Recent Works on 61499 at Saarland University

Prof. Dr.-Ing. Georg Frey

Chair of Automation

georg.frey@aut.uni-saarland.de

UNIVERSITÄT DES SAARLANDES

AUT
Lehrstuhl für Automatisierungstechnik

Overview

- Implementation on .NET

- Automatic Deployment

Expected advantages from Application of IEC 61499

- Abstraction from
  - Controller hardware
  - Automation infrastructure
  - Communication
  - Process interfacing

- Our target
  - **Less effort for implementation (increased usability)**

- Realization
  - Implementation of IEC 61499 runtime as middleware

- Prerequisite
  - **Clarification of ambiguities in standard's text**

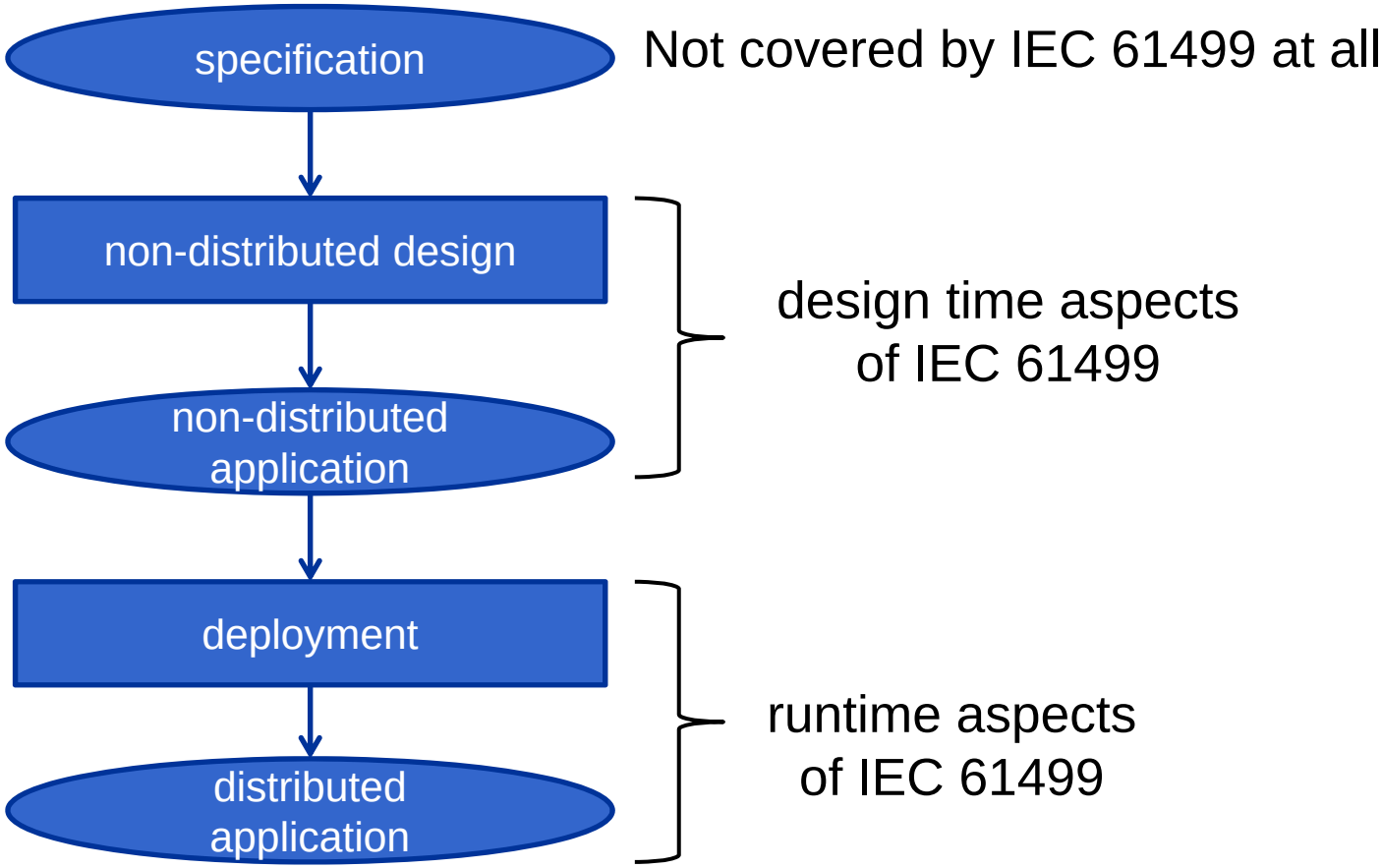
Identified ambiguities in the standard‘s text

