IEC 61499: A Standardized Architecture for Adding Value in Industrial Automation

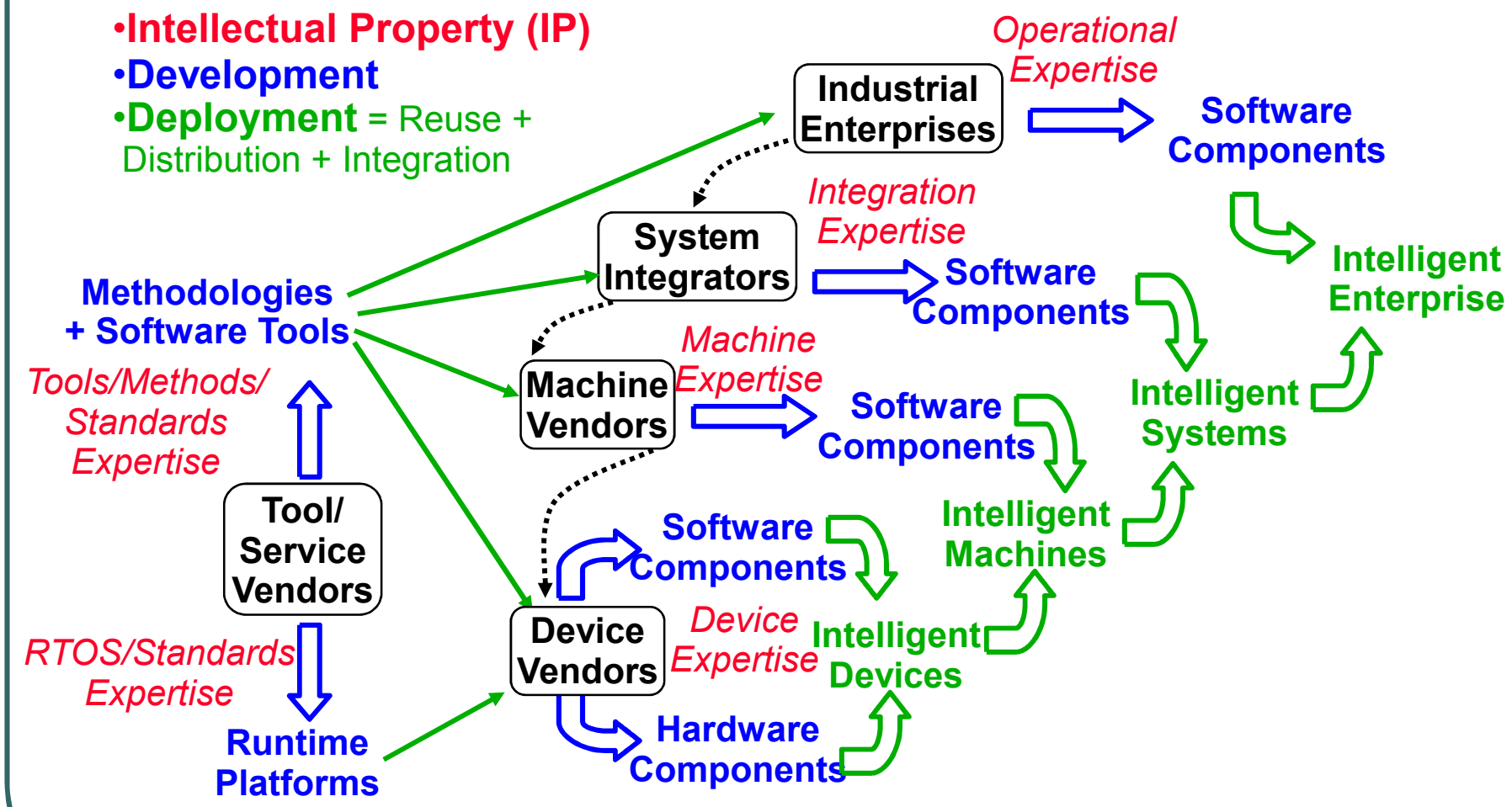
KITARA Seminar
HTC High Tech Center
September 5, 2007

James H. Christensen
Holobloc, Inc.
jhchristensen@holobloc.com

Adding Value with IEC 61499

- **Background**
- Requirements
- Architecture
- Design Patterns & Frameworks
- Software Tools
- Runtime Platforms

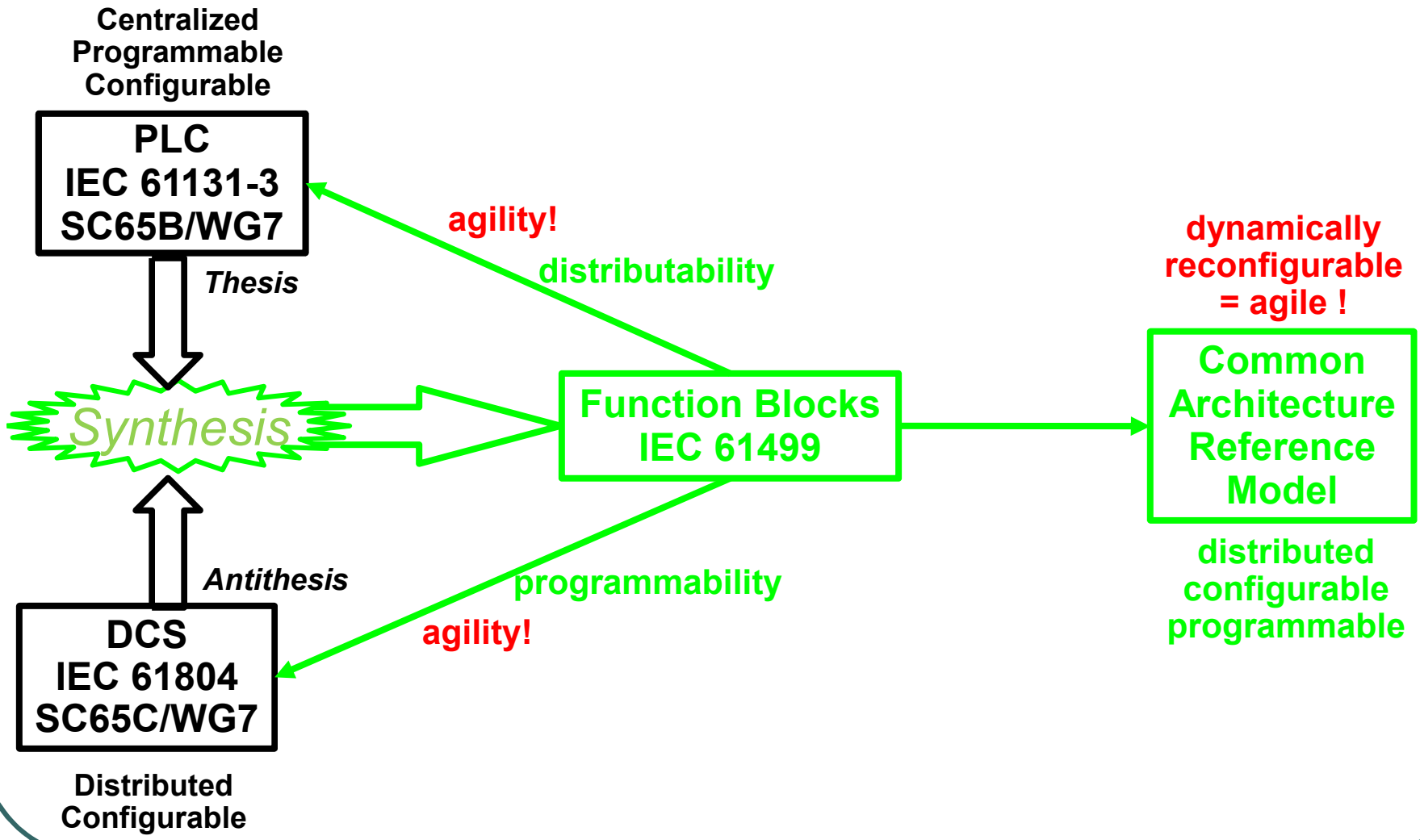
The Industrial Automation Value-Add Chain



What is IEC 61499?

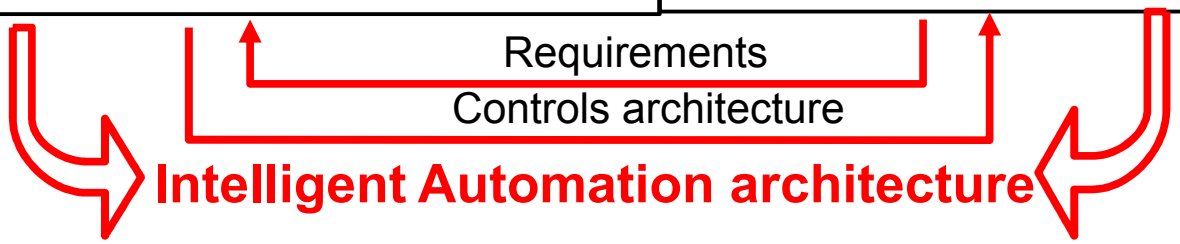
- An IEC (International Electrotechnical Commission) Standard for the use of **function blocks** in **distributed industrial-process measurement and control systems**
- **Part 1, Architecture**
  - IEC Standard, January 2005
- **Part 2, Software Tool Requirements**
  - IEC Standard, January 2005
- **Part 4, Rules for Compliance Profiles**
  - IEC Standard, May 2005
- **Part 3, Tutorial Information**
  - Withdrawn (obsolete), 2007

Function Blocks:
The Architectural Dialectic



Architectural Co-Evolution

IEC 61499	Holonic Manufacturing Systems (HMS)
Parent organization: IEC	**Parent organization**: IMS
Working group: TC65/WG6	**Working group**: HMS Consortium
Goal: Standard model (function blocks) for control encapsulation & distribution	**Goal**: Agile, intelligent manufacturing through holonic (autonomous, cooperative) modules
Started: 10/90	**Feasibility study**: 3/93-6/94
Active development: 3/92	**First phase**: 2/96 - 6/00
Trial period: 2001-03	**Second phase**: 6/00-6/03
Completion: 2005	



Adding Value with IEC 61499

- Background
- **Requirements**
- Architecture
- Design Patterns & Frameworks
- Software Tools
- Runtime Platforms

Agile Manufacturing Requirement: Dynamic Reconfiguration

- **The Vision (Iacocca Institute, 1991)**
  - Production to Order
  - Lot/Batch size ≥ 1 Unit
  - "Information intensive, reprogrammable, reconfigurable, continuously changeable"
- **Physical Reconfiguration**
  - Modular Machines and Workcells
  - Distributed Automation
- **Logical Reconfiguration**
  - Dynamic Reorganization of Control Plans
  - Minimum Human Intervention (zero preferred)
  - Maintain Configuration Control
- **Not just Parameterization**
  - Leads to Large, Complex Software Modules
  - Reduces Distributability, Flexibility, Reliability

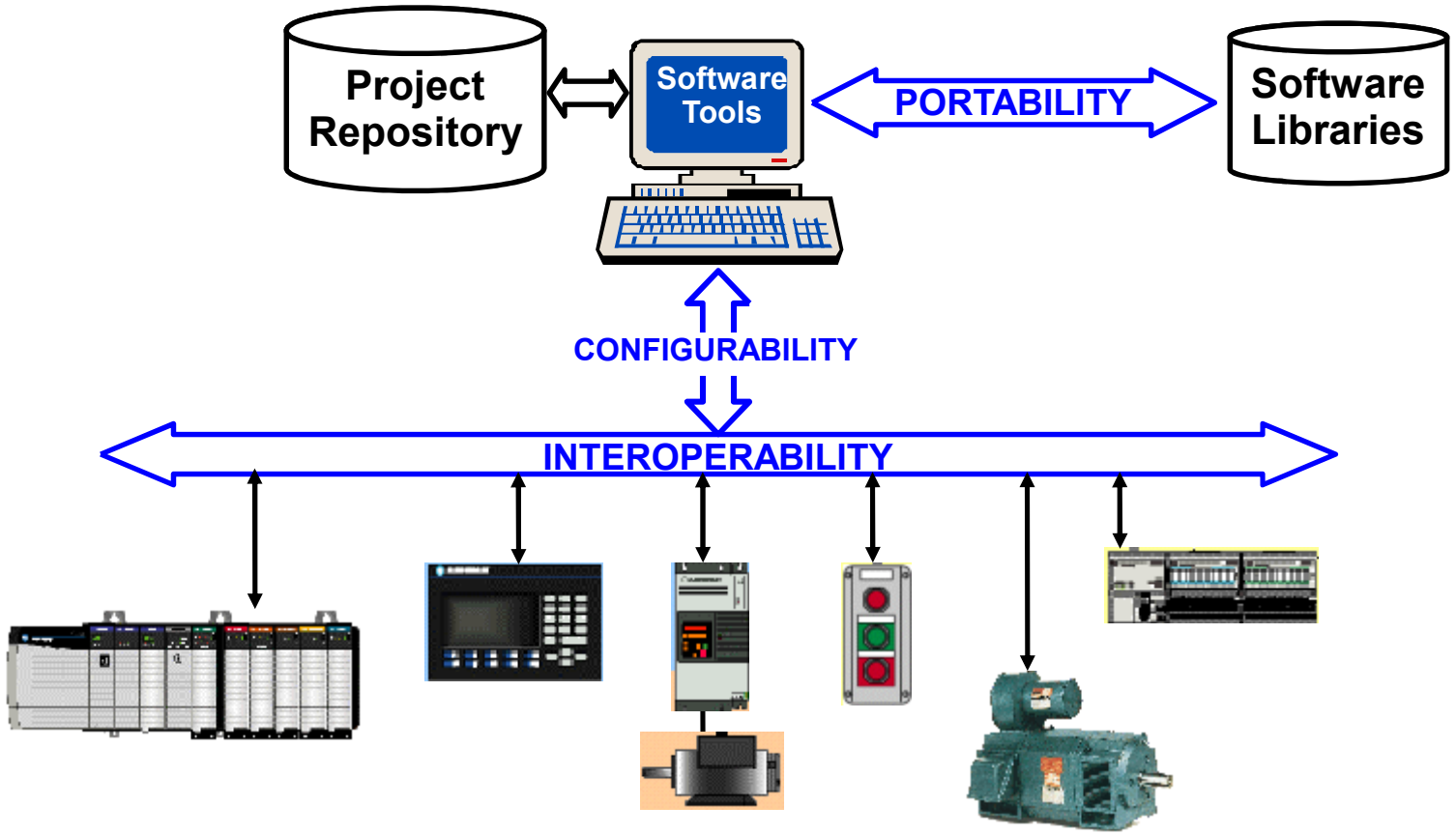
Architectural Requirements

- **Component-Based**
  - Support encapsulation/protection of Intellectual Property (IP)
  - IP Portable across Software Tools and Runtime Platforms
- **Distributed**
  - Map IP modules into distributed devices
  - Integrate IP Modules into distributed applications
- **Functionally Complete**
  - Control/Automation/Diagnostics components
  - Machine/Process Interface components
  - Communication Interface components
  - Human/Machine Interface (HMI) components
  - Software Agent ("Holonic") components
- **Extendable**
  - Encapsulate new types of IP
  - Create new IP through Functional Composition of existing IP modules
- **OPEN!**
  - Multiply the value of IP through widest possible deployment
  - Benefits available to all market players

What is an Open Architecture?

- A *system architecture* whose *functional units* are capable of exhibiting *portability, interoperability* and *configurability:*
  - **portability**: Software tools can accept and correctly interpret *library elements* produced by other software tools.
  - **interoperability**: Devices can operate together to perform the functions specified by one or more *distributed applications*.
  - **configurability**: Devices and their software components can be configured (selected, assigned locations, interconnected and parameterized) by multiple software tools.

