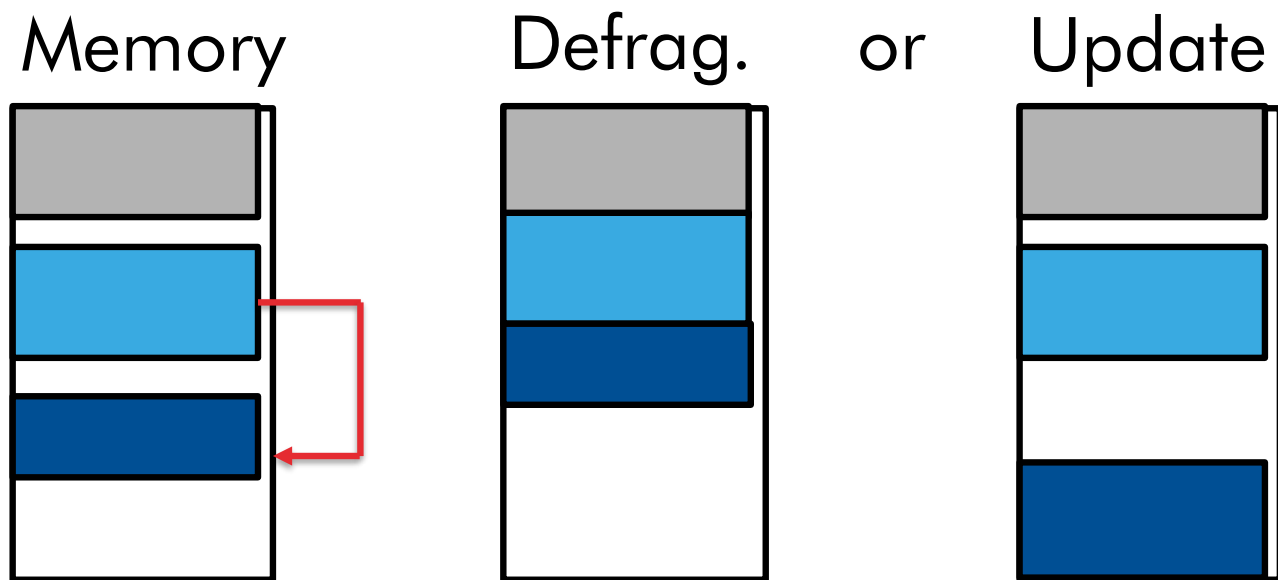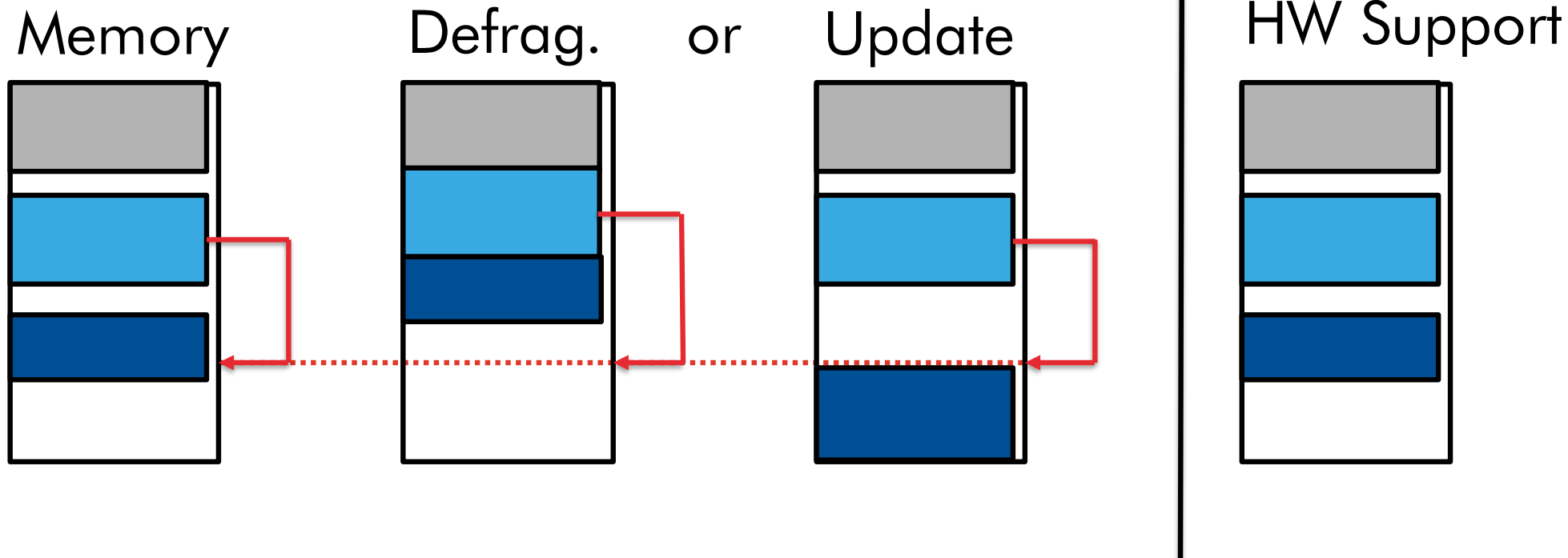
Partial Updates

Maintainability
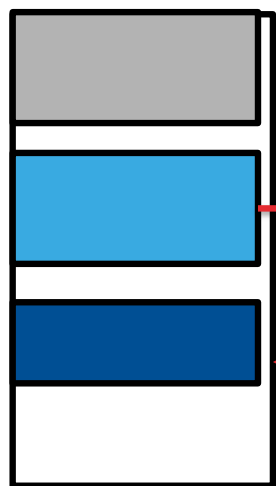
CODES'21

Memory   Defrag.   or   Update
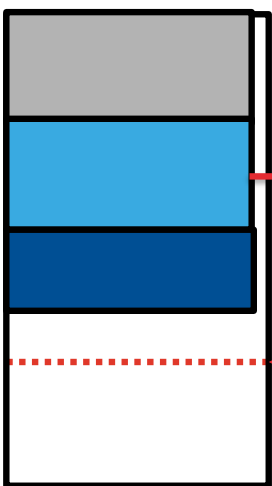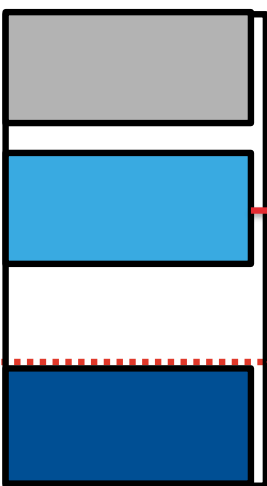
HW Support
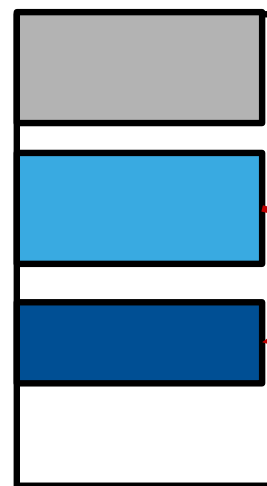
HW Relocation

Correctness

Our Focus

Behavior

(Non-)Functional Requirements

Correctness

Our Focus

- Real-Time

Behavior

(Non-)Functional Requirements

Correctness

Our Focus

- Real-Time

- Liveness

Behavior

(Non-)Functional Requirements

Correctness

Our Focus

- Real-Time

- Liveness

Dining Philosophers

[DP]

Correctness

Our Focus

- Real-Time

- Liveness
  - Freedom of Starvation

Dining Philosophers

[DP]

Correctness

Our Focus

- **Real-Time**

- **Liveness**
  - **Freedom of Starvation**

Dining Philosophers

[DP]

Correctness
Our Focus

- Real-Time

- Liveness
  - Freedom of Starvation

Dining Philosophers

[DP]

Correctness

Our Focus

- **Real-Time**

- **Liveness**
  - Freedom of Starvation

Dining Philosophers

[DP]

Correctness
Our Focus

- **Real-Time**

- **Liveness**
  - Freedom of Starvation
  - Freedom of Deadlock

Dining Philosophers

[DP]