- Ambiguities identified in standard
  - **1. Development process**
  - **2. Synchronization of concurrency**
  - **3. Event and data transport**
  - 4. Invocation of FBs
  - 5. Sub-applications
  - 6. Composite FBs
  - **7. Consumation of events**
  - 8. Publication of events

→ **Eight Principles for use of IEC 61499**

→ **Two are discussed in the following**

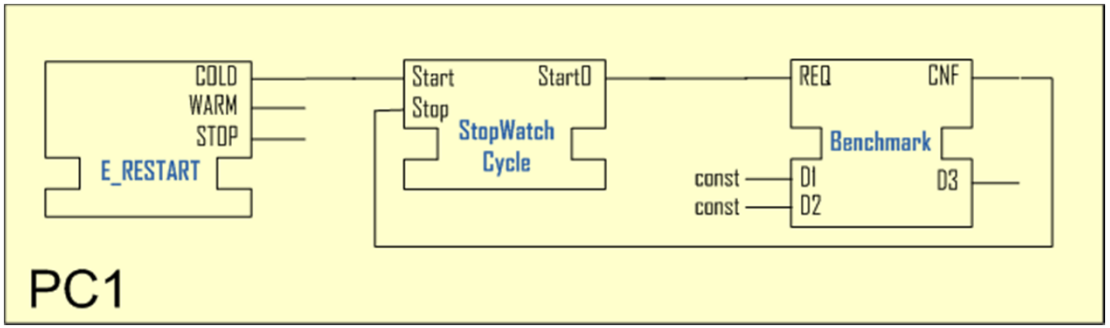
Development Process covered by IEC 61499



→ **Principle 1: Application design is to be done independently of the actual implementation!**

Sample application of the development process

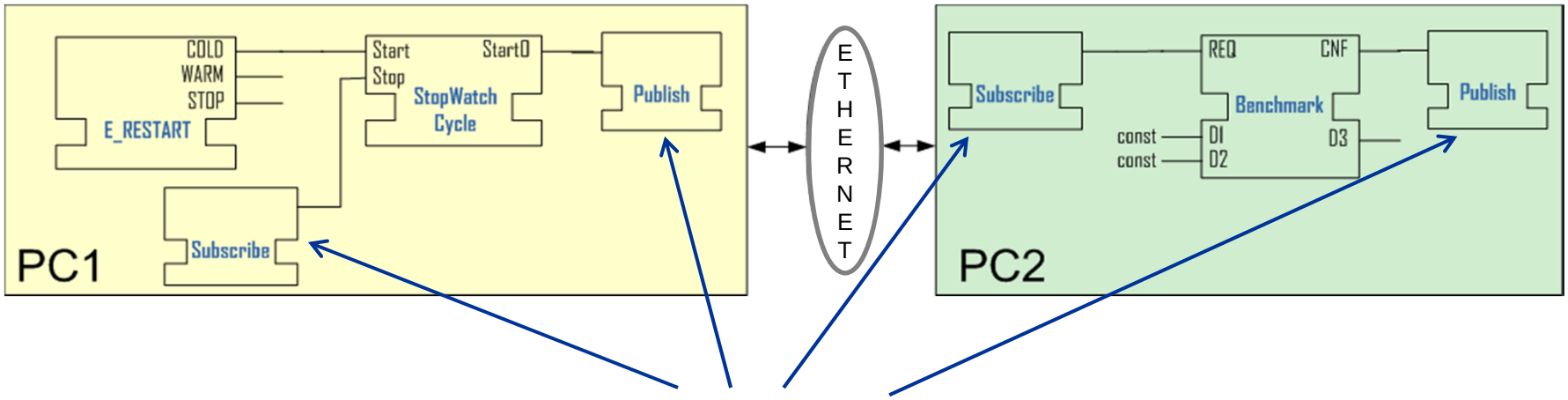
- Non-distributed Application



Definition of semantics!