Open Distributed Architecture Requirements

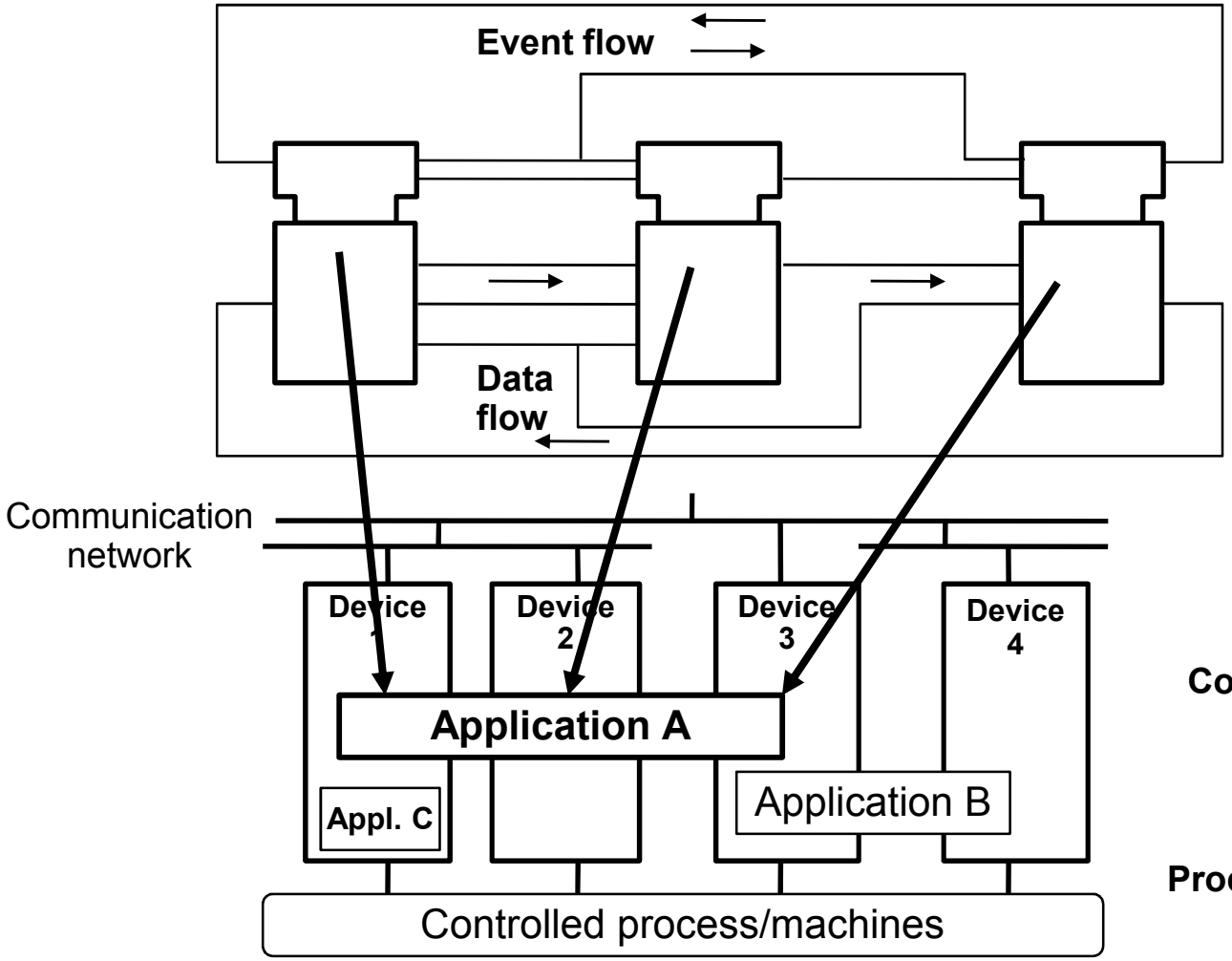


Distributed intelligent devices & machines

Adding Value with IEC 61499

- Background
- Requirements
- **Architecture**
- Design Patterns & Frameworks
- Software Tools
- Runtime Platforms

IEC 61499-1: Distributed System Architecture

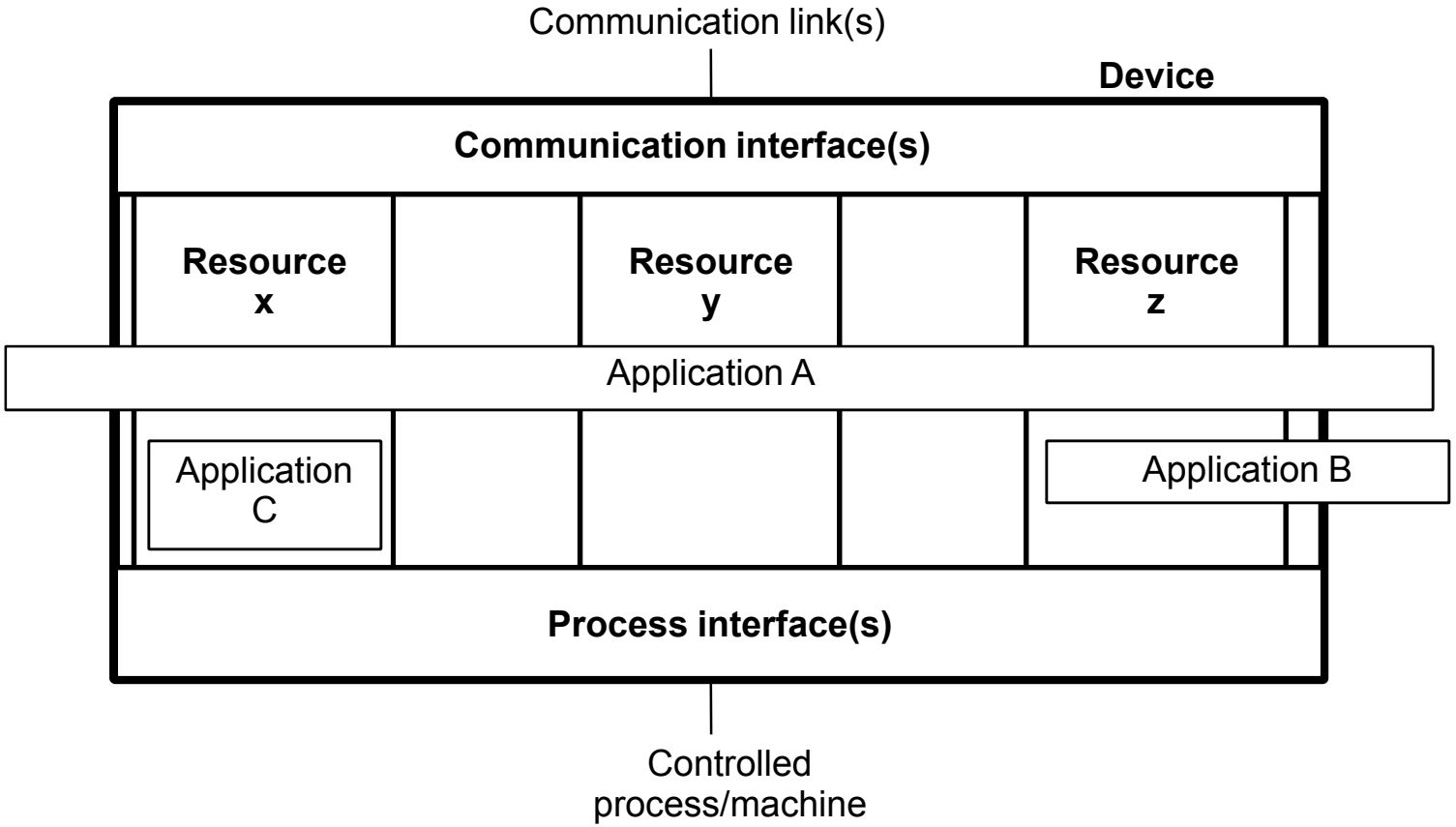


Application = Function Block Network

System = Communication Network + Devices + Process/Machines

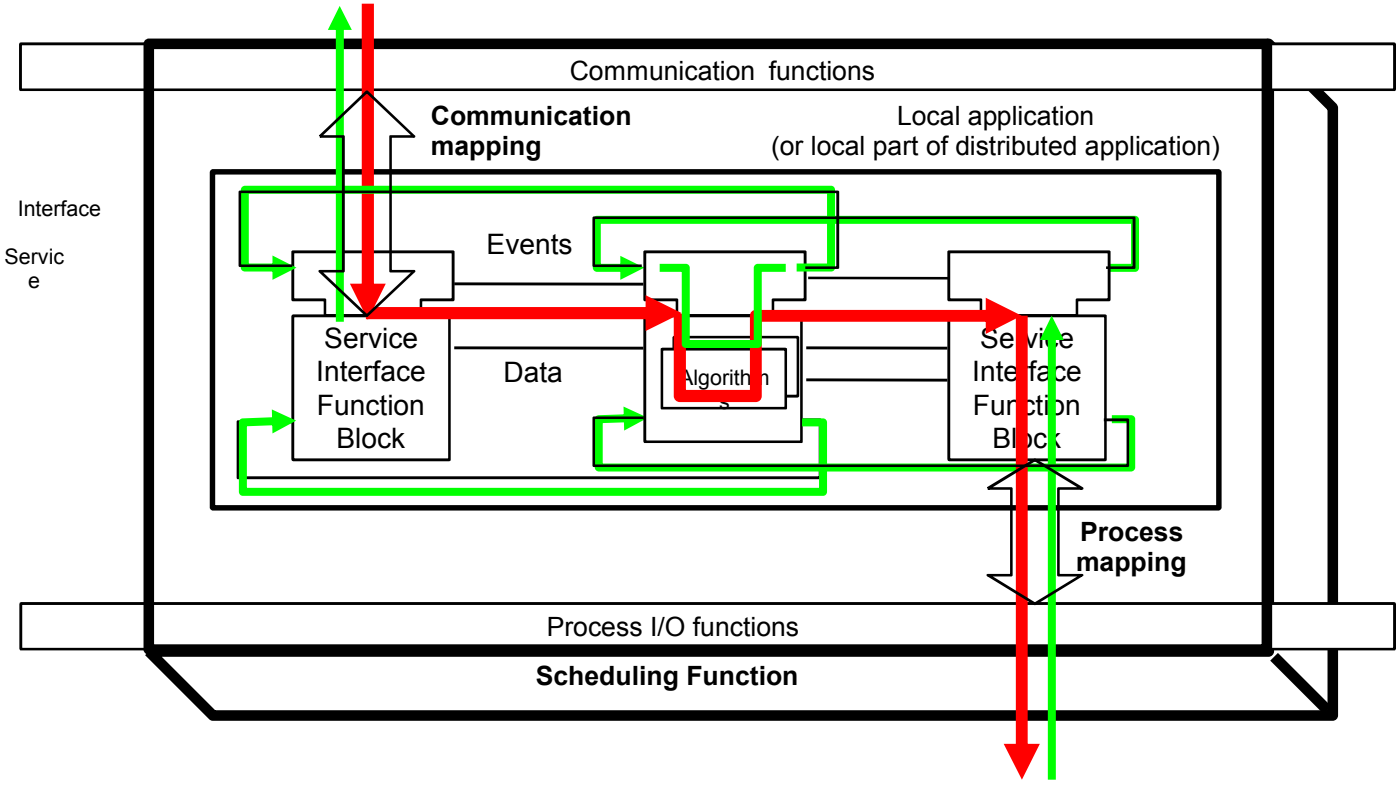
Device Architecture

- Device = Container for Resources
- Device provides Communications & Process Interfaces

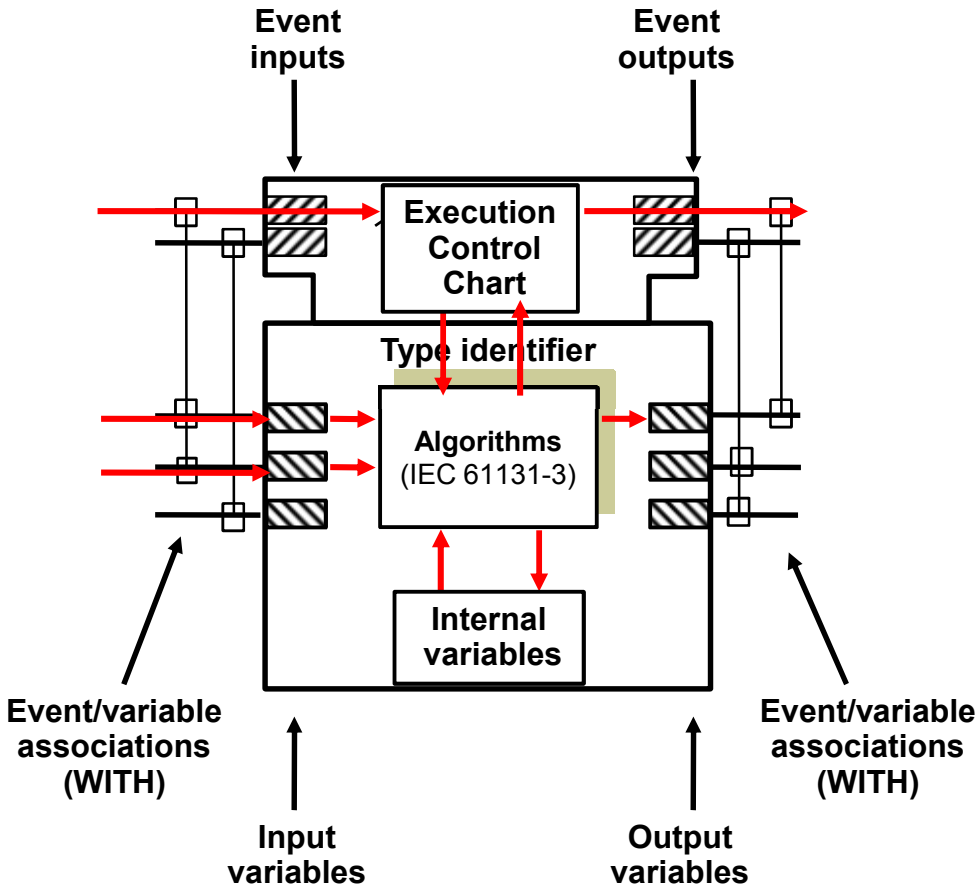


Resource Architecture

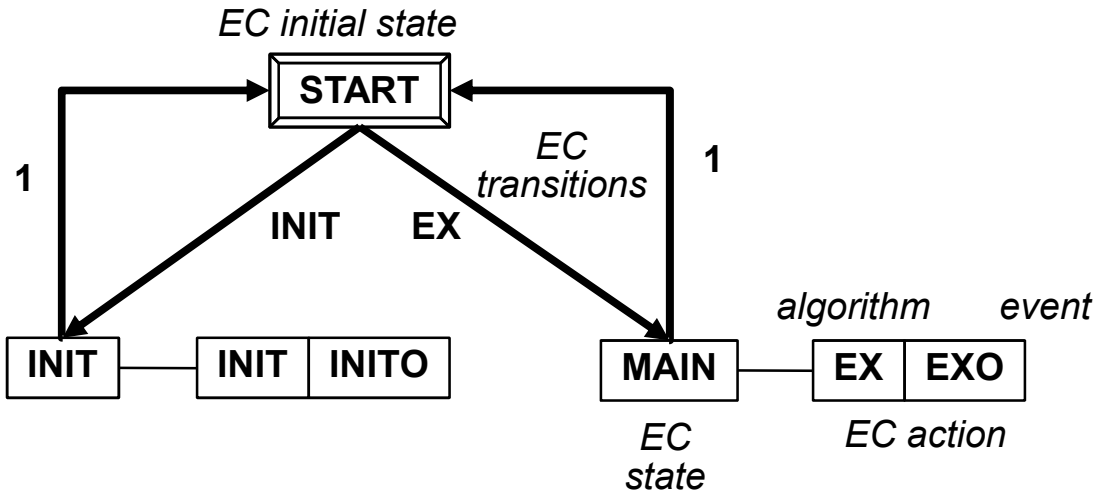
- Resource schedules & executes FB algorithms
- Resource maps Communications & Process I/O Functions to Service Interface Function Blocks



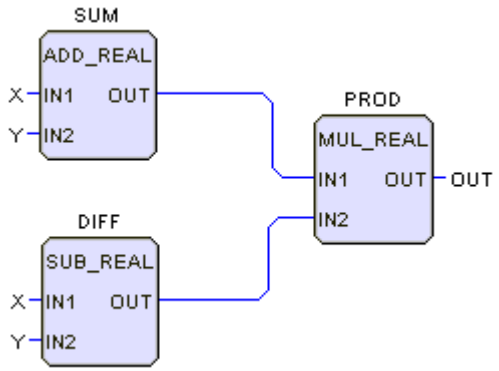
IEC 61499:
Basic Function Block Types



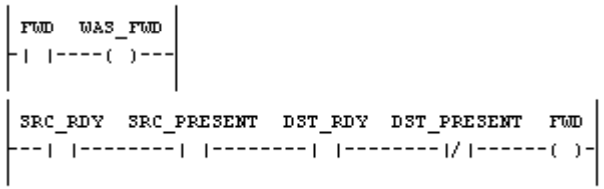
The Execution Control Chart (ECC): An Event-Driven State Machine



IEC 61499 Algorithms are programmed in IEC 61131-3 Languages



Function Block Diagram (FBD)



Ladder Diagram (LD)

```
OUT := (X-Y)*(X+Y);
```

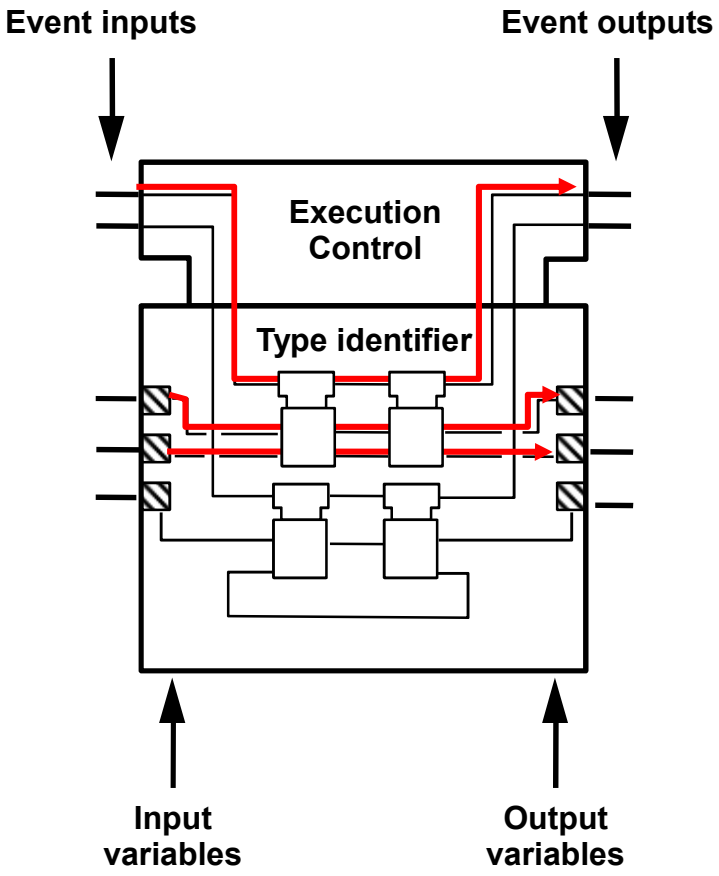
Structured Text (ST)