Correctness
Our Focus

- **Real-Time**

- **Liveness**
  - Freedom of Starvation
  - Freedom of Deadlock

Dining Philosophers

[DP]

Correctness

Formal Methods

- Model and Verify Software Behavior

Correctness

Formal Methods

- Model and Verify Software Behavior

Correctness

Formal Methods

- Model and Verify Software Behavior

Correctness

Formal Methods

- Model and Verify Software Behavior

Correctness

Formal Methods

- Model and Verify Software Behavior

- Examples of Requirements
  - A deadlock must never occur
  - A given error location must never be reached
  - Internal lists must be sorted at all times
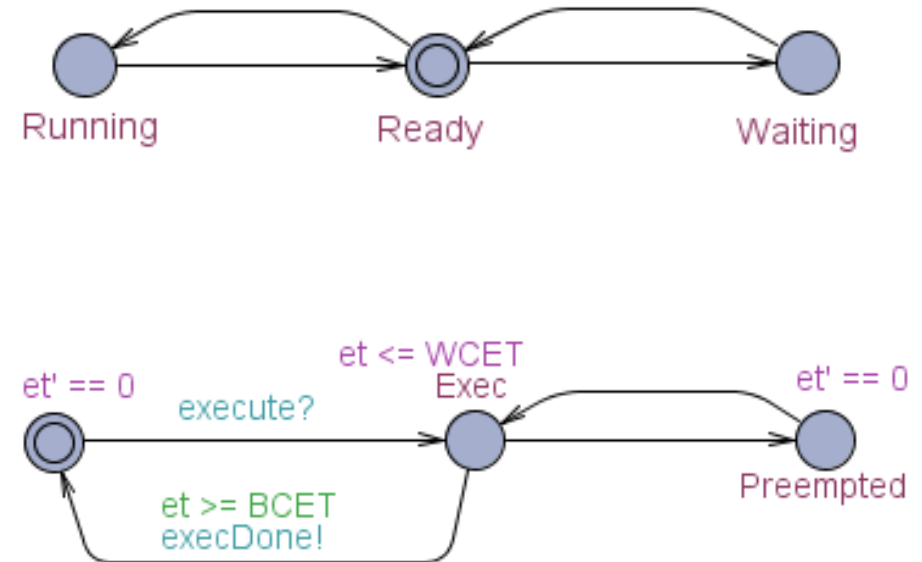
Correctness