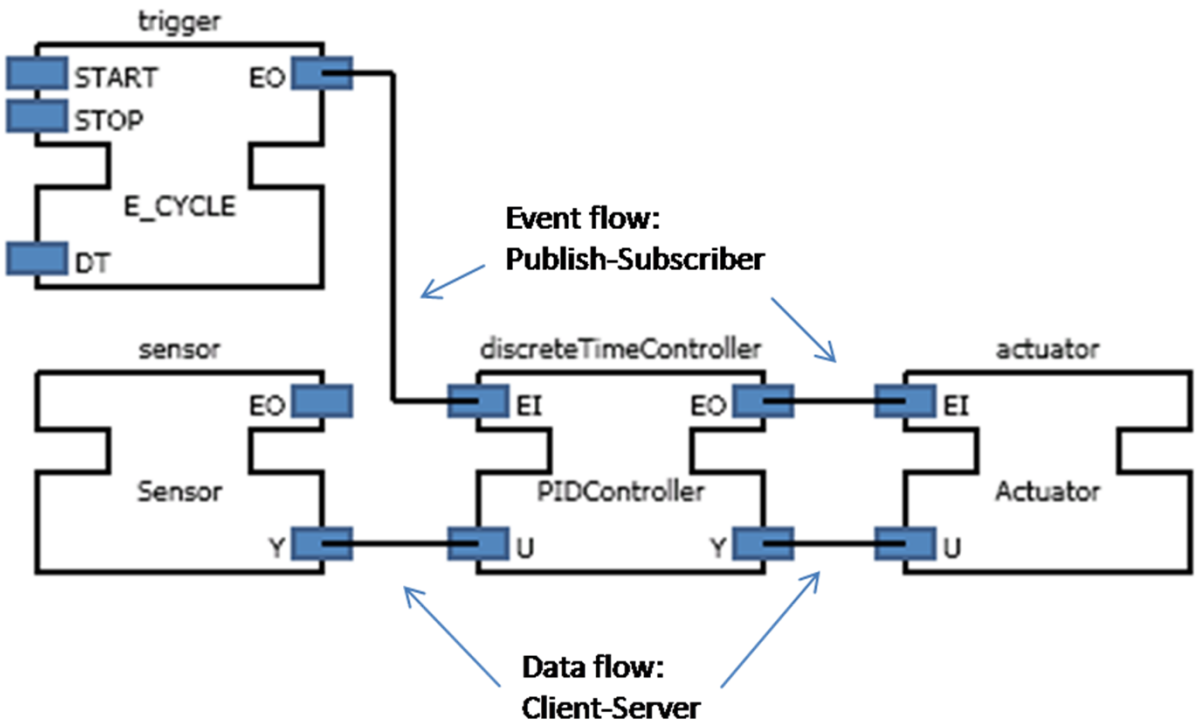
- Distributed Application

Influence on runtime behavior!



Automatically inserted!