- Also Java, C++, etc, depending on software tool support

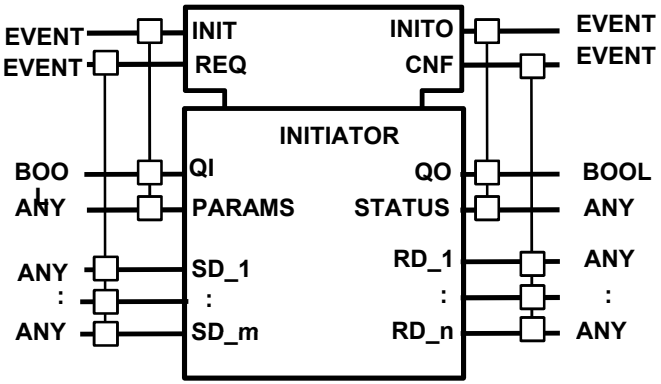
IEC 61499 Reuses 61131-3 Data Types

- **IEC 61131-3 and 61499 are Strongly Typed**
  - No mixed-type operations, assignments or connections allowed
  - Generic type hierarchy for overloading of functions
- **Signed Integers:** SINT(8), INT(16), DINT(32), LINT(64)
- **Unsigned Integers:** USINT(8), UINT(16), UDINT(32), ULINT(64)
- **Floating Point:** REAL(32), LREAL(64)
- **Bit Strings:** BOOL(1), BYTE(8), WORD(16), DWORD(32), LWORD(64)
  - TRUE, FALSE, 1, 0, 255, 16#FF, etc.
- **Character Strings:** STRING(8), WSTRING(16)
  - 'abc', “abc”, etc.
- **Duration:** TIME(t#2s, t#1500ms, etc.)
- TIME_OF_DAY or TOD, DATE, DATE_AND_TIME or DT
  - TOD#17:32:55.678, D#2005-06-07, DT#2005-06-07-17:32:55, etc.
- **Derived Data Types**
  - **Directly derived, subrange, enumerated, array, structured**

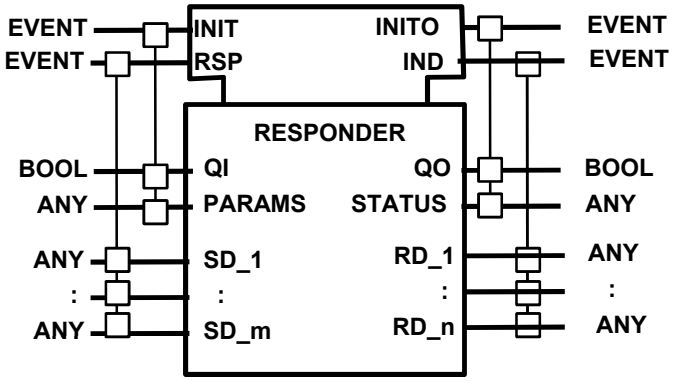
Composite Function Block Types



Service Interface Function Blocks



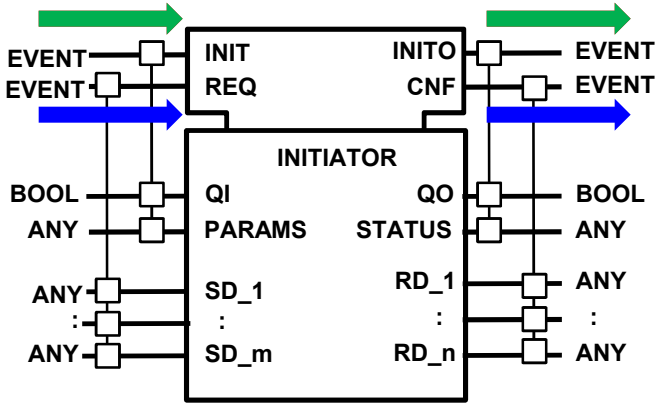
(application-initiated transactions)



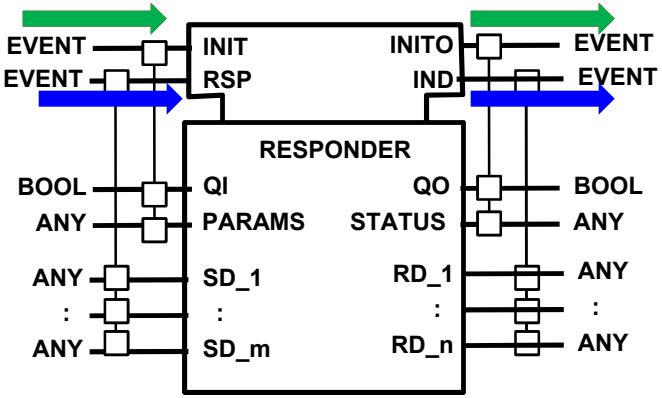
(resource-initiated transactions)

- Access to Resource functionality, e.g., **I/O, HMI, communications**
- Modelled as **sequences of service primitives**
- **Event & data interfaces standardized** as in ISO/IEC 10731, *Information technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services*

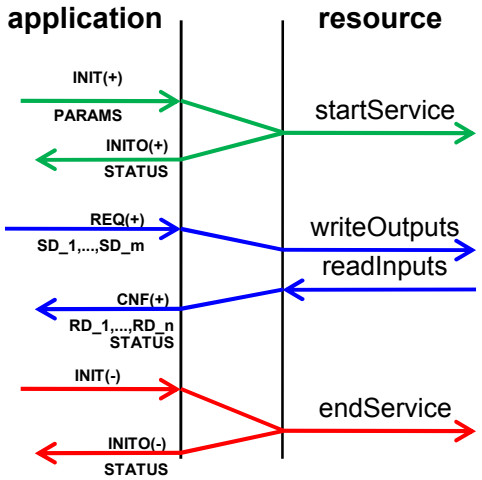
IEC 61499 Service Interface Function Blocks: Sensor/Actuator Interfaces

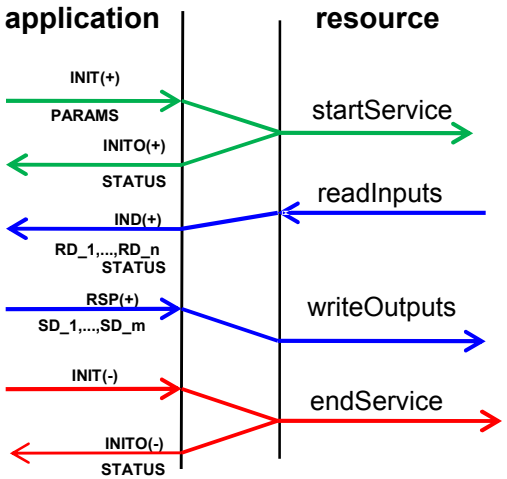


(application-initiated transactions)

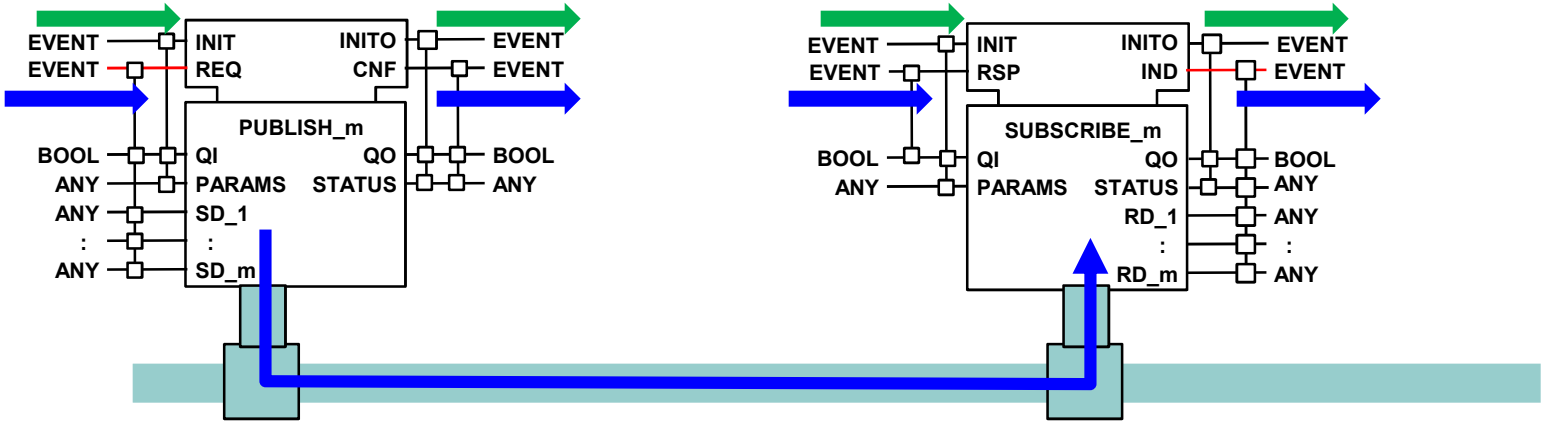


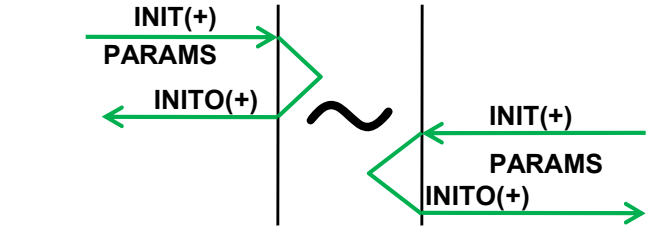
(resource-initiated transactions)

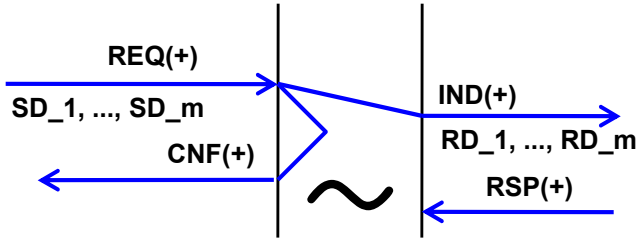




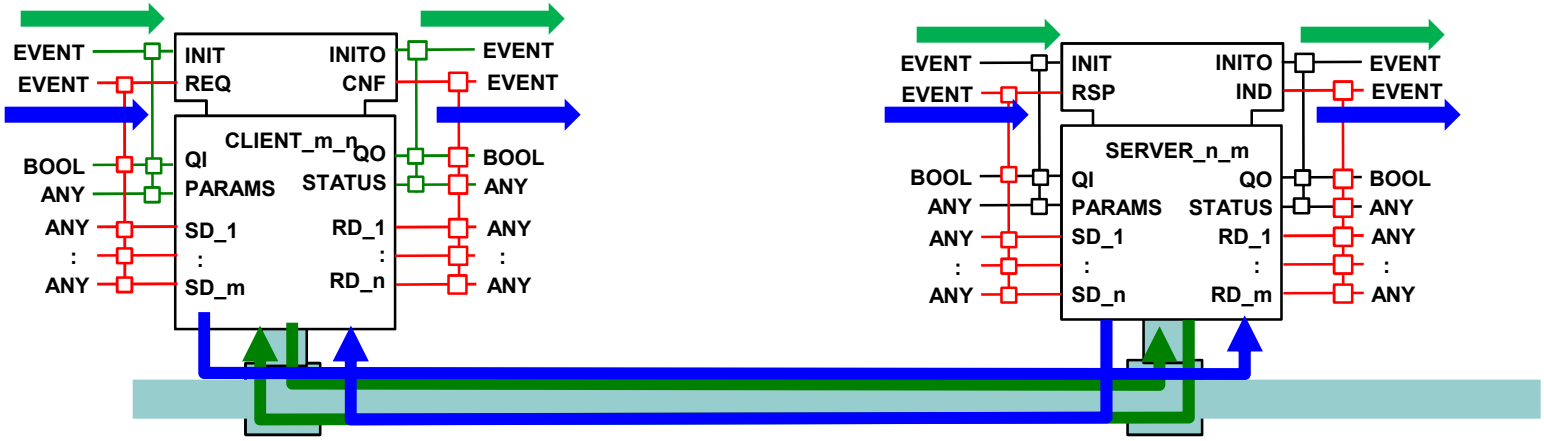
IEC 61499 Communication Service Interfaces: Publish/Subscribe Model

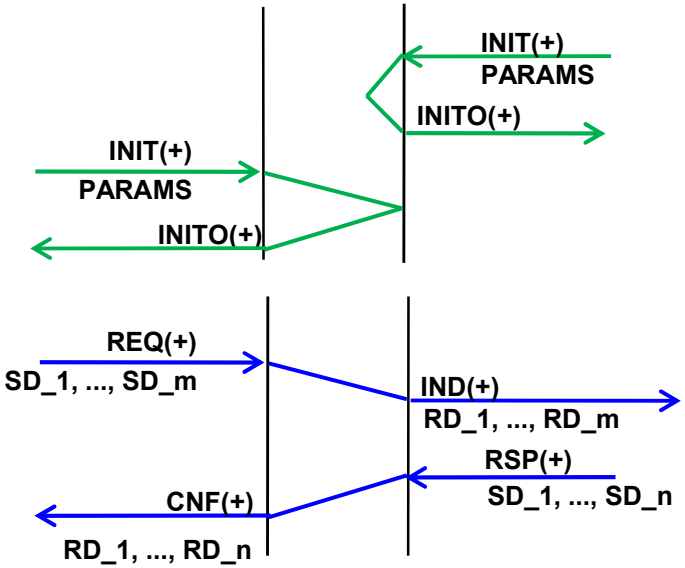




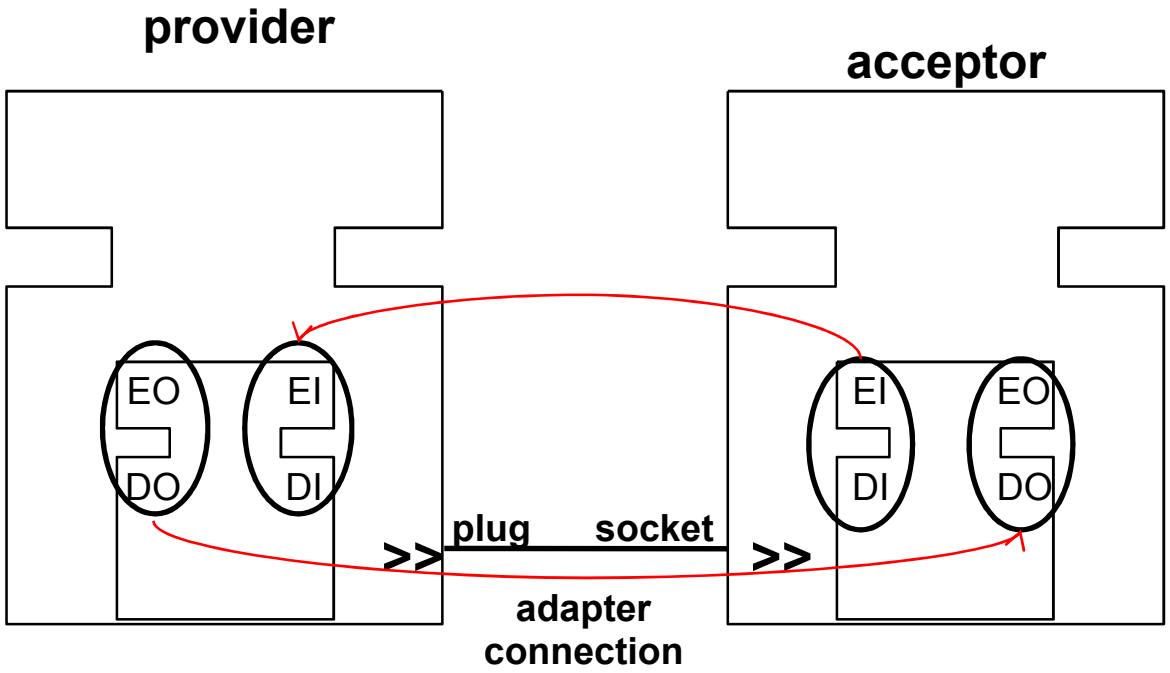


IEC 61499 Communication Service Interfaces: Client/Server Model



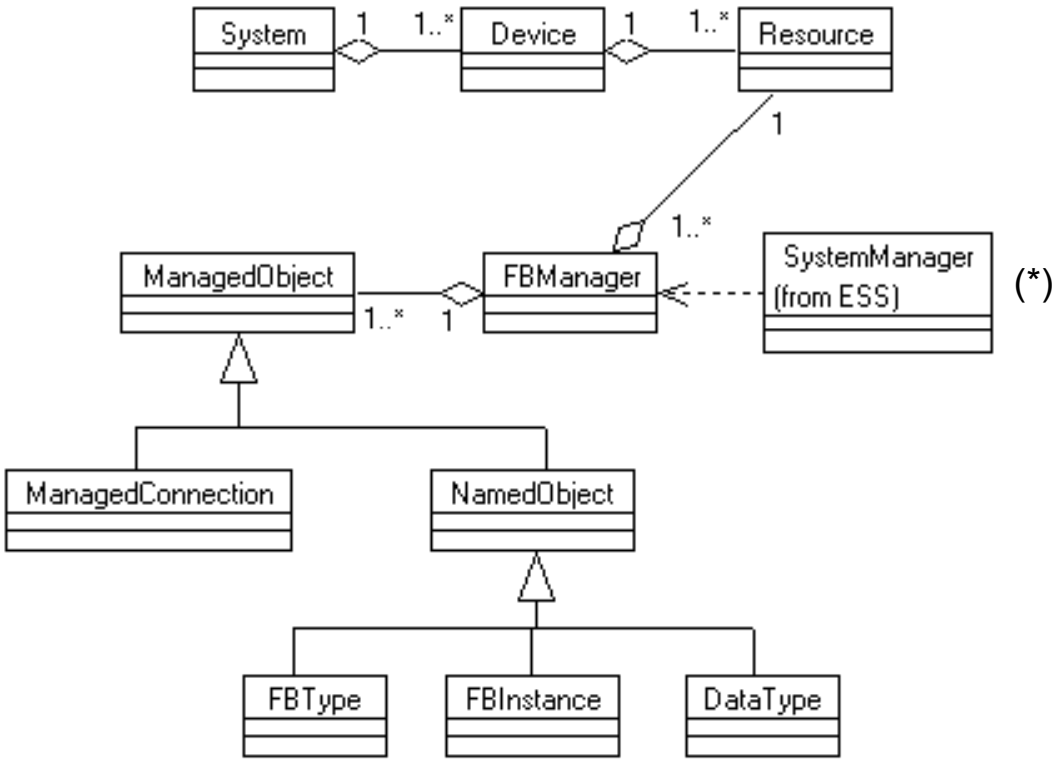


Adapter (Plug/Socket) Interfaces



- Reduce diagram clutter
- Simplify transducer interface
- Capture patterns of interaction