Layered Compositional Model

Correctness

Layered Compositional Model

Correctness

Layered Compositional Model

Correctness

Layered Compositional Model

Correctness

Layered Compositional Model

Towards Dynamic Composition of Dependable Embedded Systems
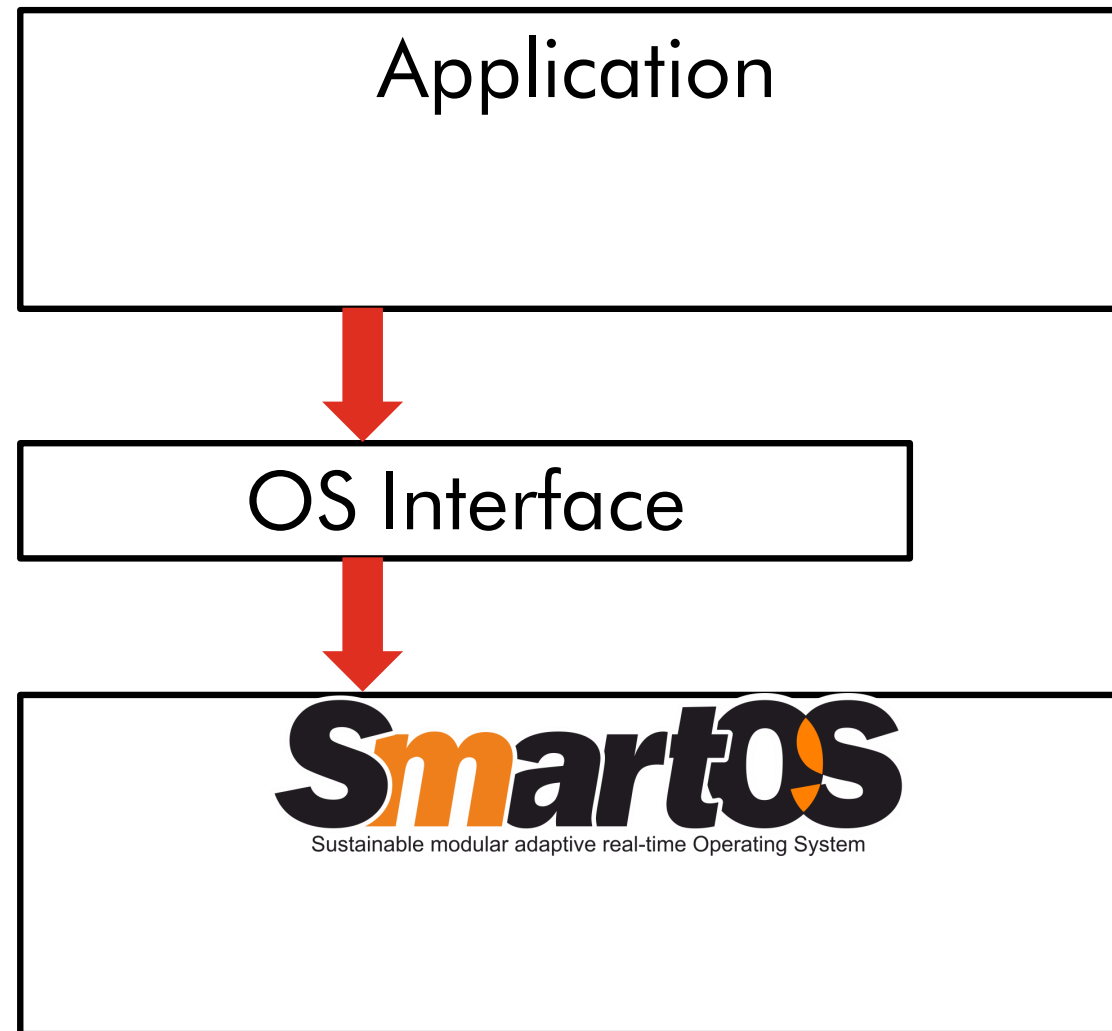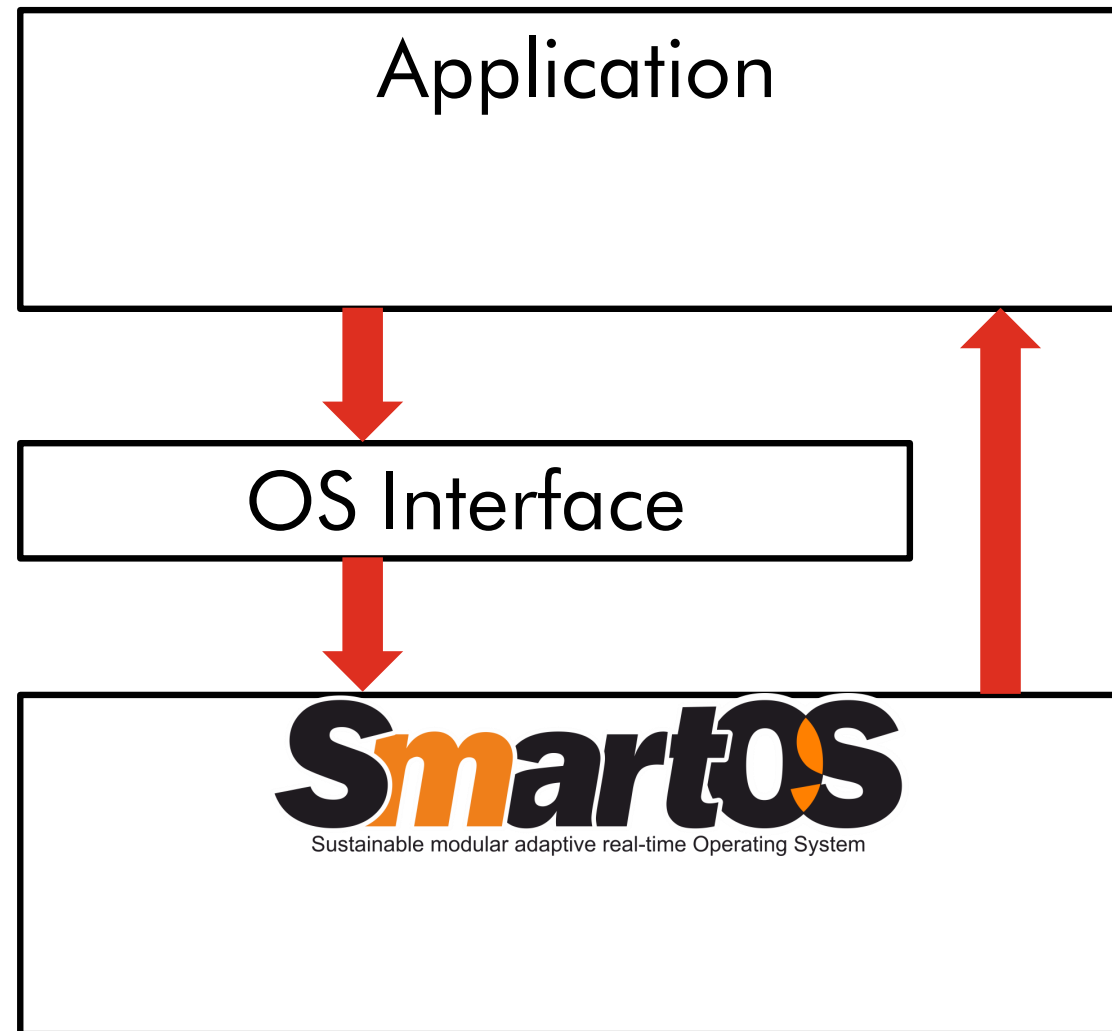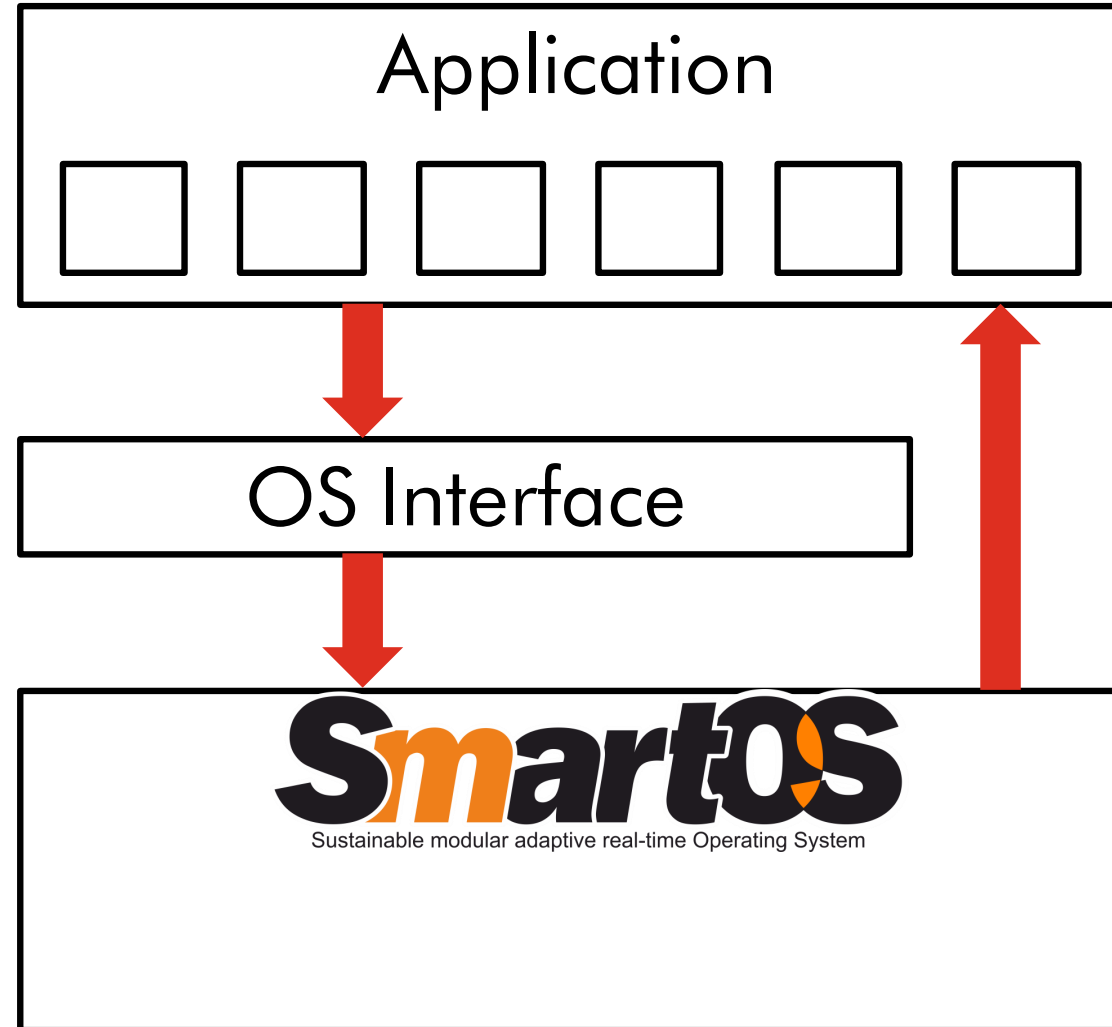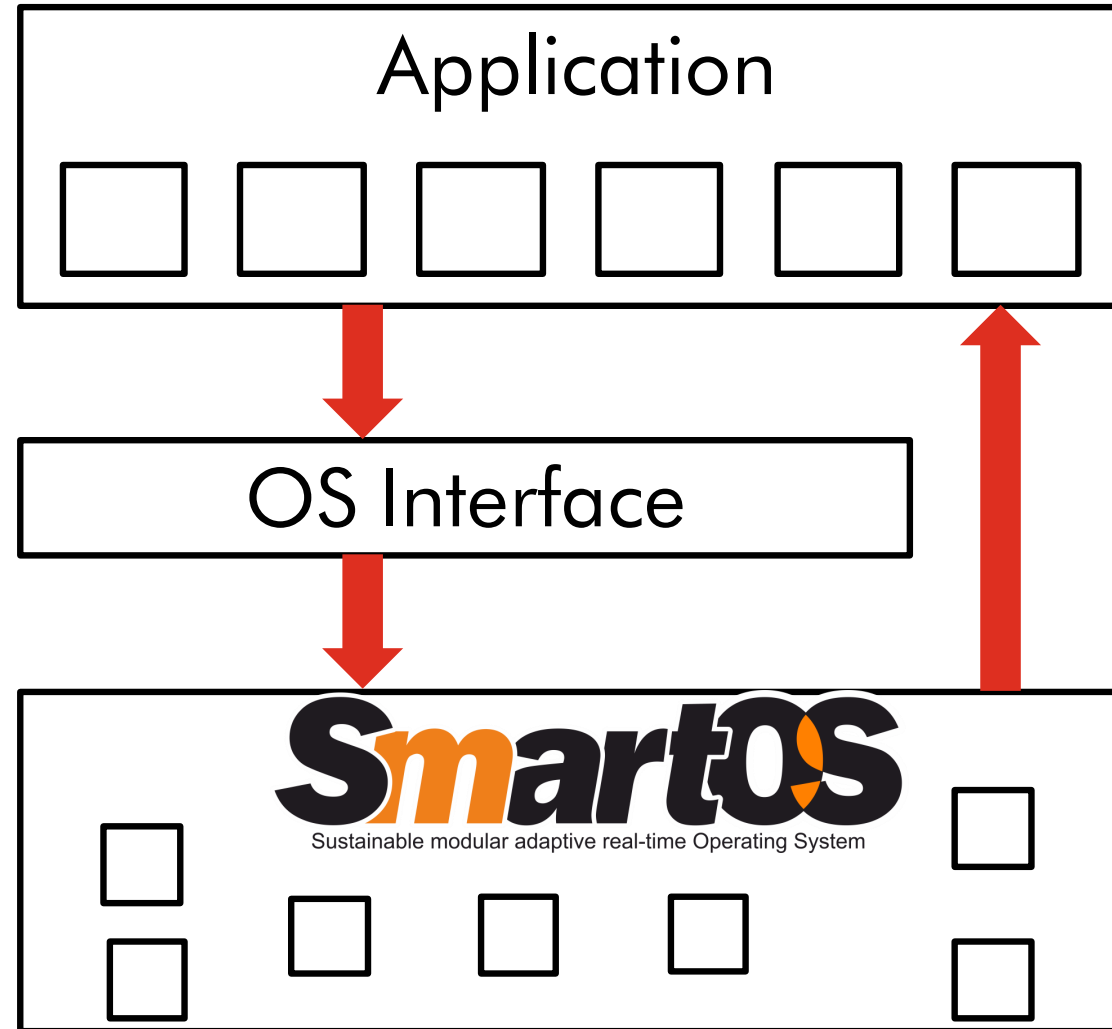
Correctness