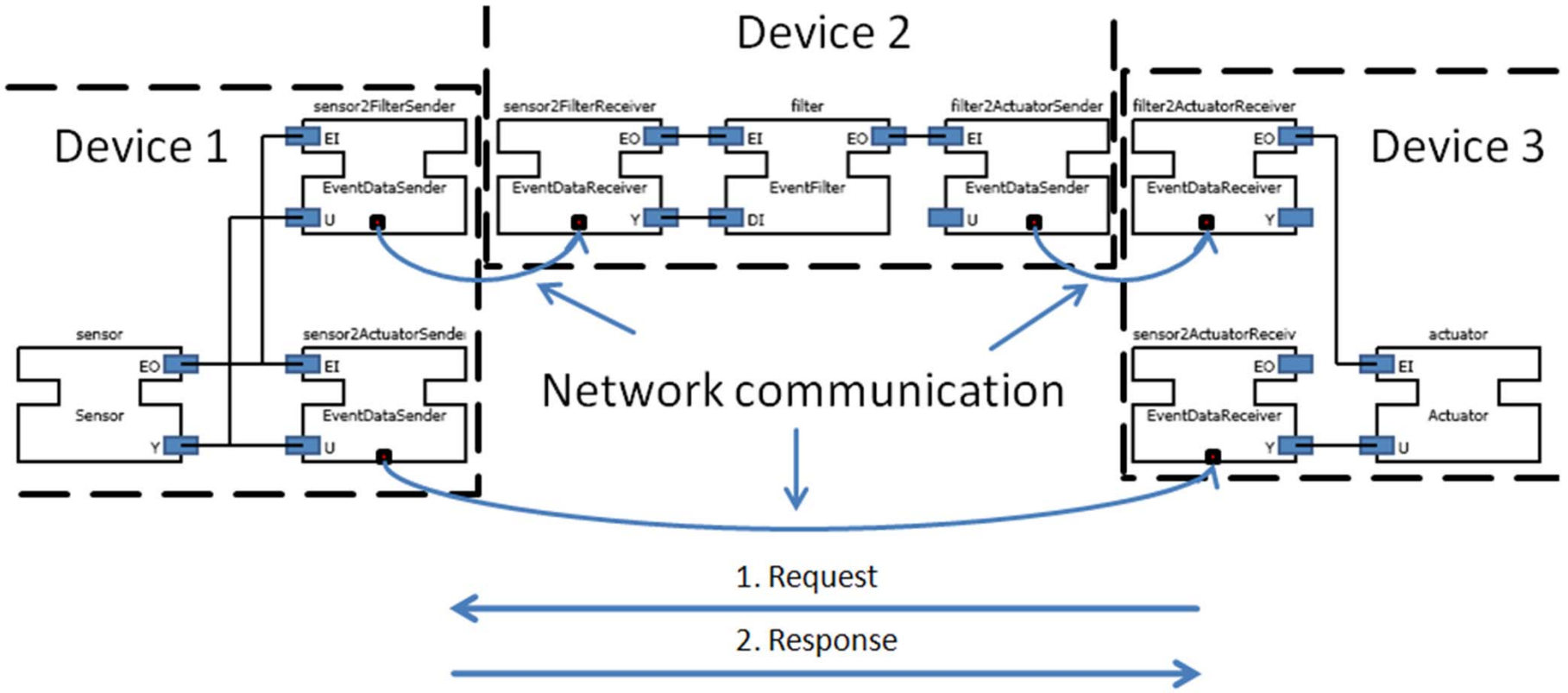
Event and data transport



- Event transport
  - Publish-Subscriber mechanism
  - Multiple sources and multiple targets for events possible

- Data transport
  - Client-Server mechanism
  - One source and multiple targets for data possible

Event and Data Transport in Distributed Applications

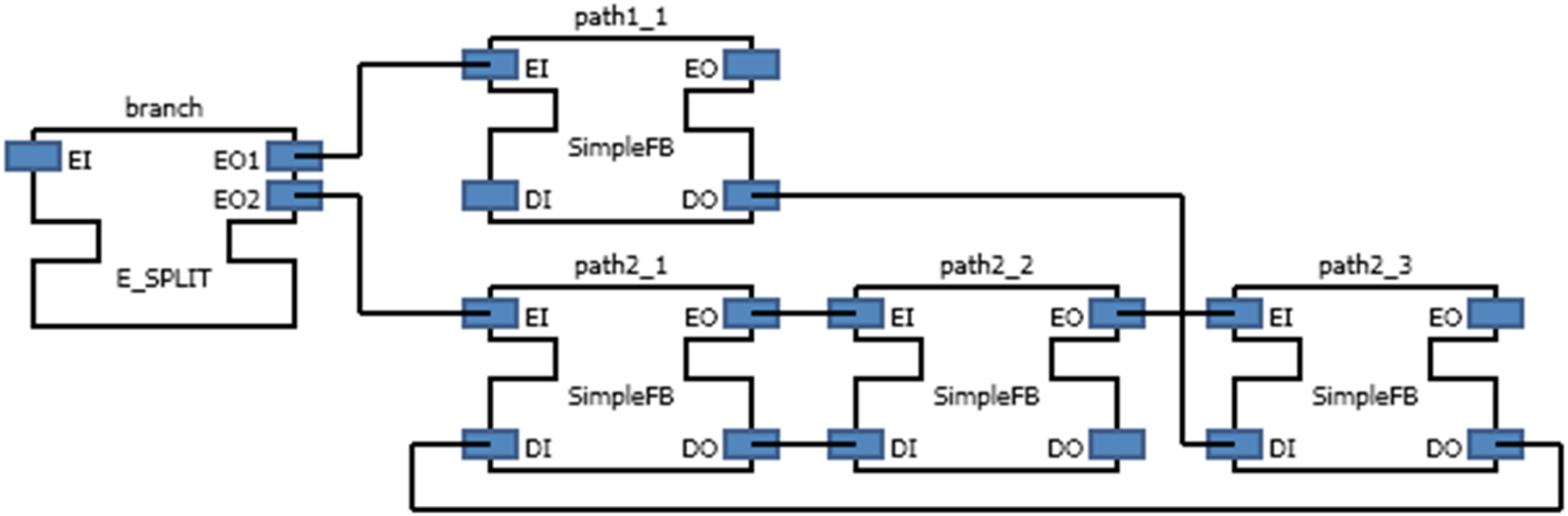


→**Principle 3:** Event flow is based on the Publish-Subscriber model while data flow uses a Client-Server mechanism. **The IEC 61499 runtime environment is in charge of delivering data and events on time!**

NO EVENTS GET LOST!