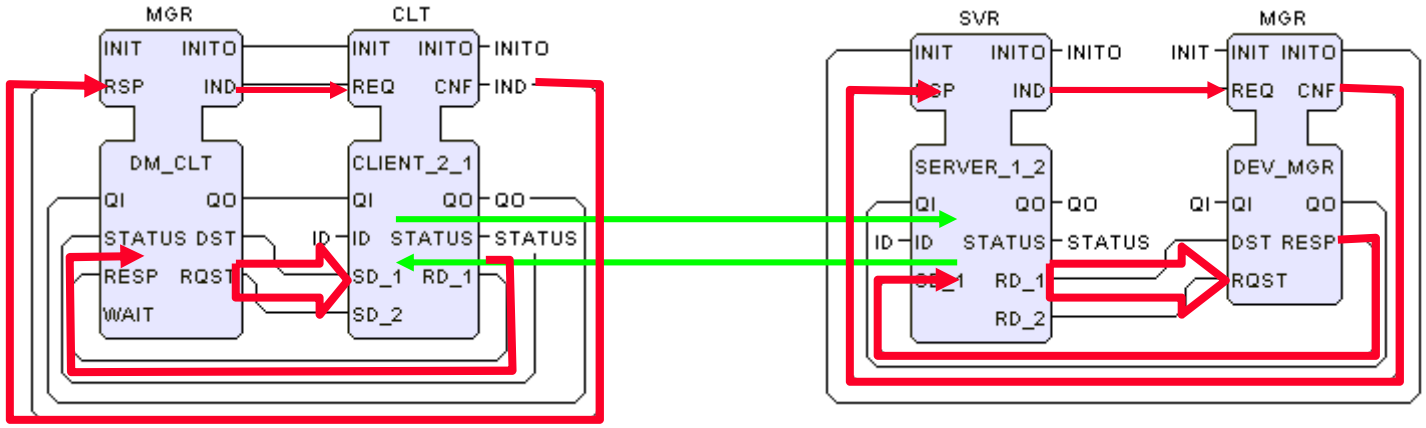
IEC 61499 System Model



*ESS = “Engineering Support System”
= Software Toolbox
= IDE (Integrated Development Environment)

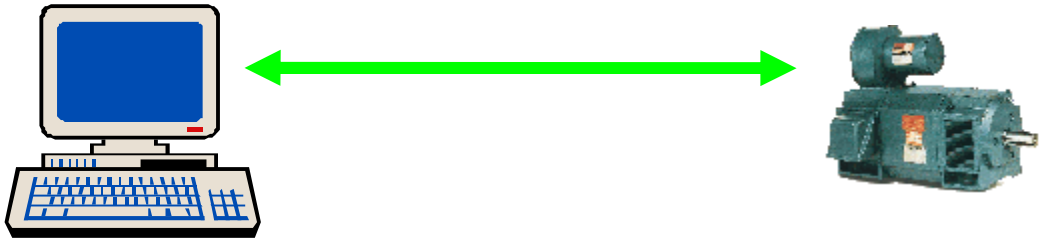
Device Management Architecture

- **Separation of Concerns**
  - Software Tools vs. Runtime Device
  - Communication Services vs. Management Services



Device Management Proxy (in Software Toolset)

Device Management Kernel (in Device)



Configuration Commands in XML

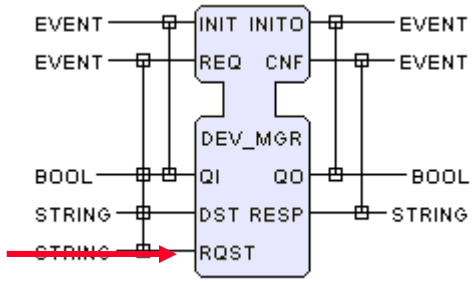
```
<Request ID="3" Action="CREATE" >
  <FB Name="DIAG" Type="SUBL_2" />
</Request>
```

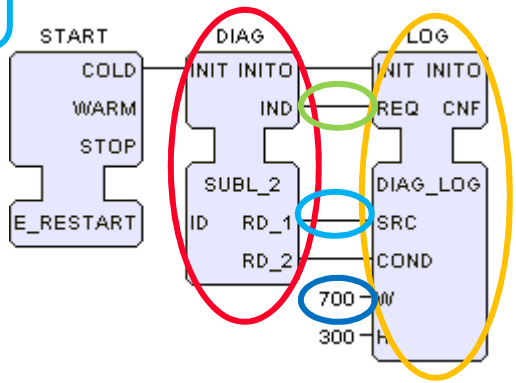
```
<Request ID="4" Action="CREATE" >
  <FB Name="LOG" Type="DIAG_LOG" />
</Request>
```

```
<Request ID="7" Action="CREATE" >
  <Connection Source="DIAG.IND" Destination="LOG.REQ" />
</Request>
```

```
<Request ID="8" Action="CREATE" >
  <Connection Source="DIAG.RD_1" Destination="LOG.SRC" />
</Request>
```

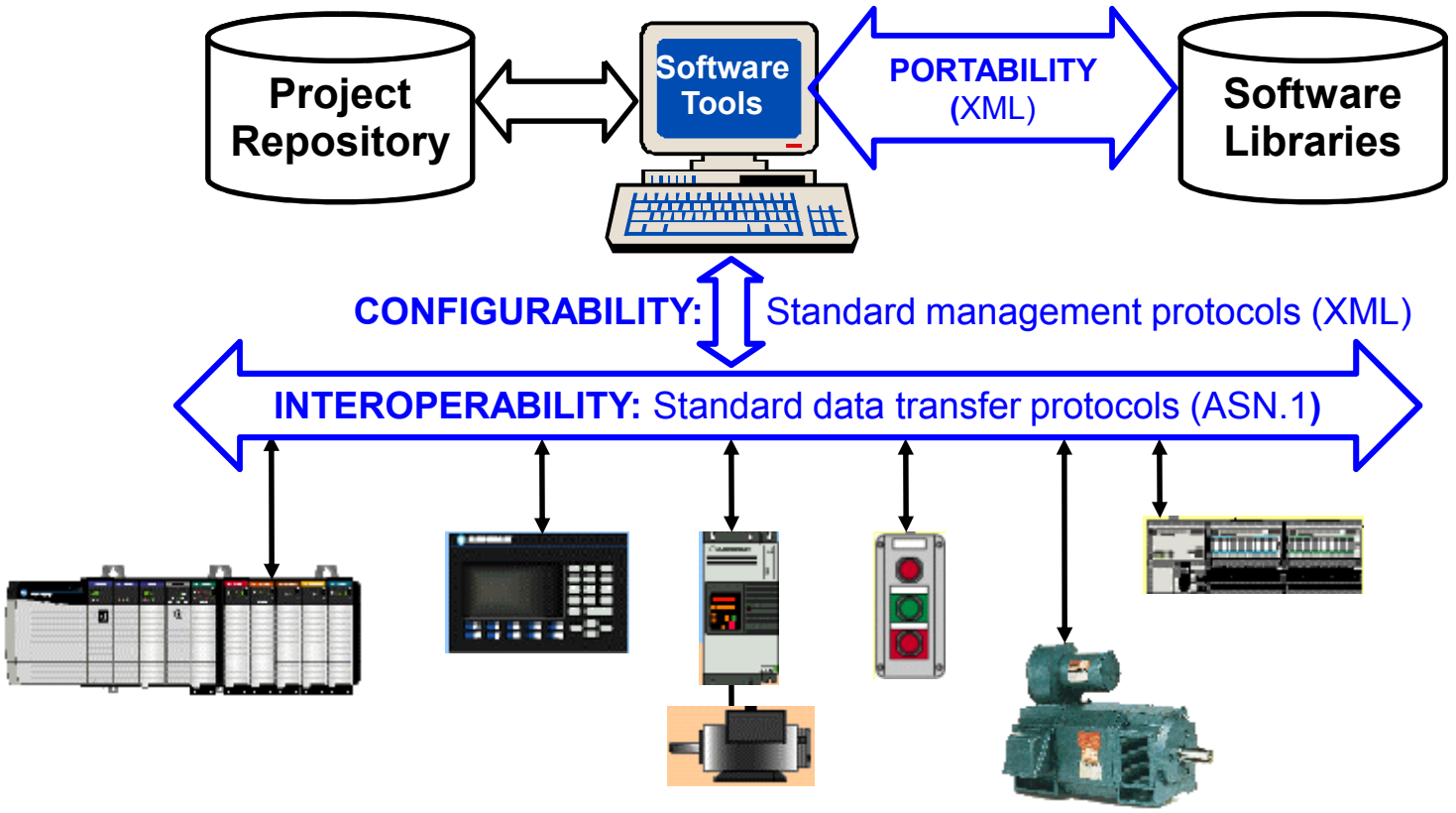
```
<Request ID="10" Action="WRITE" >
  <Connection Source="700" Destination="LOG.W" />
</Request>
```





- For details see IEC 61449-1, Clause 6.3.
- For additional information see Clause 6 at http://www.holobloc.com/doc/ita/index.htm.

Open Distributed Architecture - The IEC 61499 Solution



Distributed intelligent devices & controllers

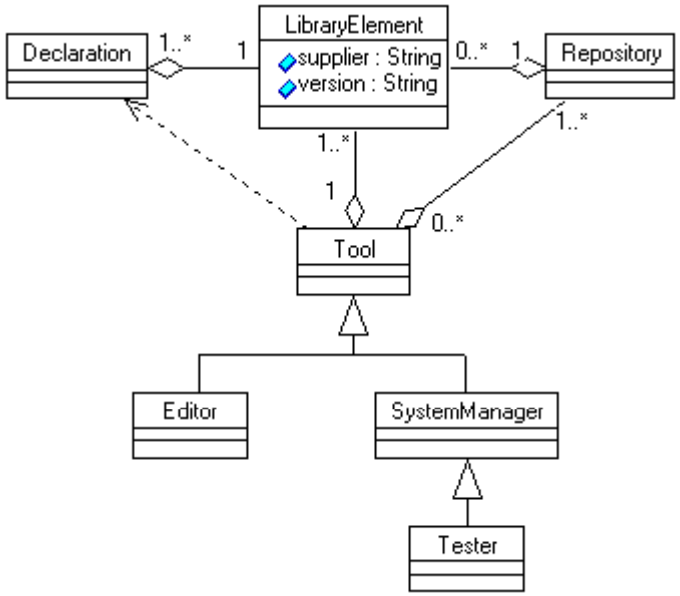
Adding Value with IEC 61499

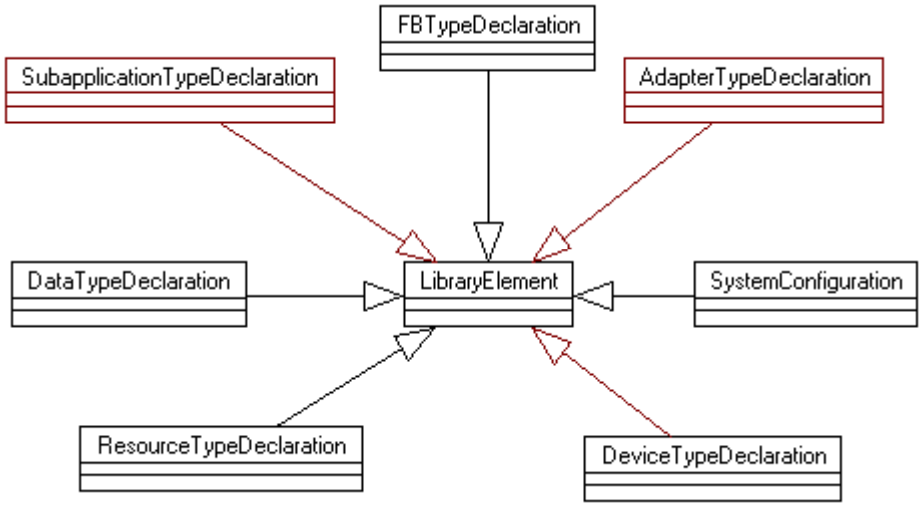
- Background
- Requirements
- Architecture
- **Software Tools**
- Design Patterns & Frameworks
- Runtime Platforms

IEC 61499-2: Software Tool Requirements

- Exchange of library elements
- Information to be provided by the supplier of library elements
- Display of declarations
- Modification of declarations
- Validation of declarations
- Implementation of declarations
- System operation, testing and maintenance
- XML DTDs for Library Elements
  - Document Type Definition (DTD) = "File Exchange Format"
  - Library Elements: Data Type, FB Type, Adapter Type, Subapplication Type, Resource Type, Device Type, System Configurations

IEC 61499 Software Tool Model





IEC 61499 Software Tools

- **FBDK/FBRT (Function Block Development Kit & Runtime Platform)**
  - Publicly available
  - http://www.holobloc.com/doc/fbdk/index.htm
- **ISaGRAF**
  - Commercially supported
  - http://www.isagraf.com/
- **FBench**
  - Open source
  - http://oooneida-fbench.sourceforge.net/
- **4DIAC**
  - Open source
  - C++ runtime, Eclipse-based IDE
  - http://www.fordiac.org/
- **CORFU**
  - Framework, Methodology, Toolset, Runtime Environment
  - http://seg.ee.upatras.gr/corfu/dev/index.htm
- **Other research tools**
  - http://en.wikipedia.org/wiki/IEC_61499

Adding Value with IEC 61499

- Background
- Requirements
- Architecture
- Software Tools
- **Design Patterns & Frameworks**
- Runtime Platforms

Design Patterns & Frameworks

- **Design patterns**
  - A **formalized approach** to a **common problem** within a **context** (B.P. Douglass, *Real-Time UML*,1998)
  - **Our context**: distributed control and automation with IEC 61499 architecture
- **Frameworks**
  - “A skeletal structure...that must be fleshed out to build a complete application” (R. Wirfs-Brock *et.al., Designing Object-Oriented Software*,1990)
  - May integrate **multiple design patterns**

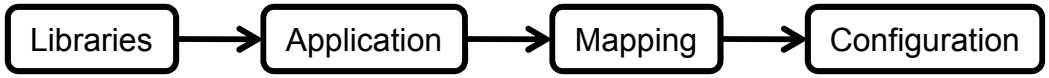
Example Pattern: Scanned Execution

- **Function blocks executed (scanned) in a fixed order**
  - Typically triggered by periodic tasks
  - Individual blocks enabled/disabled with Boolean variables
  - Sequences implemented with state variables IEC 61131-3 Sequential Function Charts (SFCs)
- **Advantage**
  - Predictable performance in small, high-performance systems (e.g., DSPs for motor control) or larger low-speed systems (e.g., some process control applications)
- **Disadvantages**
  - High processor loading in centralized systems (e.g., PLCs)
  - High communications bandwidth loading in distributed systems