Layered Compositional Model
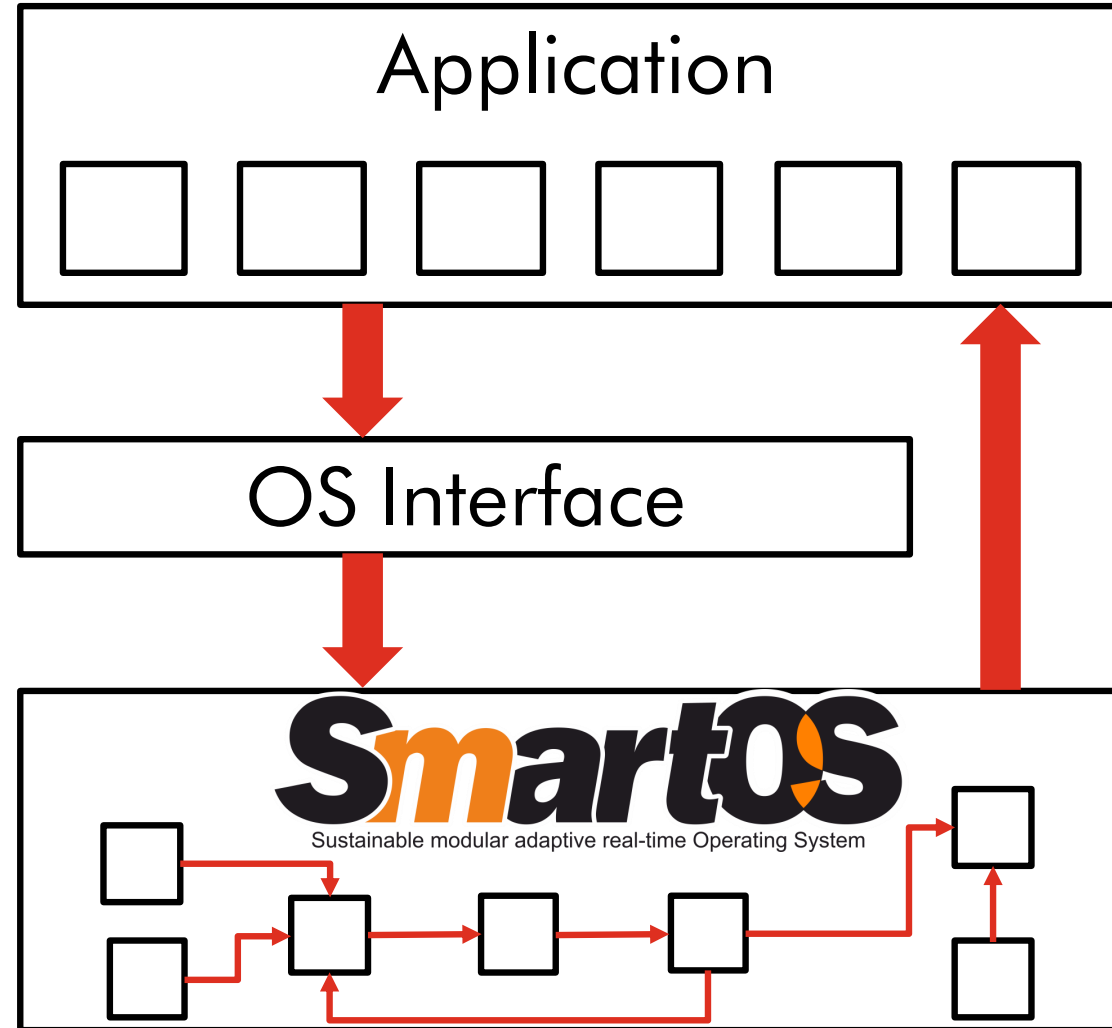
Correctness

Verification Caveats
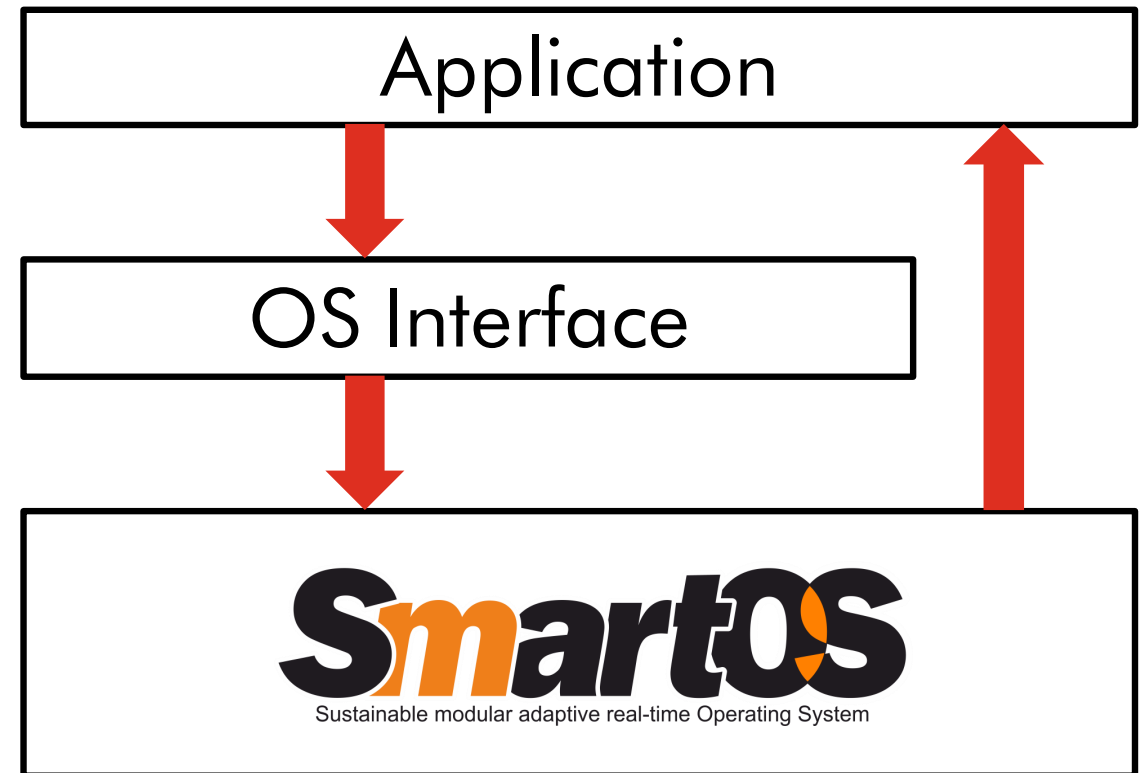
Correctness

Verification Caveats

- Successful Verification
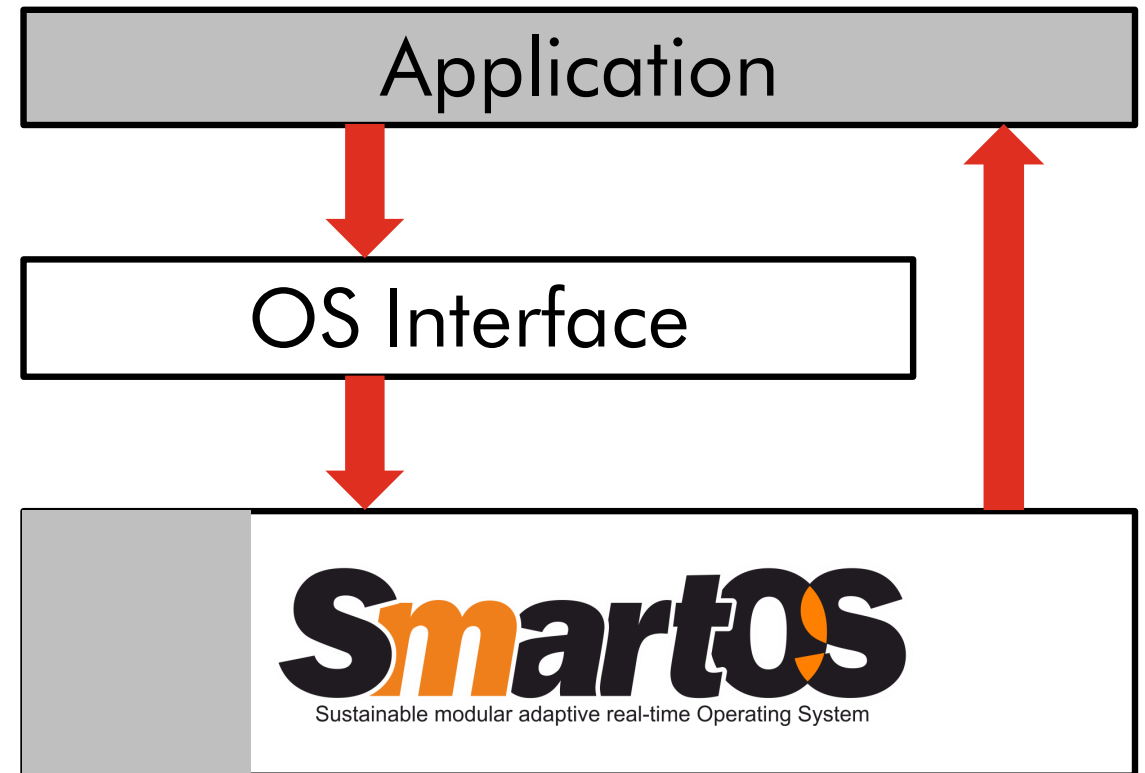
Correctness

Verification Caveats

- Successful Verification
  - OS layer not fully verified

Correctness

Verification Caveats