Execution Order of Function Blocks

- Function block networks define a partial order of execution

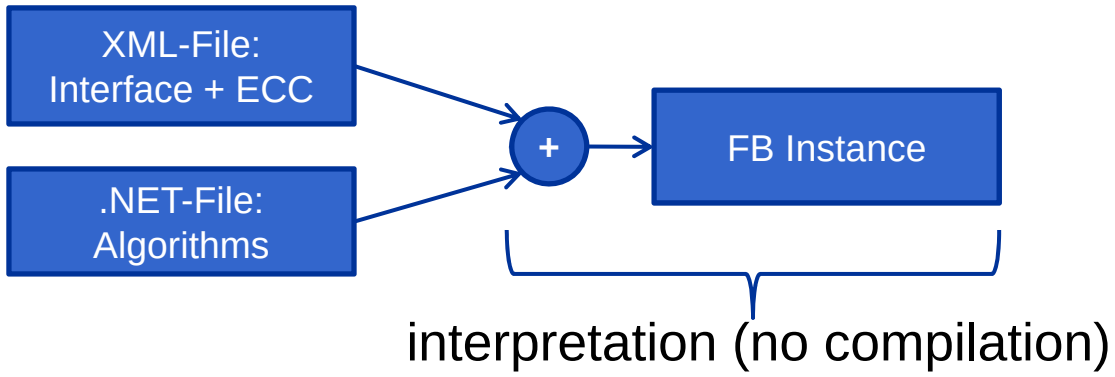


- Some interpretations of IEC 61499 try to convert the partial order to a fully specified order
  - ➢ Only works on resource level (contradiction to **Principle 1)**

→ **Principle 2: FBNs specify a partial order of FB execution. Synchronization between FBs has to be explicitly specified in the design!**

61499.NET

- IEC 61499 runtime implementation based on .NET
- Features of .NET implementation
  - Free choice of programming language: C#, Visual Basic, C++,…
  - .NET software library
  - Visual Studio as development environment
- Framework specifics
  - Clear distinction between definition of interface and algorithms
  - Automatic insertion of communication SIFBs
  - No compilation of interface and ECC of FBs



Deployment Formulation

- Finding the optimal deployment:
  - ➢ The problem is divided into two parts

Master problem

- One where the constraints are static
  - ❖ Residence : need of specific hardware or software facility
  - ❖ Co-residence: close dependence of certain artifacts so that those are to be on the same hardware
  - ❖ Exclusion: redundant elements created for the sake of fault-tolerance should be on different hardware (also need of specific hardware e.g. timer)
  - ❖ Utilization: for the sake of scheduling utilization should not exceed a pre-defined limit
  - ❖ Memory capacity
  - ❖ Network use

Sub-problem

- One where the constraints are dependent upon the dynamics
  - ❖ Time constraints (WCET)
  - ❖ Schedulability: the deployed tasks should have to be schedulable

Deployment Solution

- The master problem can be solved using simple constraint solving algorithm (i.e., backtracking search)
- Multiple solutions of the master problem are needed since they need to be compared with respect to its suitability to sub-problems
- For solving sub-problems response time analysis or schedulability analysis is needed.
- Master problem and sub-problems are inter-related and the inter-dependence can be learned through explanation-based learning
- Prototypical Implementation works

Conclusions and Outlook

- Utilization of IEC 61499 as middleware
  - Abstraction
  - Simplification of design and implementation
- Interpretation of standard's text focused on usability
  - Clear distinction between design and deployment
  - Clarification of ambiguities (with focus on usability)
- Prototypical Implementation in .NET
  - Free choice of programming language
  - Visual Studio as development environment
  - Can be run on Windows and Linux (with project Mono)
  - Not necessarily in combination with 61131!
- Automatic Deployment is possible

Thank you!

Any questions?