Design Patterns and Frameworks for IEC 61499

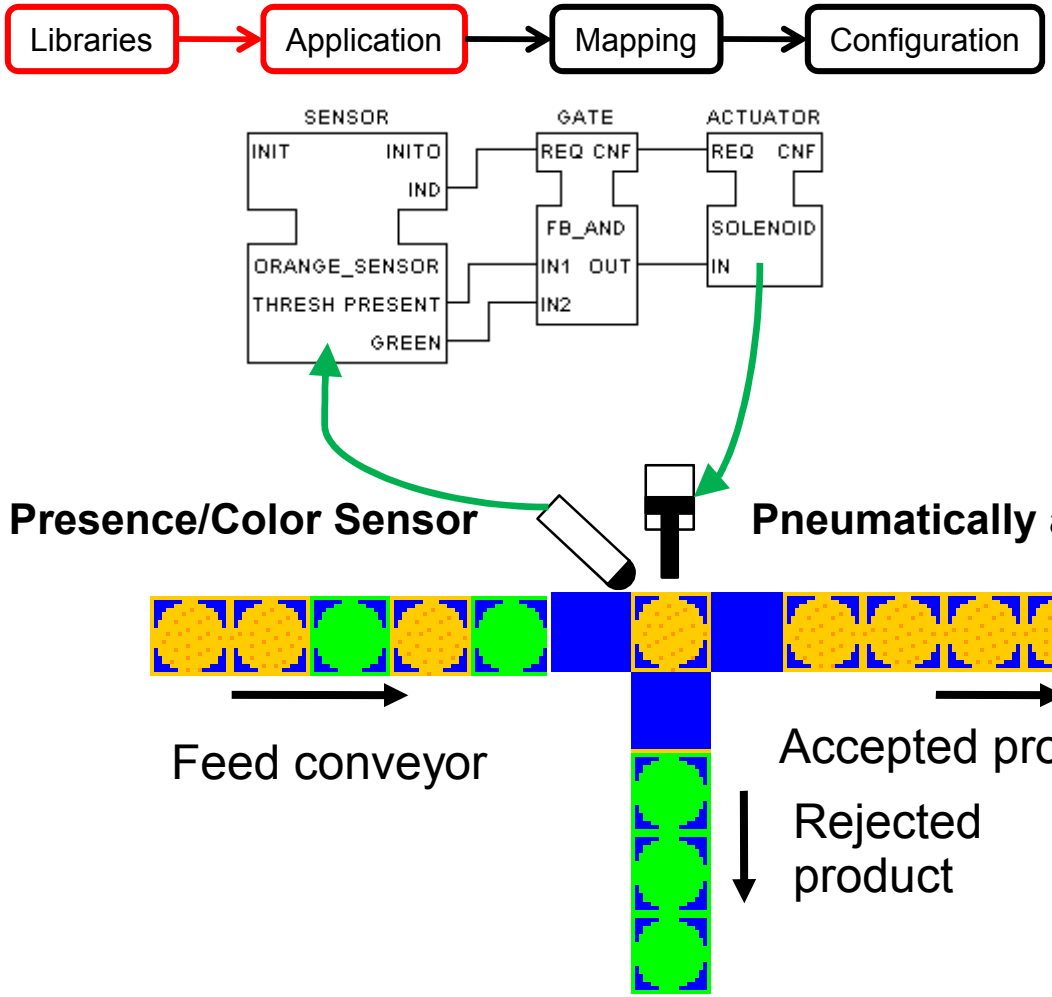
- **Problem**: Account for communication and synchronization in distributed systems
  - **Methodology**: Distributed Application Development
- **Problem**: Improve efficiency of intraprocess communication
  - **Design Pattern**: Local Multicast
- **Problem**: Maintain consistency of communicated data
  - **Design Pattern**: Tagged Data
- **Problem**: Remove I/O dependencies from applications
  - **Design Pattern**: Proxy
- **Problem**: Integrate simulation and deployment
  - **Framework**: Layered Model/View/Controller
- **Problem**: Utilize processing power in physical devices
  - **Design Pattern**: Mechatronic
- **Problem**: Implement features of Harel Statecharts
  - **Design Pattern**: Statechart Mapping
- **Problem**: Ensure consistent operating mode throughout a distributed application.
  - **Design Pattern**: Multicast Mode
  - Uses **Tagged Data** pattern

Methodology for Distributed Applications

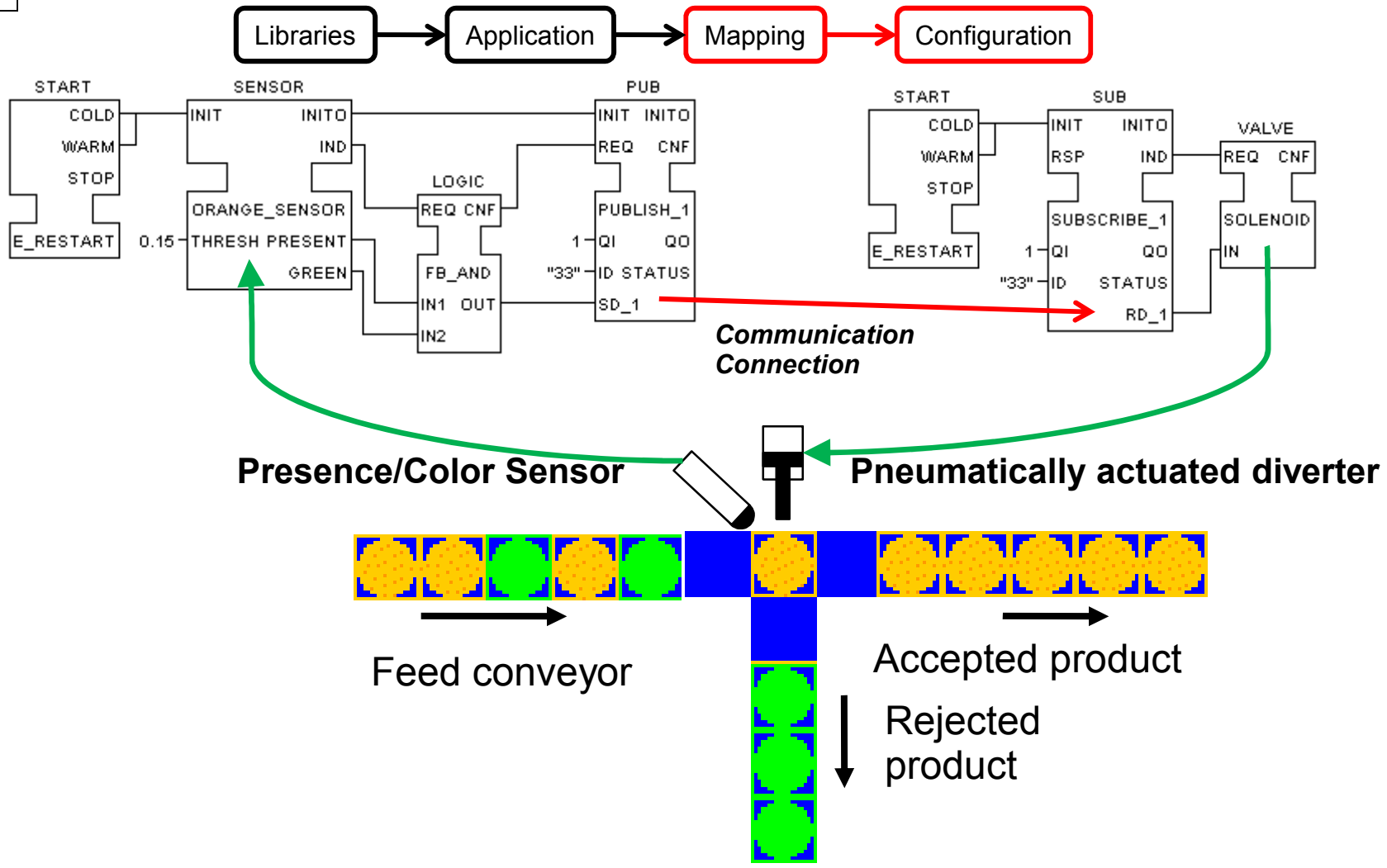


1. Obtain or develop a library of function block, resource and device *types*.
2. Define and develop the *application*.
3. Map function block *instances* from the *application* to distributed *resources*.
4. Configure *devices* and *resources*.
5. Configure *communication connections*, using *communication service interface function blocks* to implement the *event connections* and *data connections* of the *application* across resource boundaries

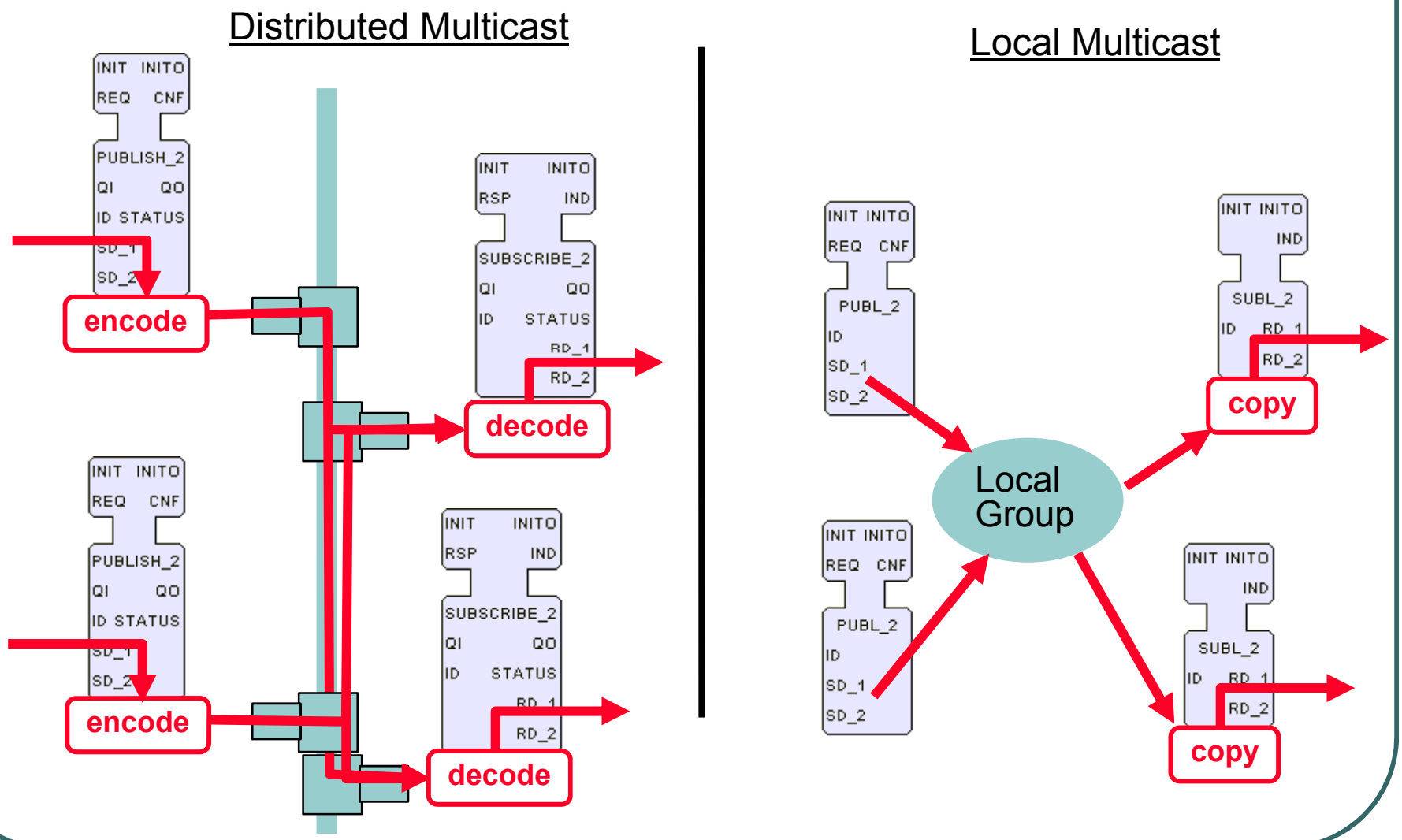
Example: Orange Sorter



Distributed Orange Sorter

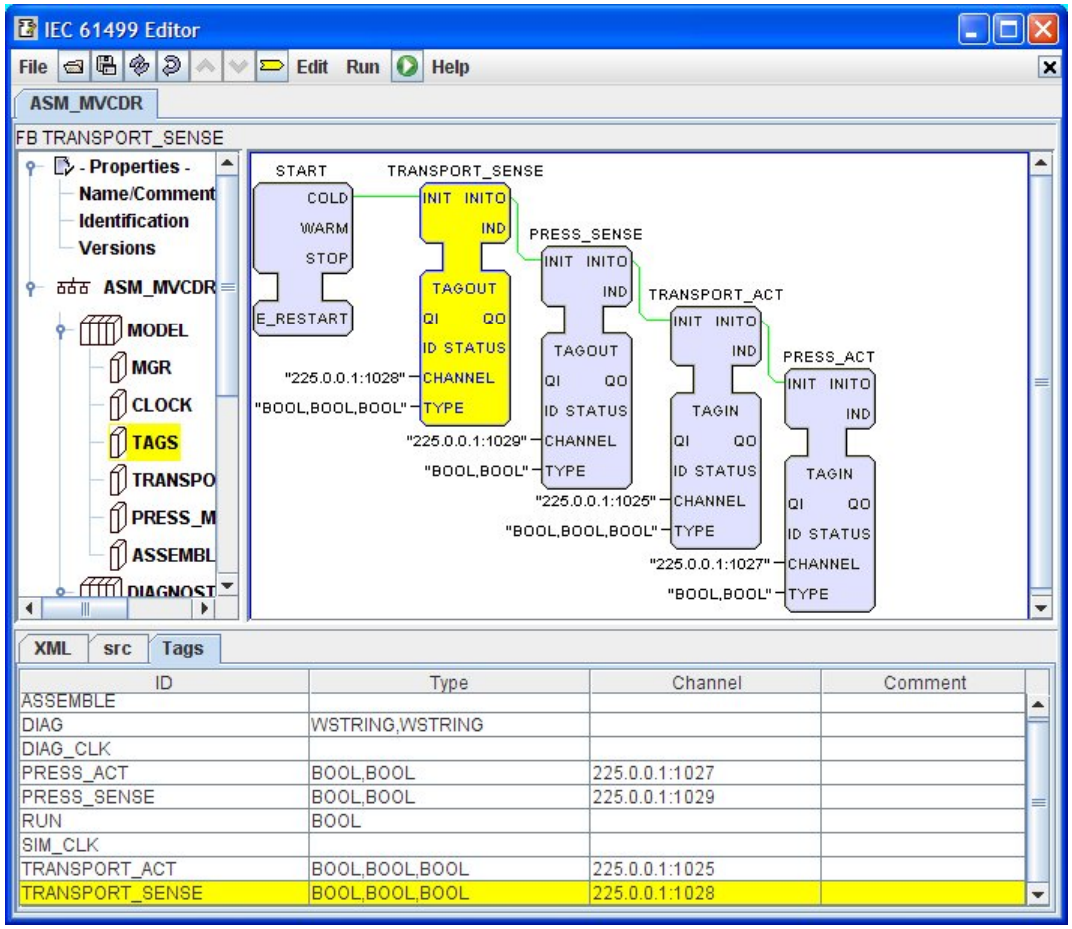


Design Pattern: Local Multicast



Design Pattern: Tagged Data

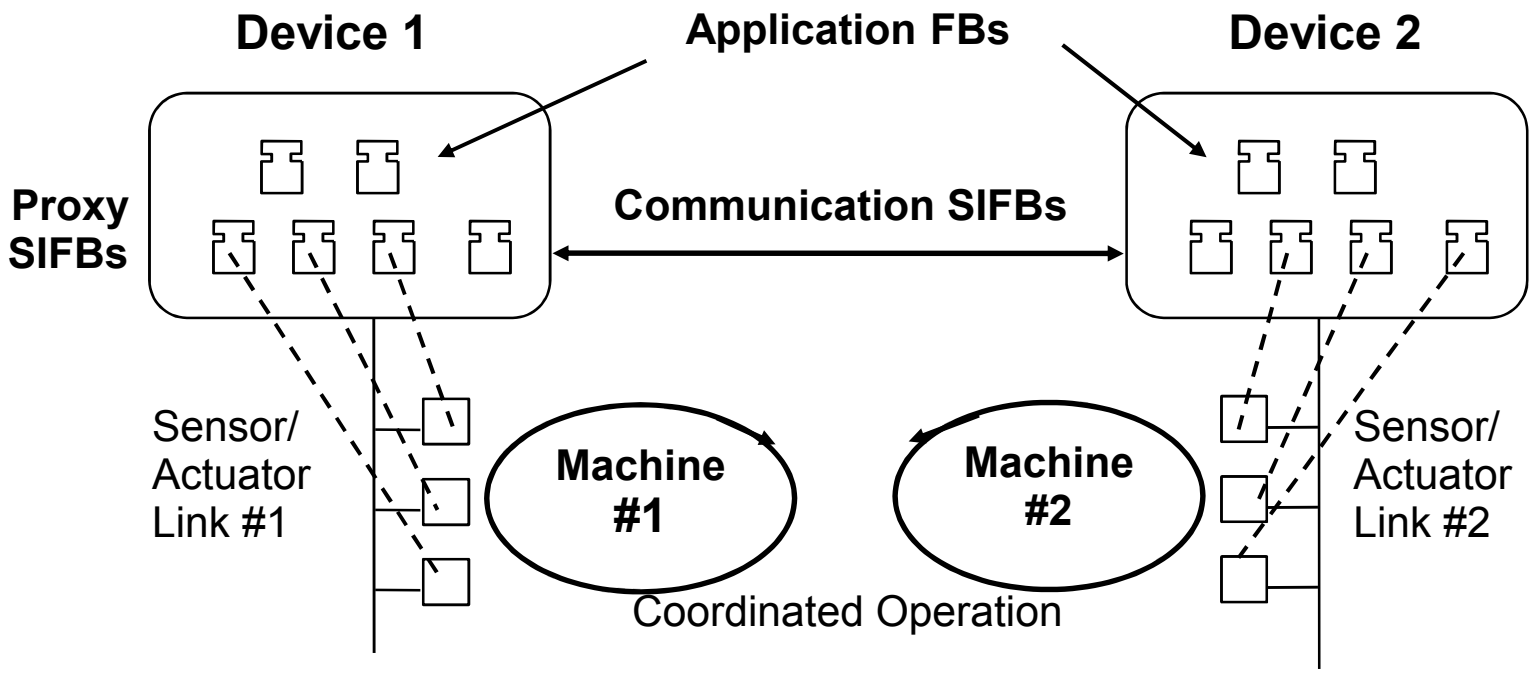
- Ensures that:
  - data used in a local multicast channel is consistent with that used in the corresponding distributed multicast channel
  - data subscribed from a multicast channel is consistent with that published on the channel