- Successful Verification
  - OS layer not fully verified

- Unsuccessful Verification

Application

OS Interface
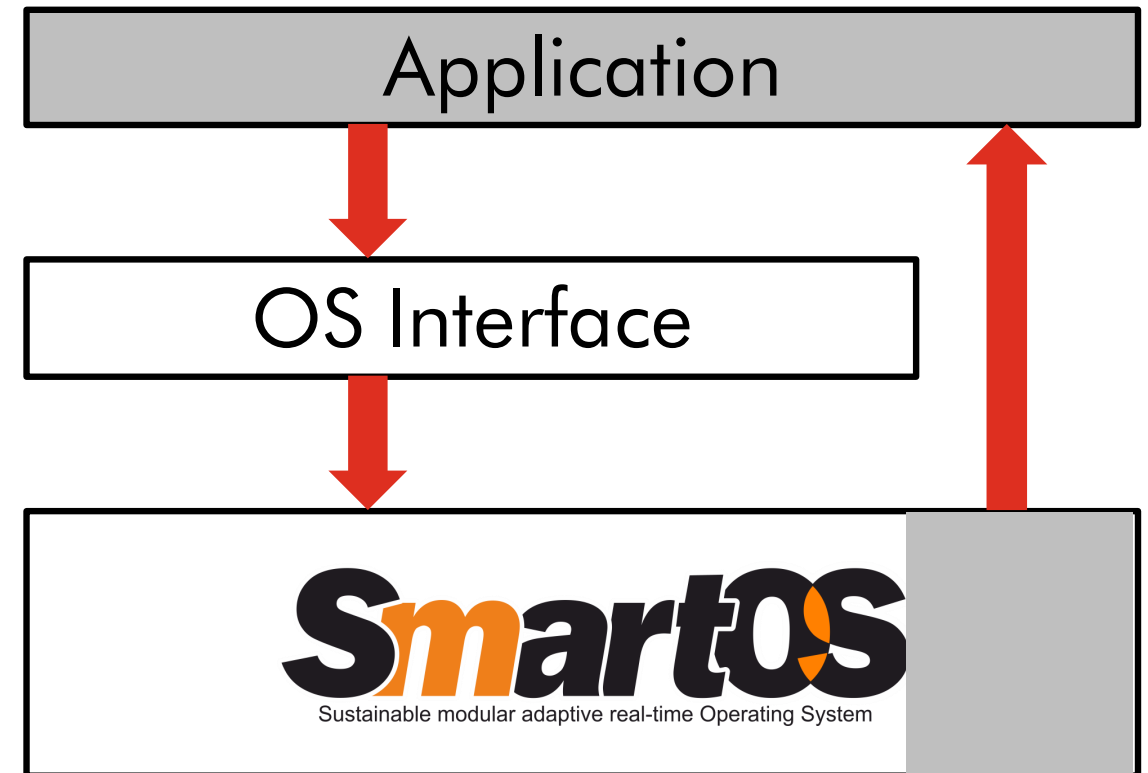
Correctness

Verification Caveats

- **Successful Verification**
  - OS layer not fully verified

- **Unsuccessful Verification**
  - Unknown source of error
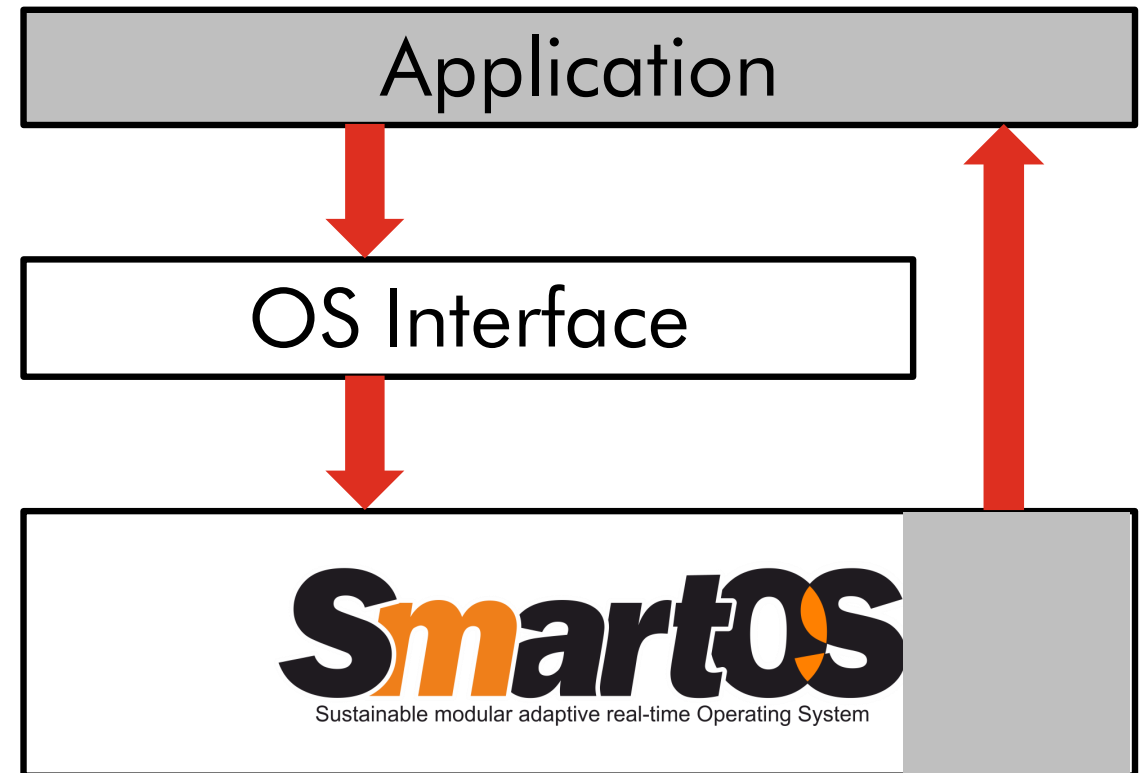
Correctness

Verification Caveats

- Successful Verification
  - OS layer not fully verified

- Unsuccessful Verification
  - Unknown source of error

- Our Approach
  - Abstract application
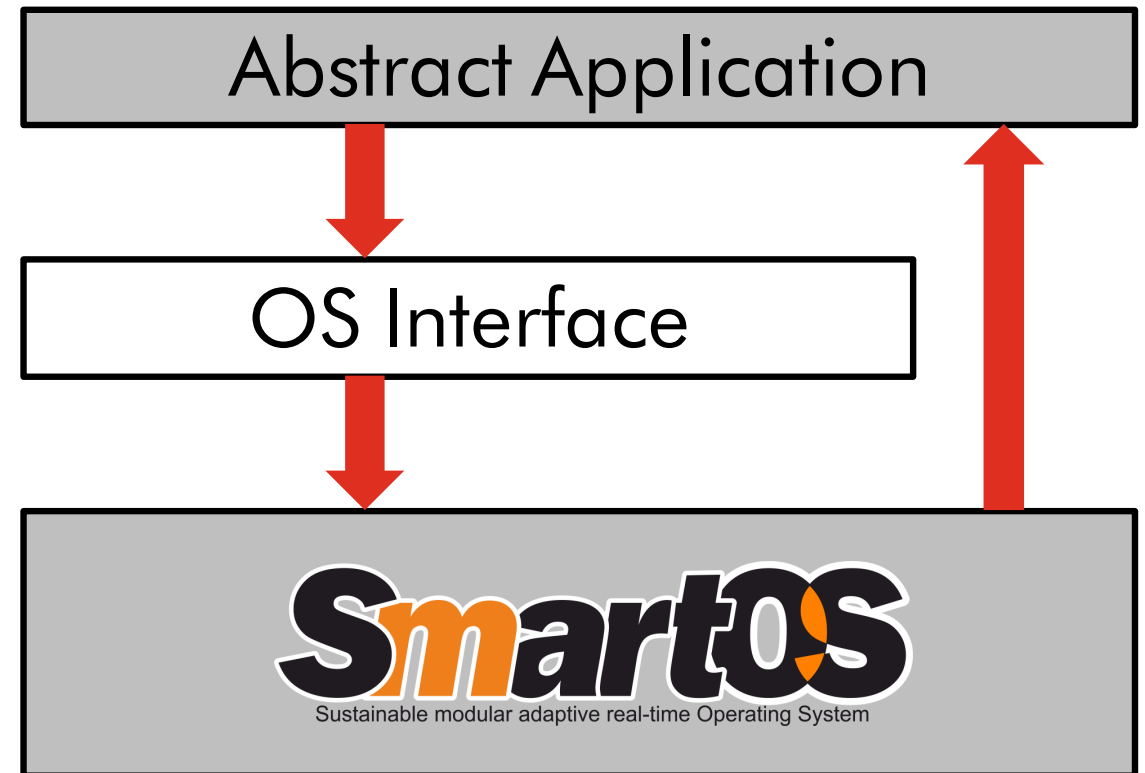
Correctness