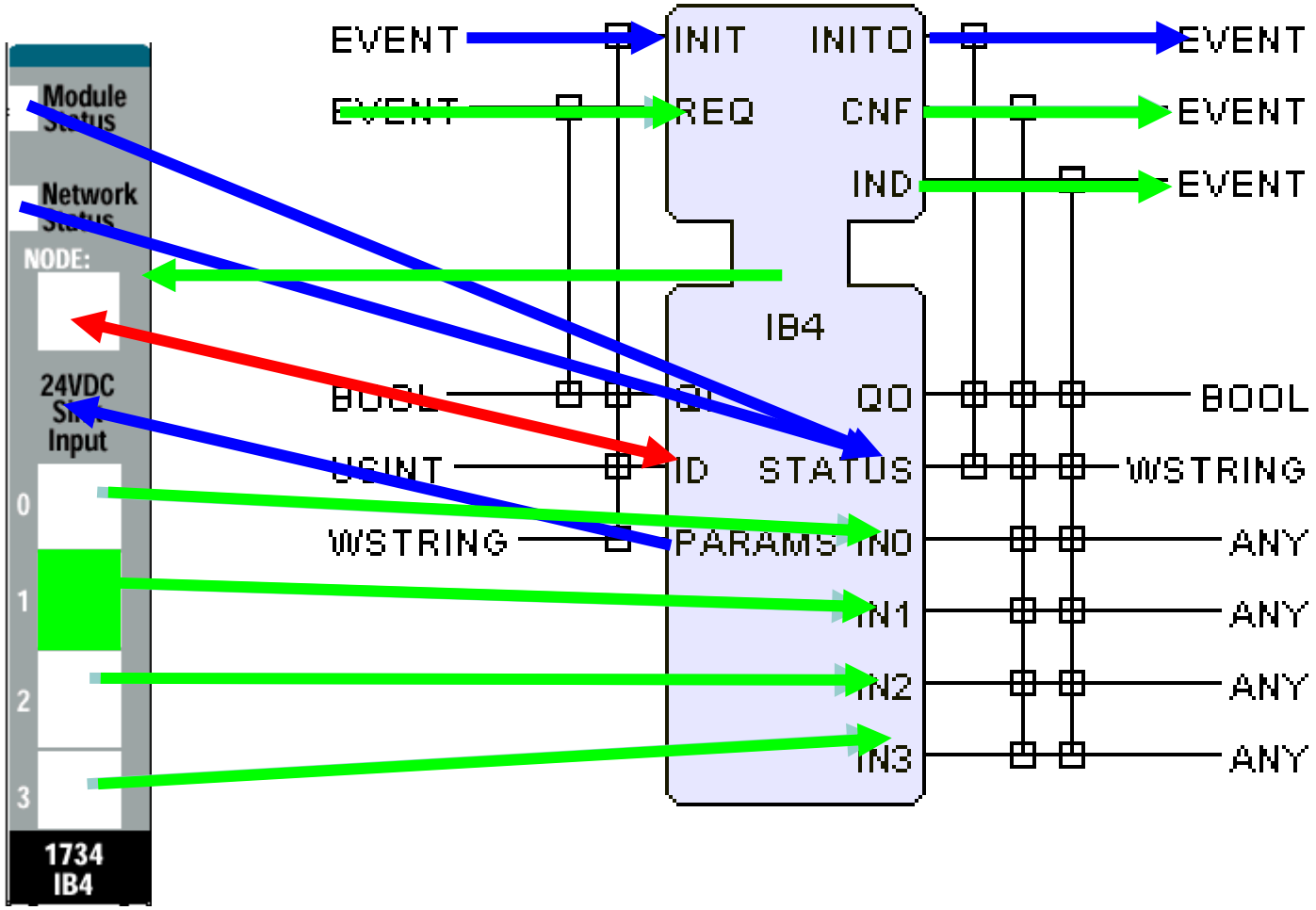
ID	Type	Channel	Comment
ASSEMBLE			
DIAG	WSTRING,WSTRING		
DIAG_CLK			
PRESS_ACT	BOOL,BOOL	225.0.0.1:1027	
PRESS_SENSE	BOOL,BOOL	225.0.0.1:1029	
RUN	BOOL		
SIM_CLK			
TRANSPORT_ACT	BOOL,BOOL,BOOL	225.0.0.1:1025	
TRANSPORT_SENSE	BOOL,BOOL,BOOL	225.0.0.1:1028	

Design Pattern: Proxy

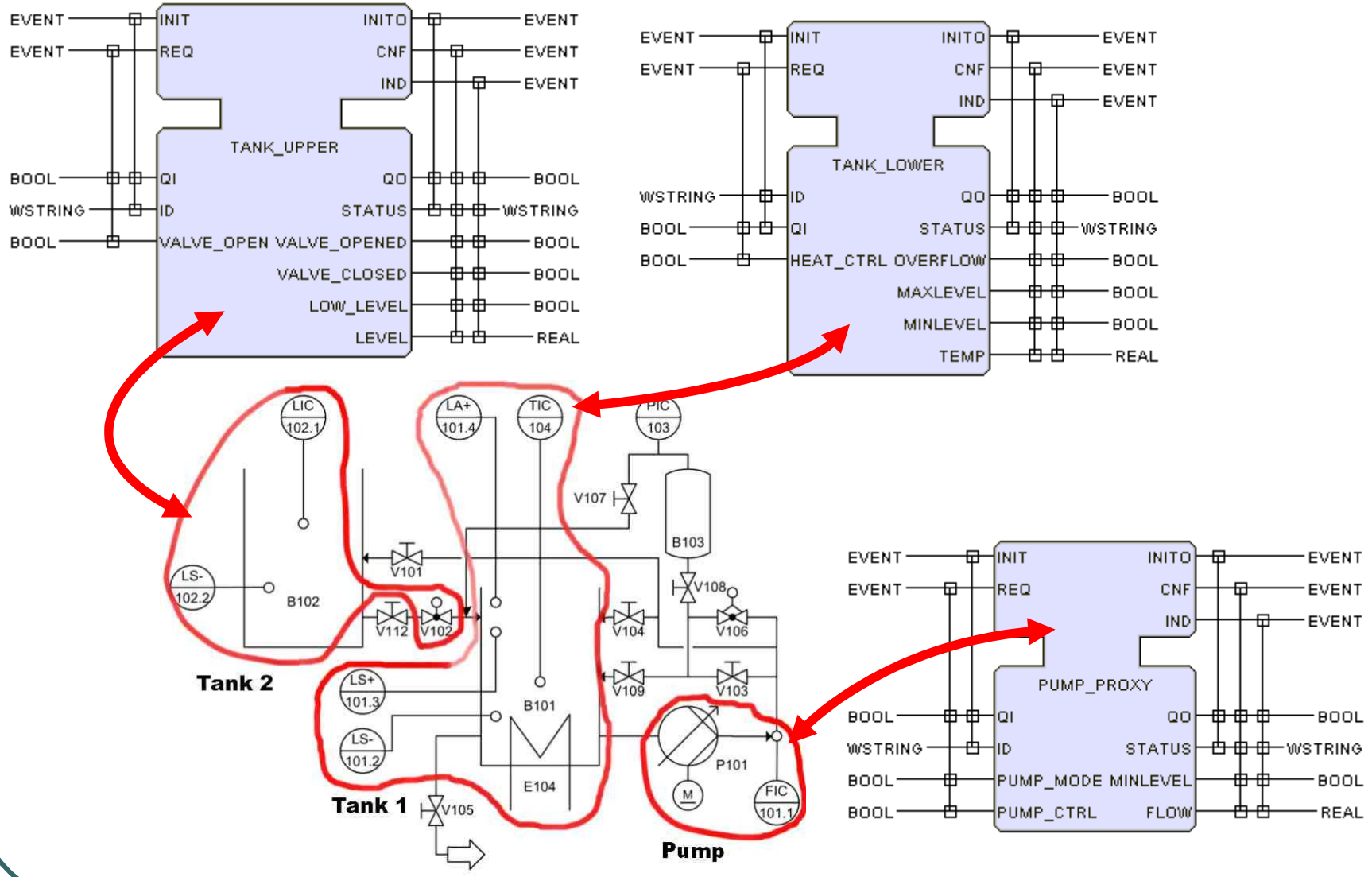
…"decouples clients from their servers by creating a local proxy, or stand-in, for the less accessible server. When the client needs to request a service from the server, such as retrieving a value, it asks its local proxy. The proxy can then marshal a request to the original server… " (Douglass, 1998)



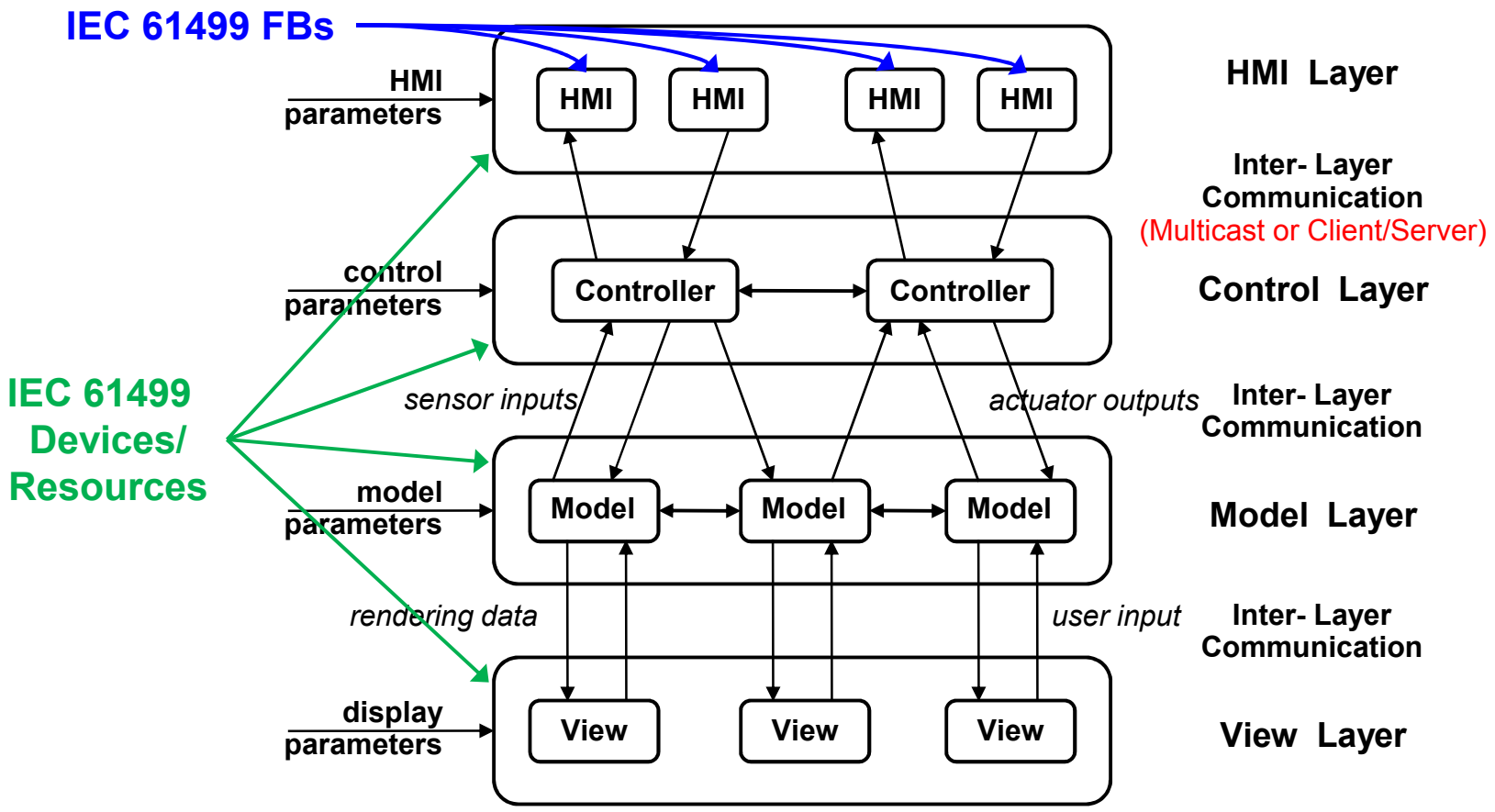
An I/O Proxy Service Interface



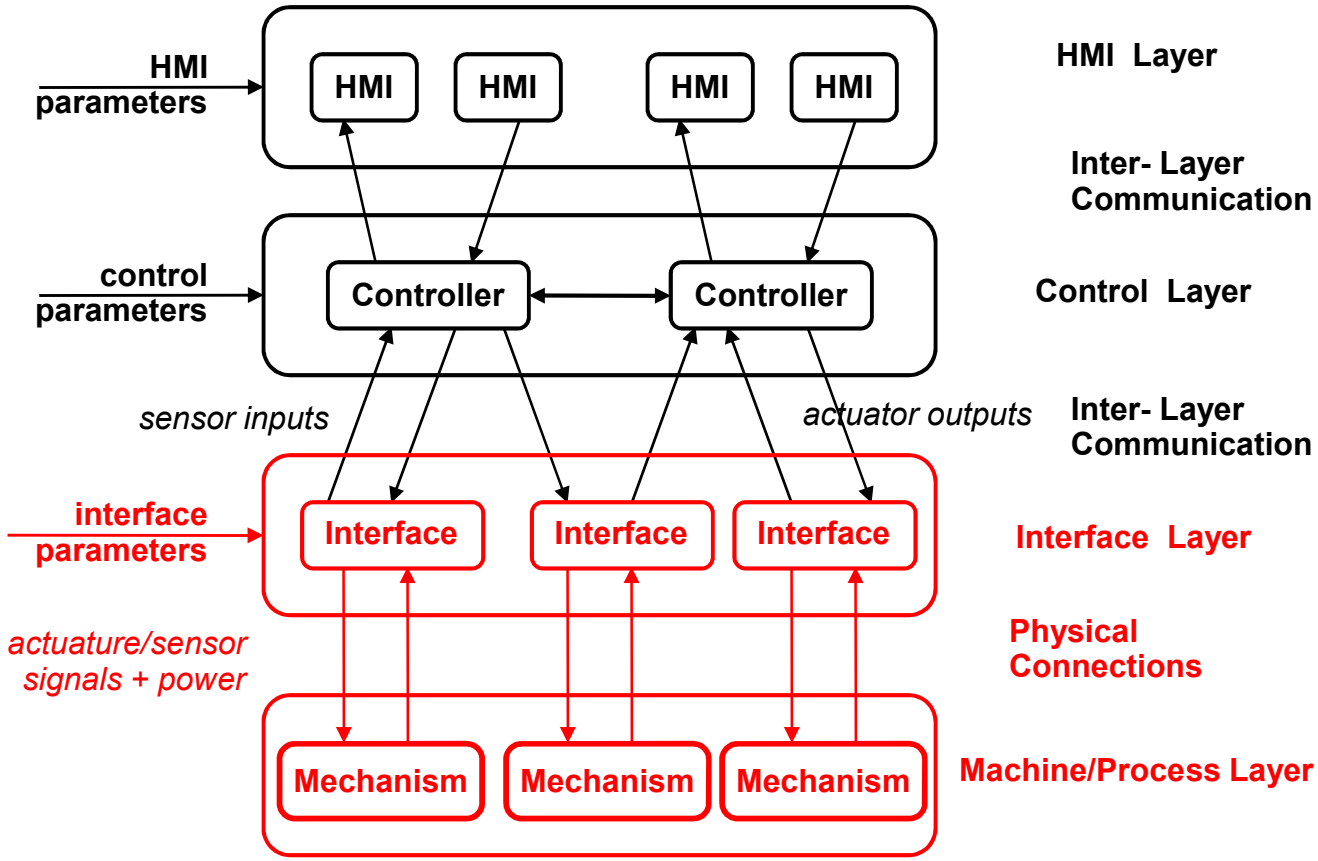
Proxies for Physical Devices



Framework: Layered MVC (Model/View/Controller)



Layered MVC Realization: Simulation => Physical Interface



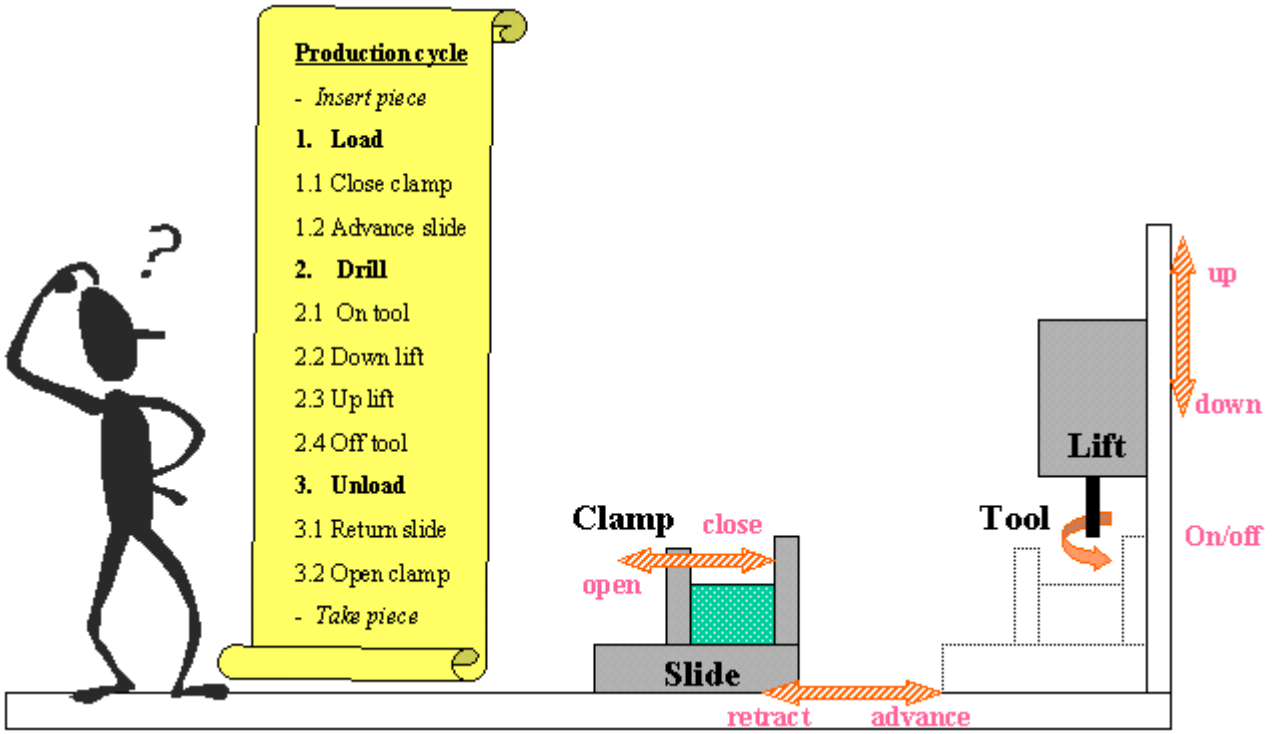
Layered MVC: Methodology

1. Sketch & describe the problem to be solved.
2. Develop & test Views.
3. Animate the desired operational sequences.
4. Develop & test Models.
5. Develop & test Controllers.
6. Develop & test Diagnostic & fault recovery elements.
7. Perform distribution design.
8. Integrate to physical components and systems.