Verification Caveats

- Successful Verification
  - OS layer not fully verified

- Unsuccessful Verification
  - Unknown source of error

- Our Approach
  - Abstract application
    - Cheap full OS layer verification
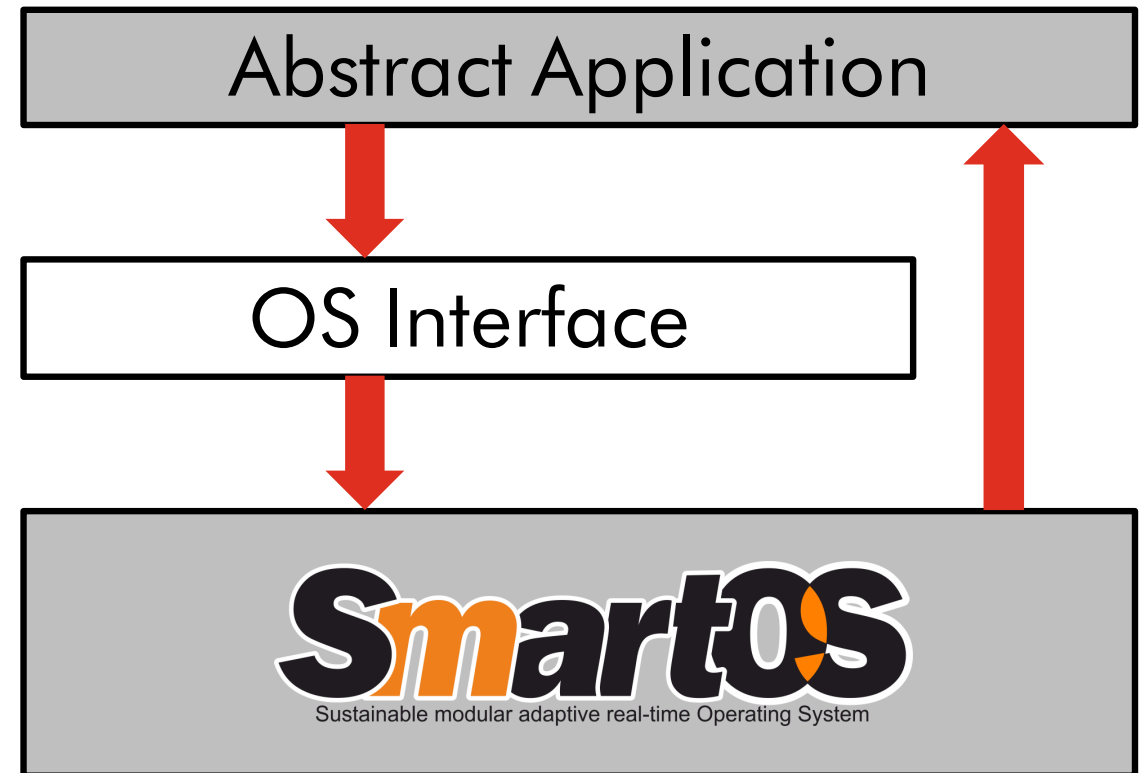
Update Overview
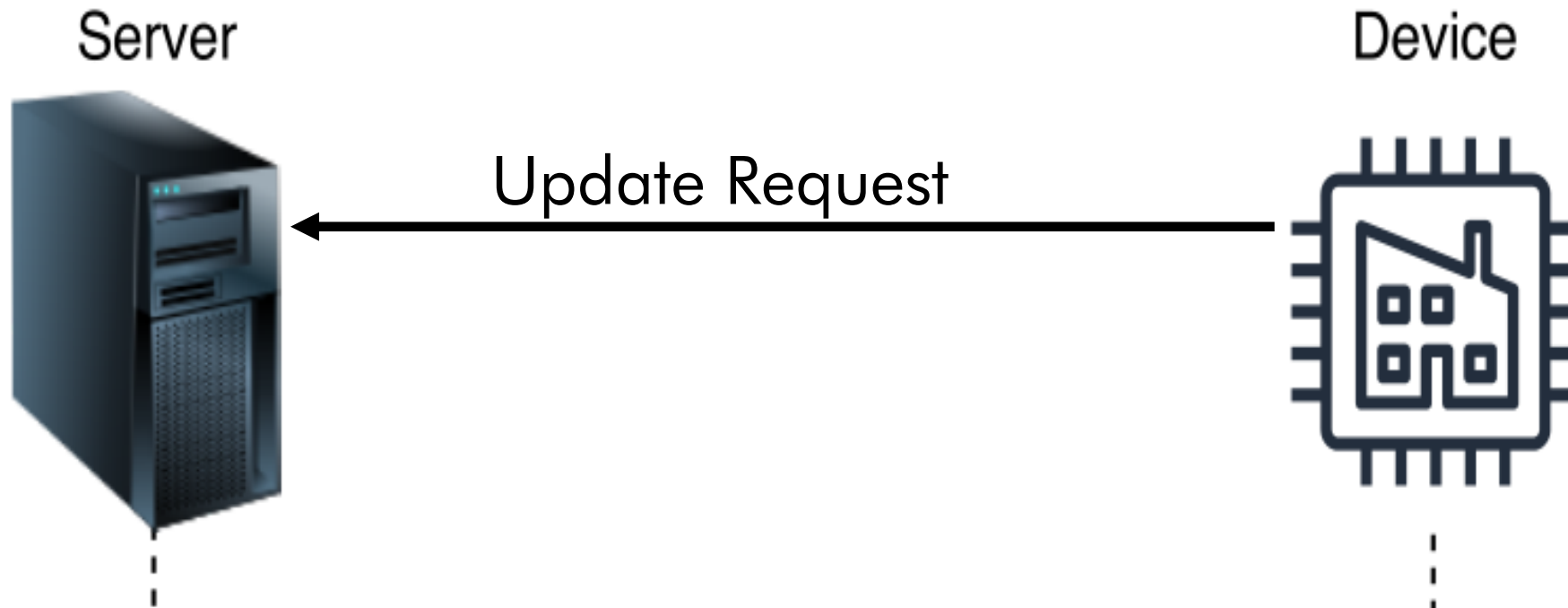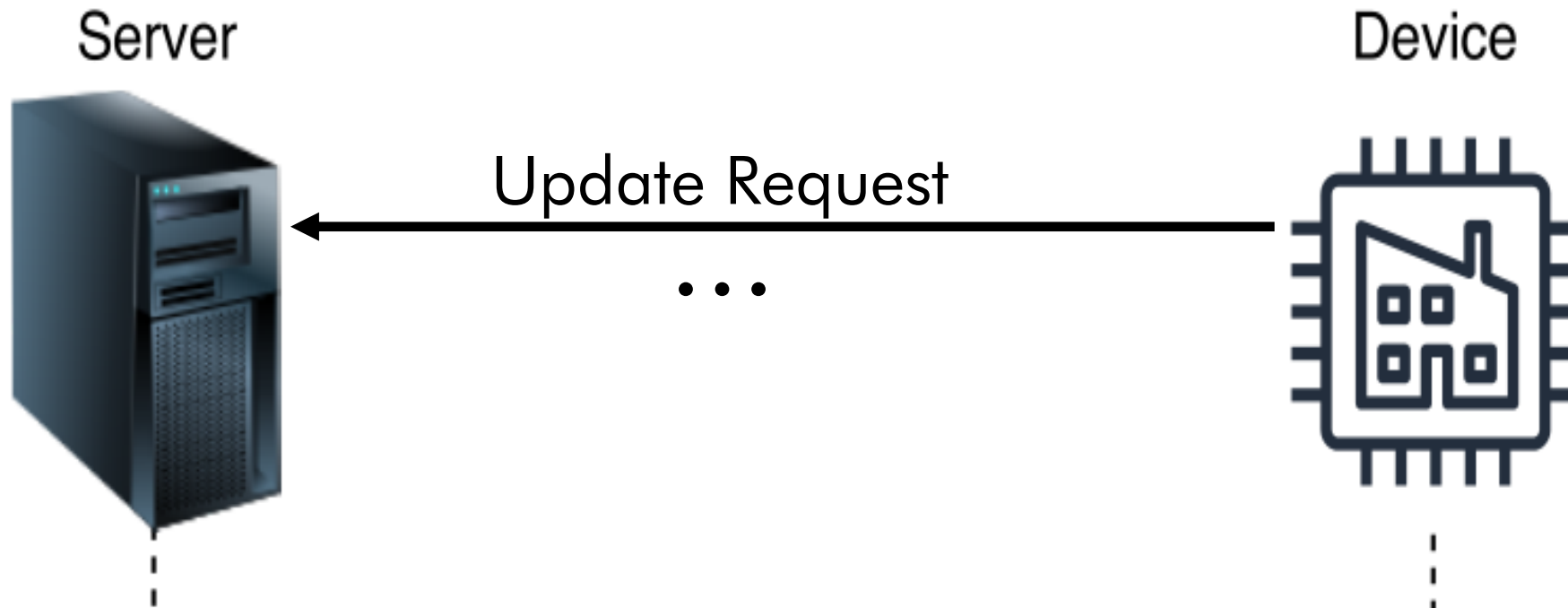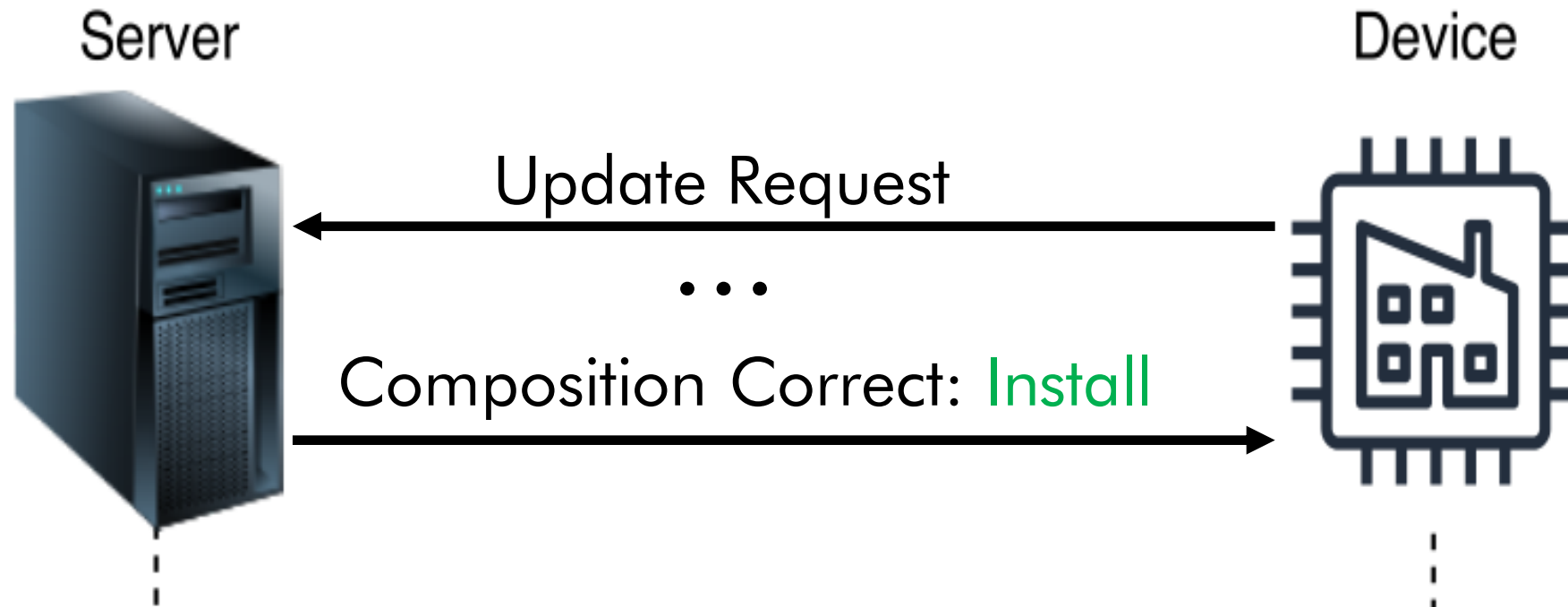
Server   Device

Update Request

Update Overview

Server   Device

Update Request

• • •

Update Overview

Server

Device

Update Request

• • •

Composition Correct: Install

Update Overview

Towards Dynamic Composition of Dependable Embedded Systems
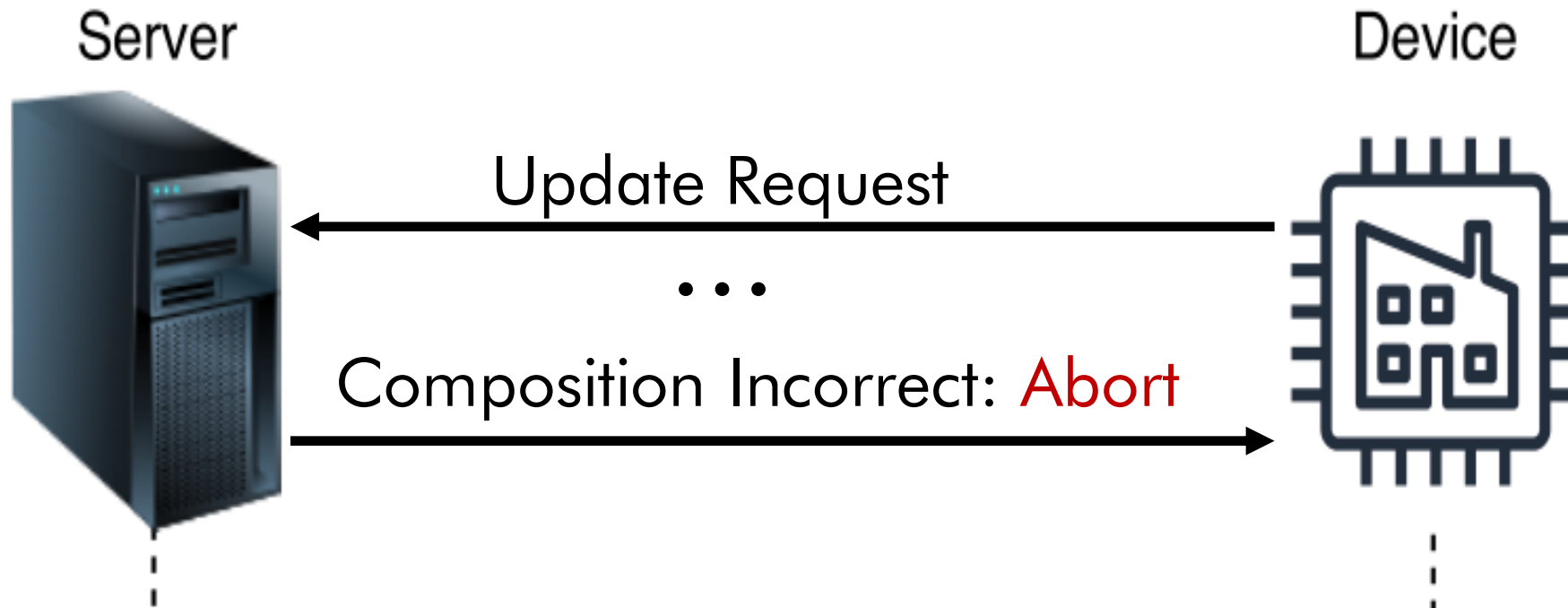
Potential Improvements & Future Work

- Partial Updates: Technical Features

- Formal Verification: Research

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects

- Formal Verification: Research

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects
  - Execution State Transfer

- Formal Verification: Research

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects
  - Execution State Transfer

- Formal Verification: Research
  - Other Requirements

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects
  - Execution State Transfer

- Formal Verification: Research
  - Other Requirements
  - Code-Model Mapping Correctness

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects
  - Execution State Transfer

- Formal Verification: Research
  - Other Requirements
  - Code-Model Mapping Correctness
  - Automatic Code Generation

Potential Improvements & Future Work

- Partial Updates: Technical Features
  - Security Aspects
  - Execution State Transfer

- Formal Verification: Research
  - Other Requirements
  - Code-Model Mapping Correctness
  - Automatic Code Generation
  - State Space Explosion

Summary

- Work divided in two major parts
  1. How to support partial updates

Summary

- Work divided in two major parts
  1. How to support partial updates

Module

Transmission

Software

Summary

- Work divided in two major parts
  1. How to support partial updates

Module

Transmission → Software

Server   Device

SmartOS
Sustainable modular adaptive real-time Operating System

Leandro Batista Ribeiro   Towards Dynamic Composition of Dependable Embedded Systems   PhD Seminar – April/2022

Summary

- Work divided in two major parts
  1. How to support partial updates

Module

Transmission → **Software**

HW Support

Summary

- Work divided in two major parts

    1. How to support partial updates

    2. How to guarantee correctness of software compositions

Summary

- Work divided in two major parts
    1. How to support partial updates
    2. How to guarantee correctness of software compositions

Summary

- Work divided in two major parts
  1. How to support partial updates
  2. How to guarantee correctness of software compositions

Application

OS Interface

SmartOS
Sustainable modular adaptive real-time Operating System

Running   Ready   Waiting

Summary

- Work divided in two major parts
  1. How to support partial updates
  2. How to guarantee correctness of software compositions

Summary

- Work divided in two major parts
    1. How to support partial updates
    2. How to guarantee correctness of software compositions
        - Current Focus
            - Liveness
            - Real-Time

Summary

- **Work divided in two major parts**
  1. How to support partial updates
  2. How to guarantee correctness of software compositions
     - Current Focus
       - Liveness
       - Real-Time

- **Future Work**
  - Partial Updates: Technical Improvements

Summary

- **Work divided in two major parts**
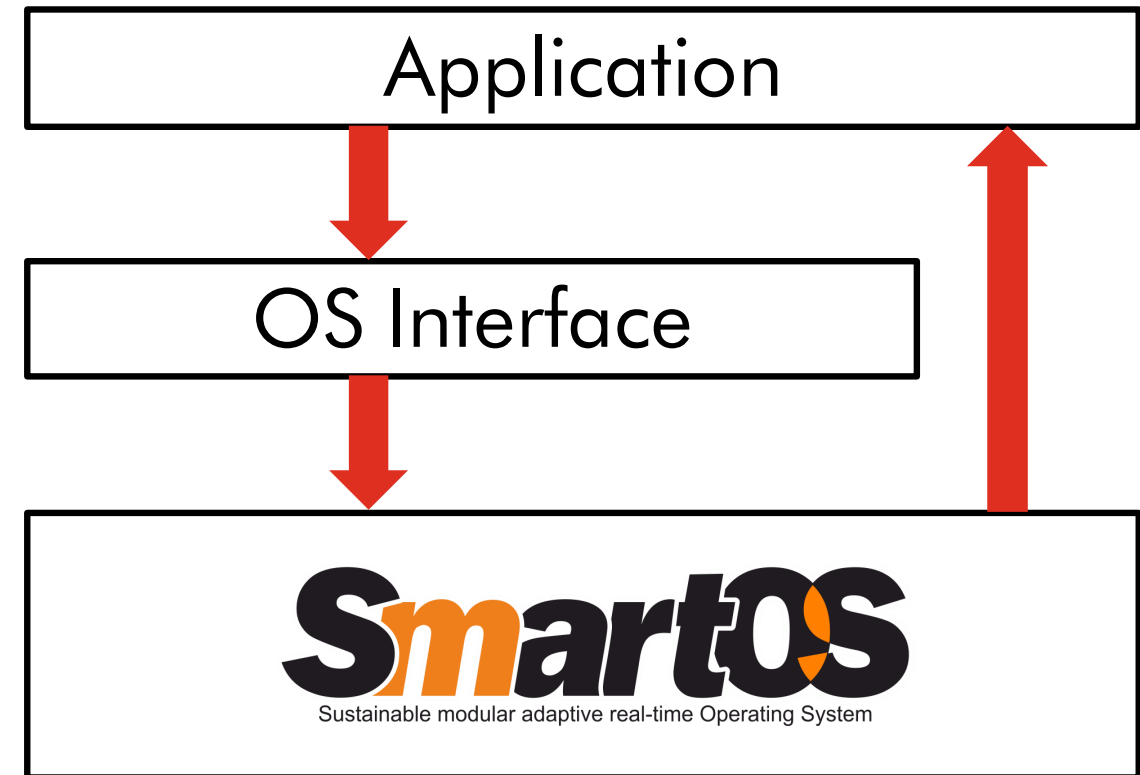 1. How to support partial updates
  2. How to guarantee correctness of software compositions
     - Current Focus
       - Liveness
       - Real-Time

- **Future Work**
  - Partial Updates: Technical Improvements
  - Correctness: Research Challenges

Thank you!

Leandro Batista Ribeiro
lbatistaribeiro@tugraz.at

Institute of Technical Informatics
Embedded Automotive Systems Group
Graz University of Technology

References

[EWSN'20] Leandro Batista Ribeiro, Fabian Schlager, and Marcel Baunach. "Towards Automatic SW Integration in Dependable Embedded Systems." *International Conference on Embedded Wireless Systems and Networks. 2020*

[CODES'21] Malenko, Maja, Leandro Batista Ribeiro, and Marcel Baunach. "Improving security and maintainability in modular embedded systems with hardware support: work-in-progress." International *Conference on Hardware/Software Codesign and System Synthesis*. 2021.

[DP] https://spin.atomicobject.com/2012/10/31/elixir-erlang-and-the-dining-philosophers/