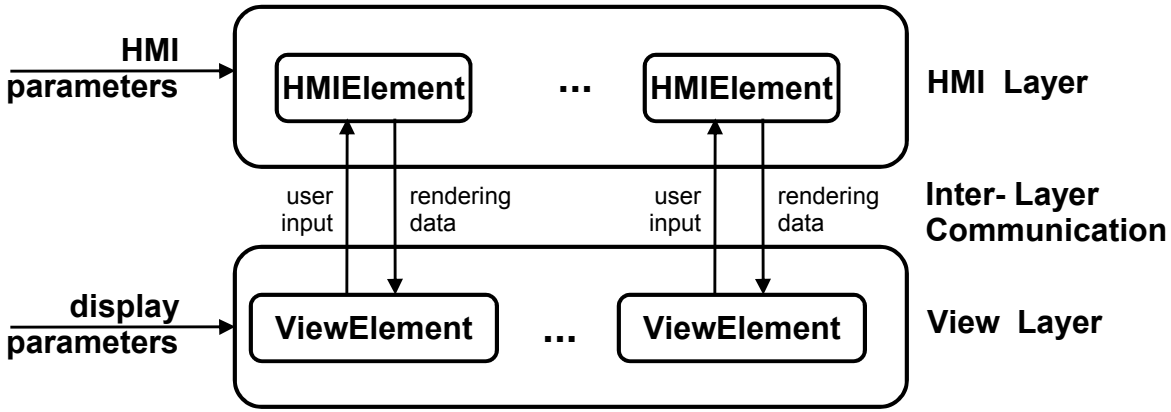
Examples: http://www.holobloc.com/doc/despats/mvc/index.htm

Layered MVC: Sketch & Description

Sketch → Views → Animation → Models → Controllers → Diagnostics → Distribution → Physical

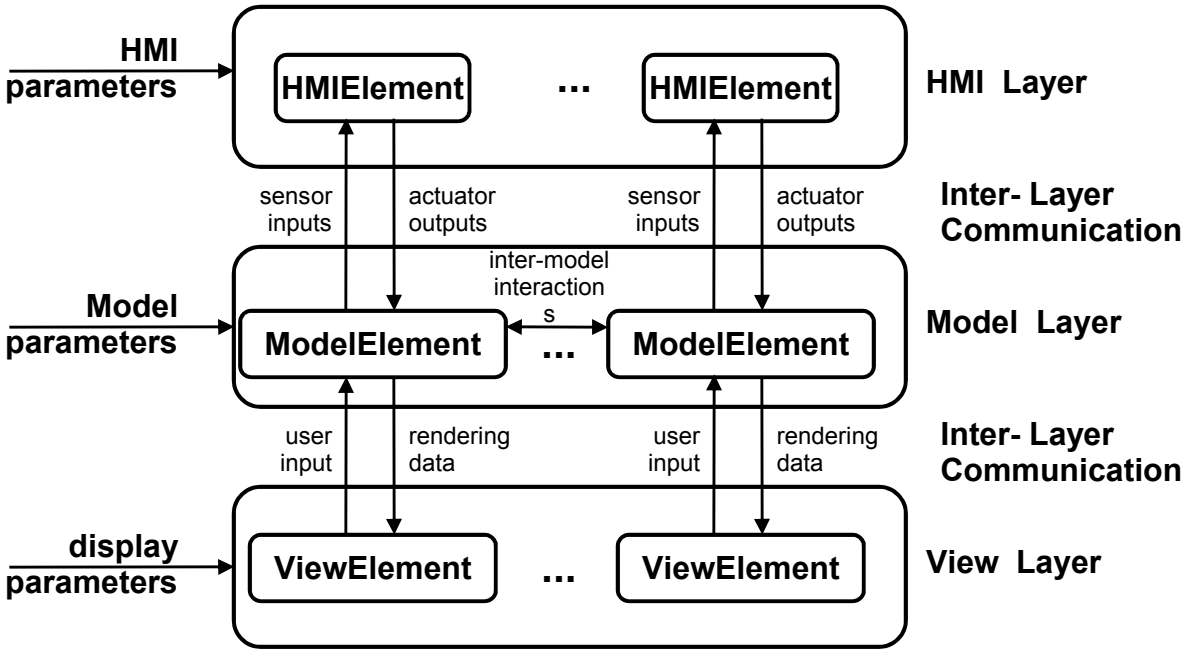


Layered MVC: View Development



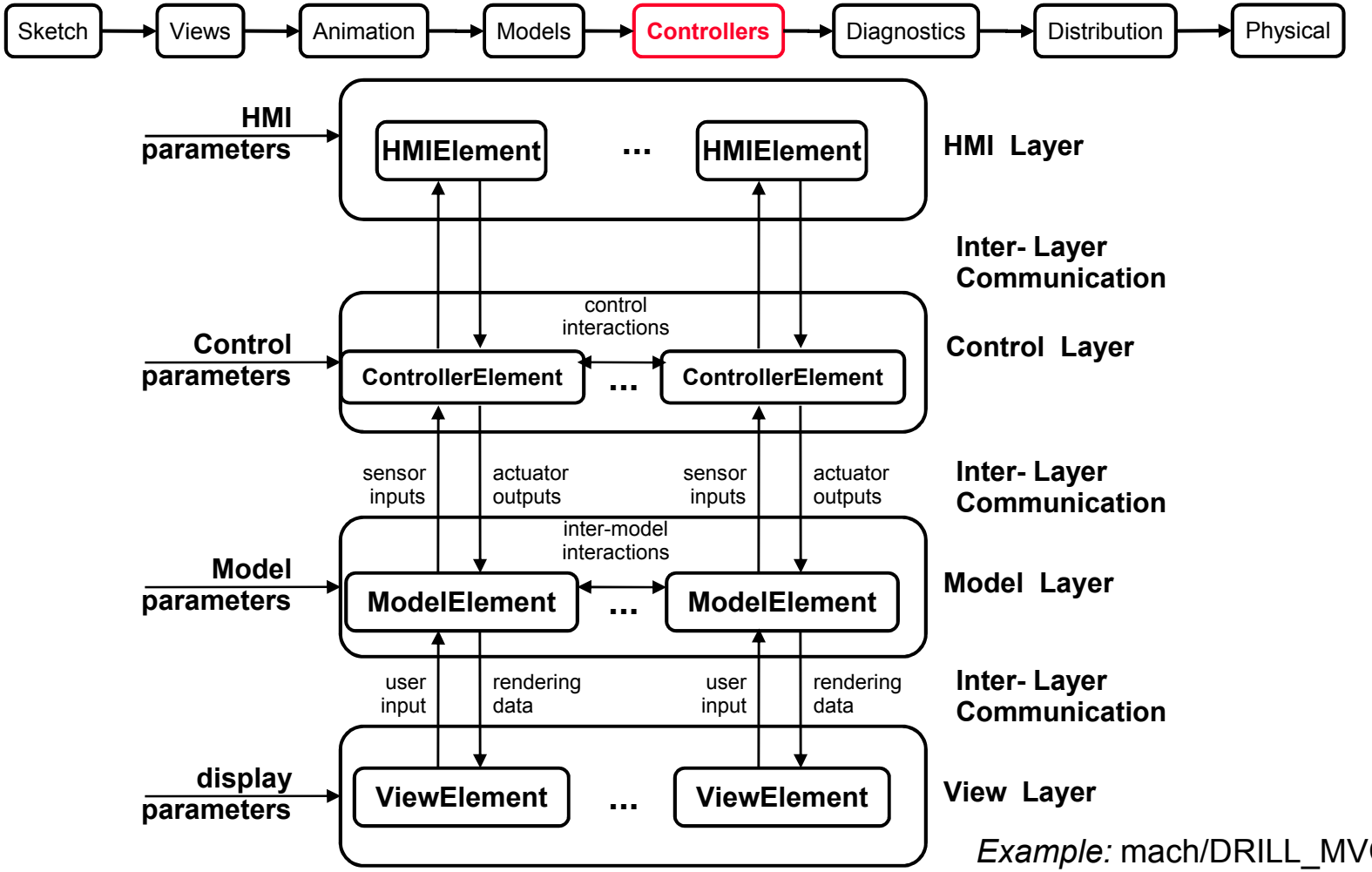
Example: mach/DRILL_VIEWL

Layered MVC: Model Development



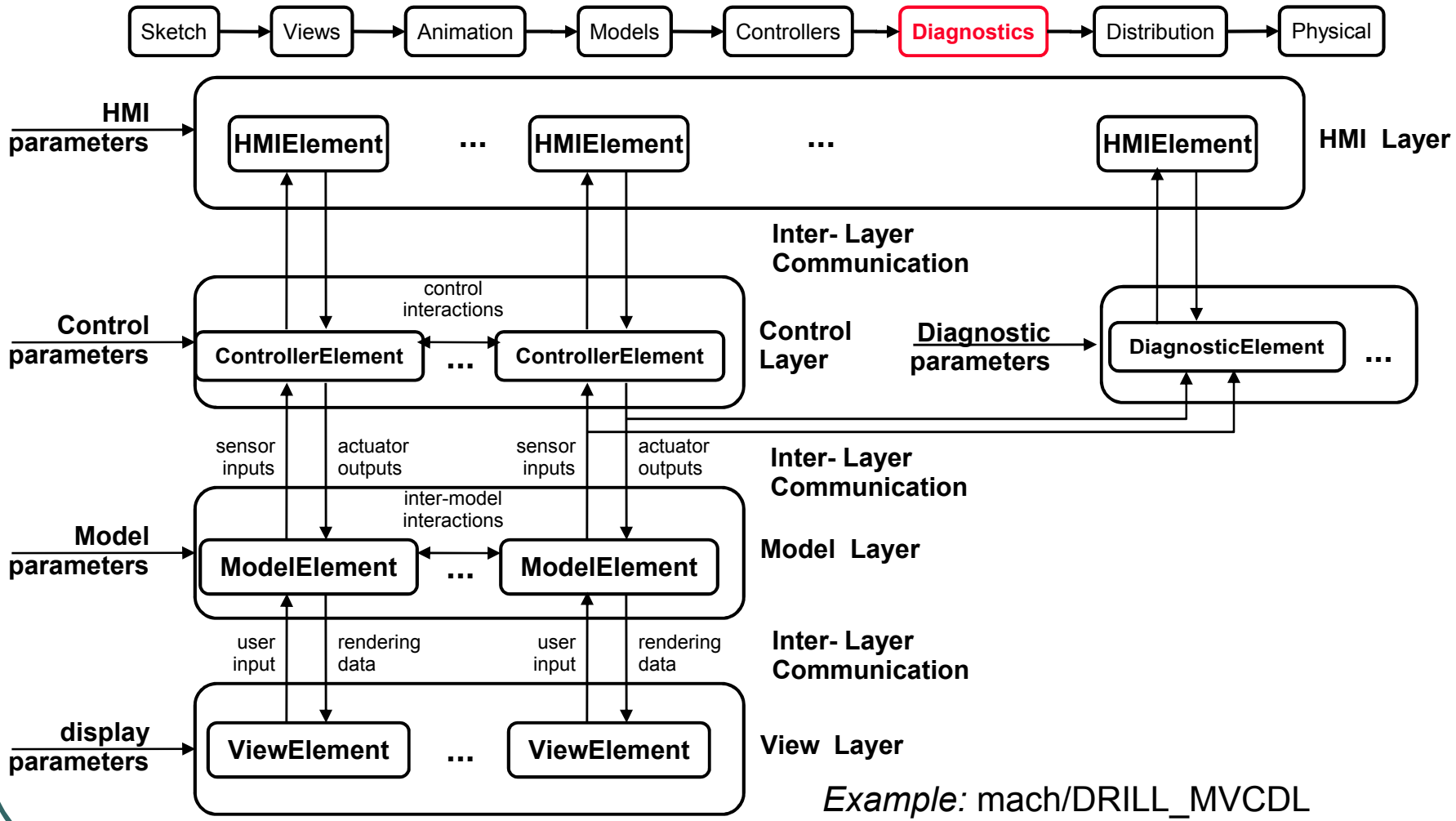
Example: mach/DRILL_MVL

Layered MVC: Controls Design



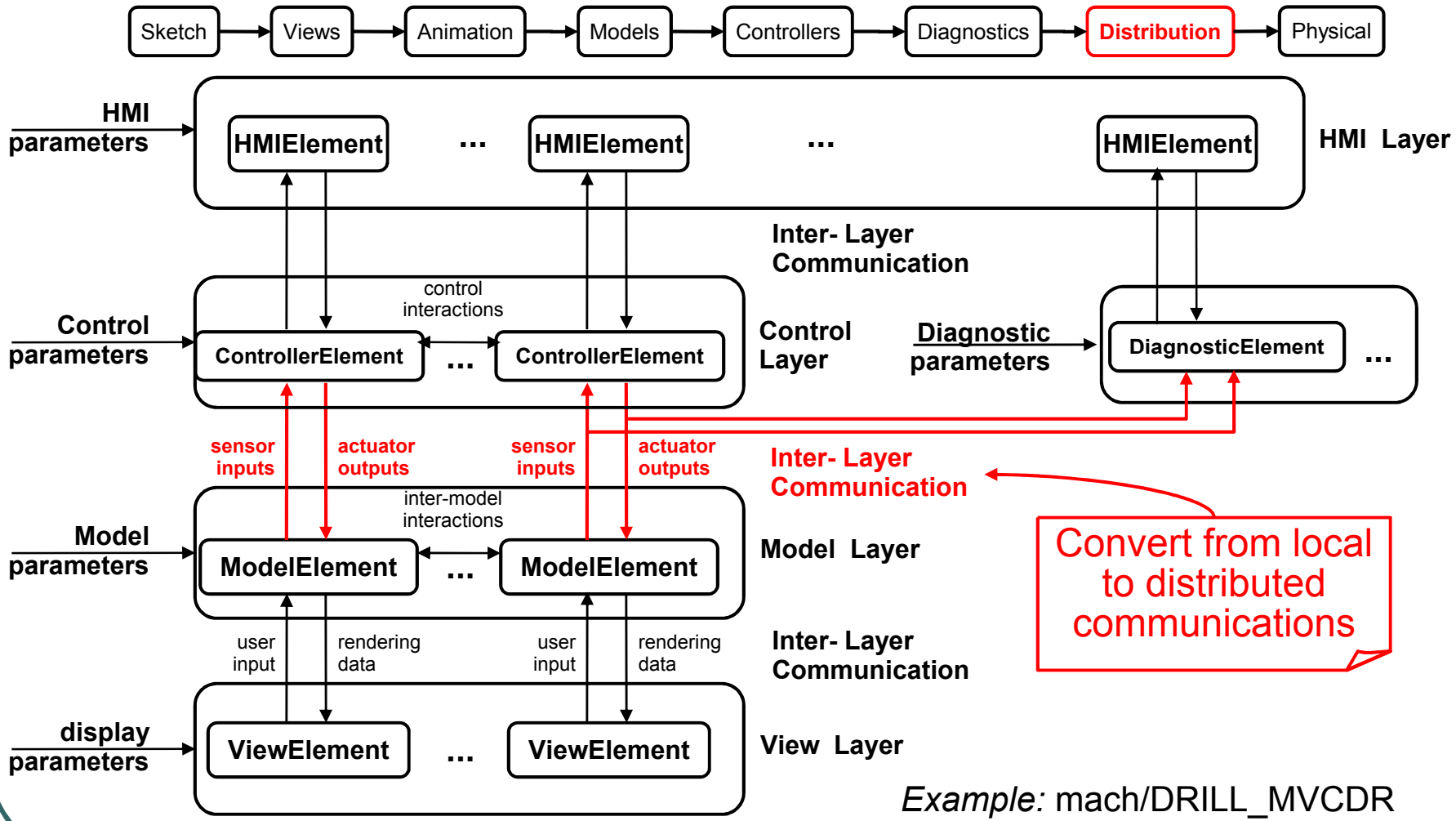
Example: mach/DRILL_MVCL

Layered MVC: Low-Level Diagnostics



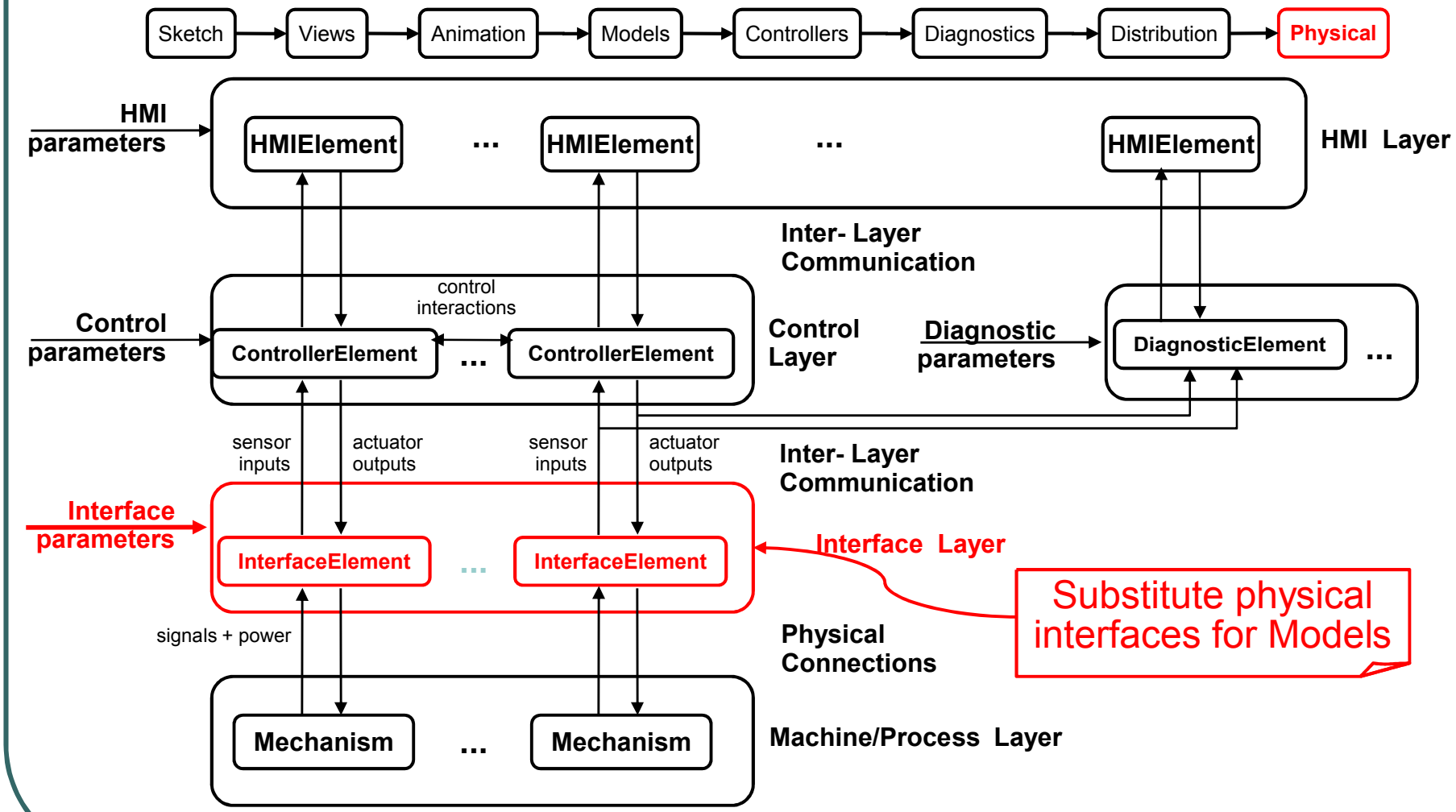
Example: mach/DRILL_MVCDL

Layered MVC: Distribution Design

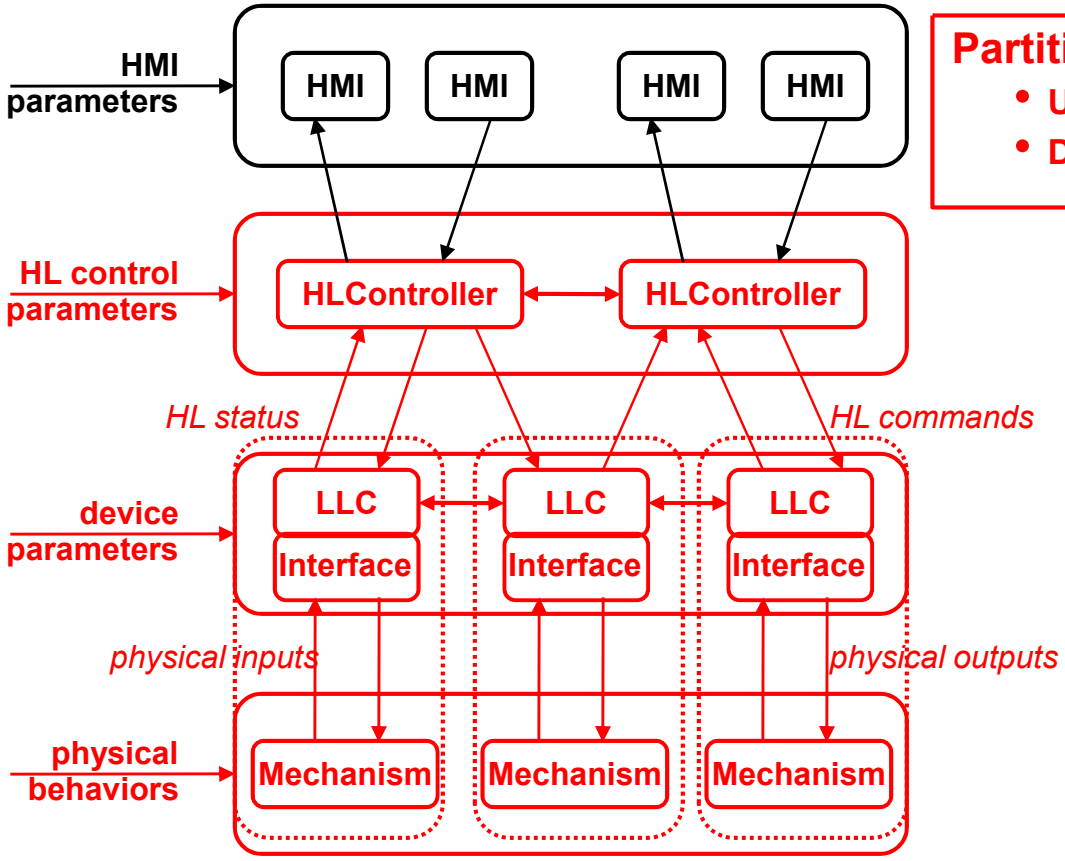


Example: mach/DRILL_MVCDR

Layered MVC: Physical Design



Design Pattern: Mechatronic



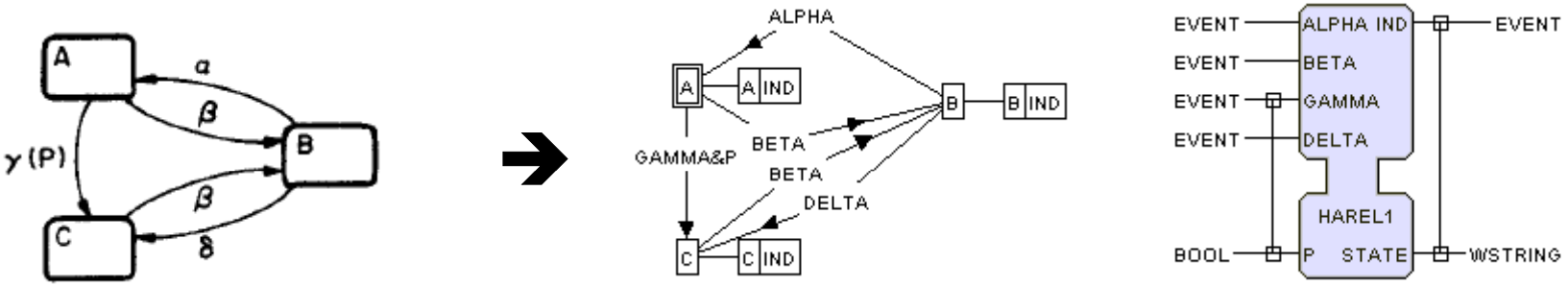
Partition control/diagnostic functions to:

- **Use existing mechatronic devices**
- **Design new mechatronic devices**

HL = High Level
LL = Low Level

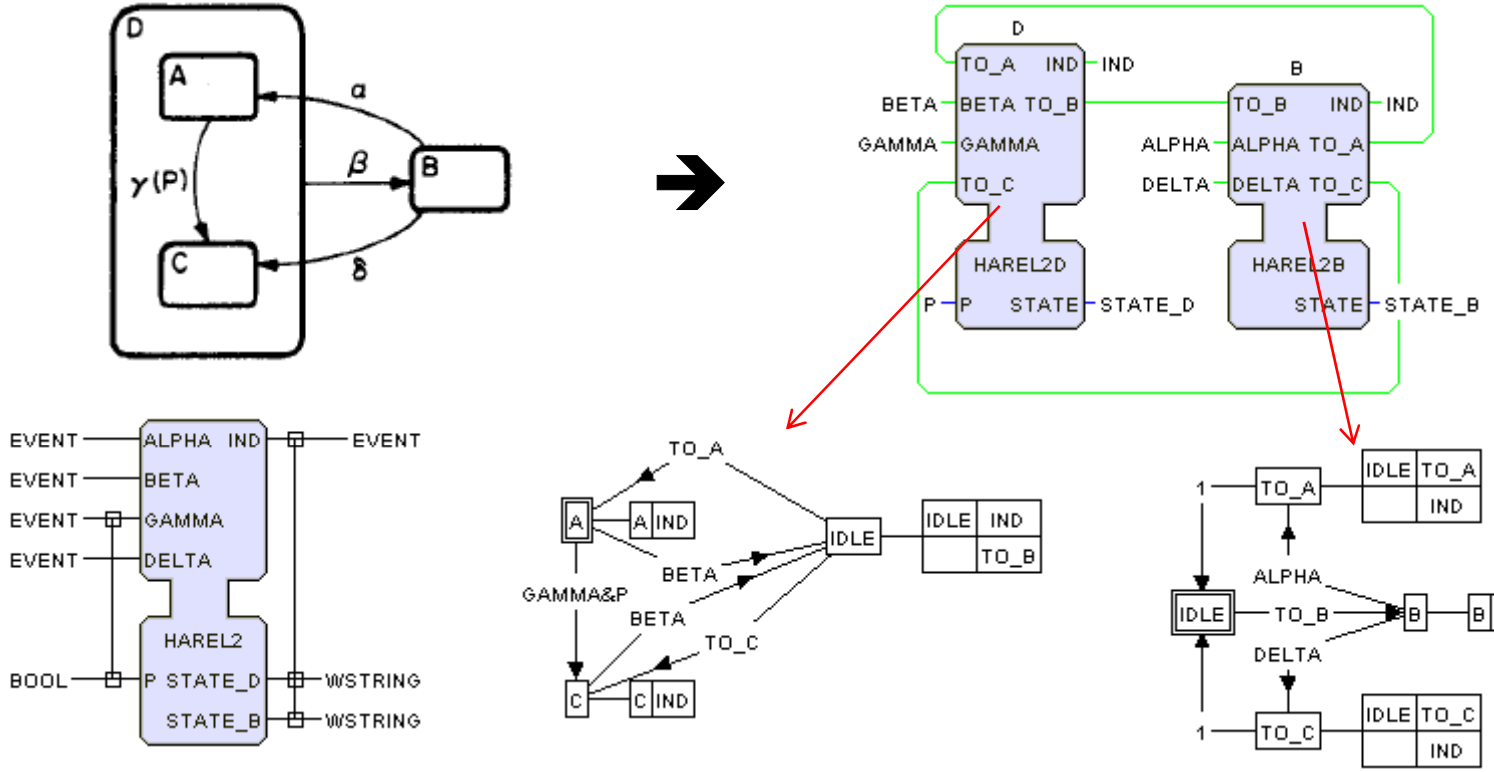
Mechatronic Element

Harel Statechart Mapping(1): Simple State Chart



Source: D.Harel, "Statecharts: A Visual Formalism for Complex Systems," *Science of Computer Programming,* 8 (1987), 231-274.

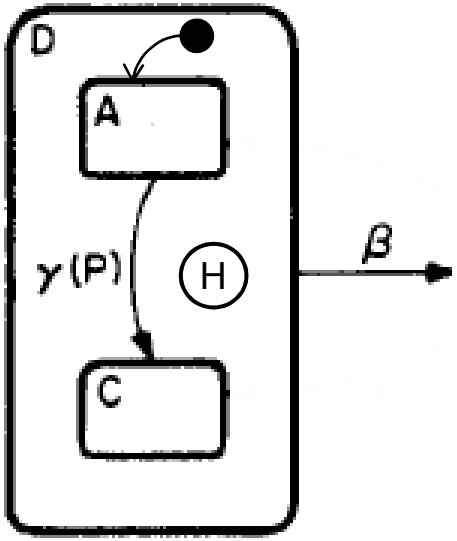
Harel Statechart Mapping (2): Clustering

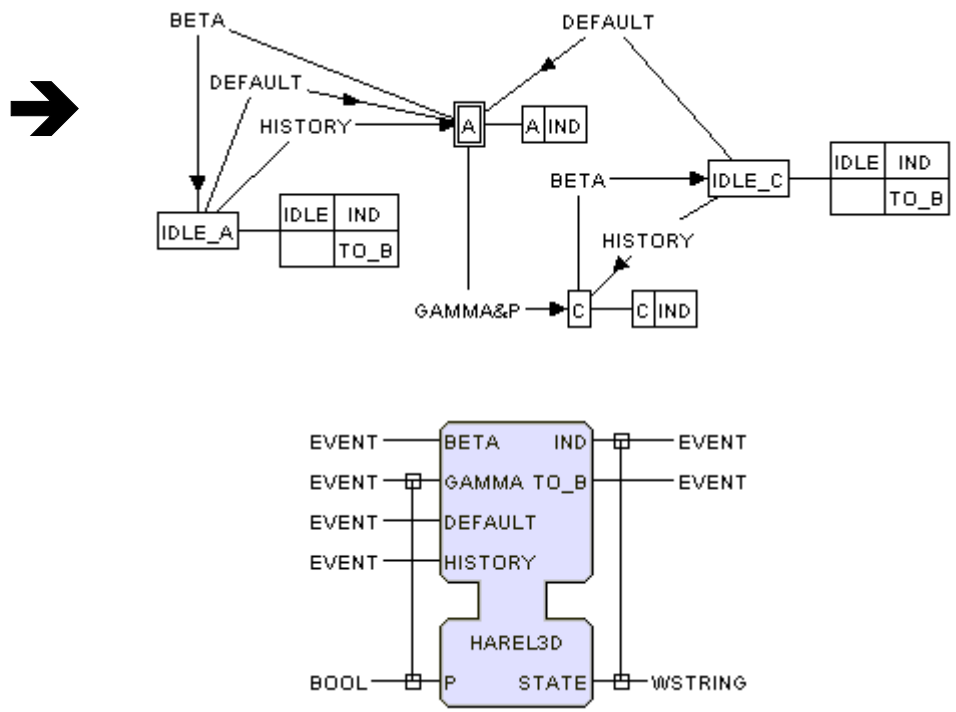


- Implementation of B can change without affecting D and vice versa.
- Any state can be implemented as default return state. ( )

Source: D.Harel, "Statecharts: A Visual Formalism for Complex Systems," *Science of Computer Programming,* 8 (1987), 231-274.

Harel Statechart Mapping (3): Default Return and History Node





- History requires an EC idle state per main state.
- Default requires an extra transition per idle state.

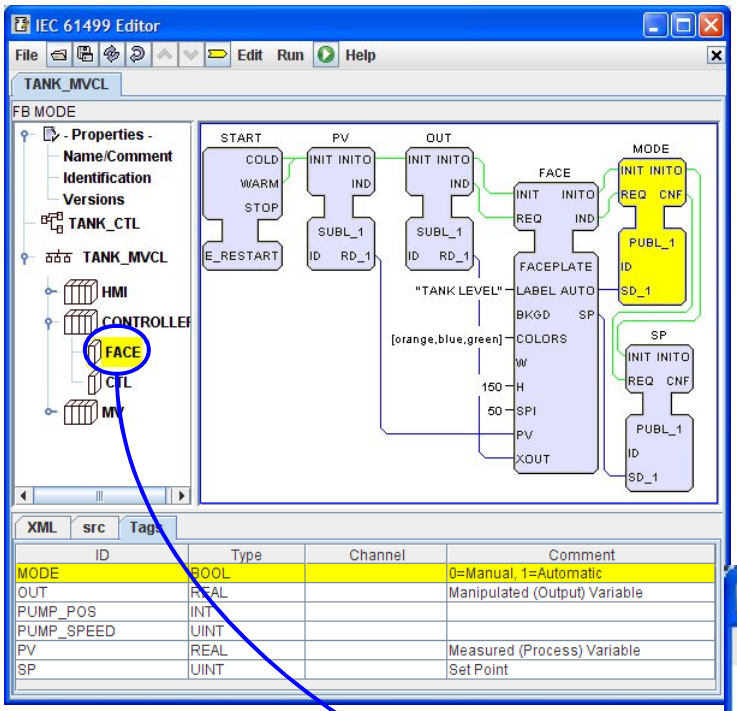
Source: D.Harel, "Statecharts: A Visual Formalism for Complex Systems," *Science of Computer Programming,* 8 (1987), 231-274.

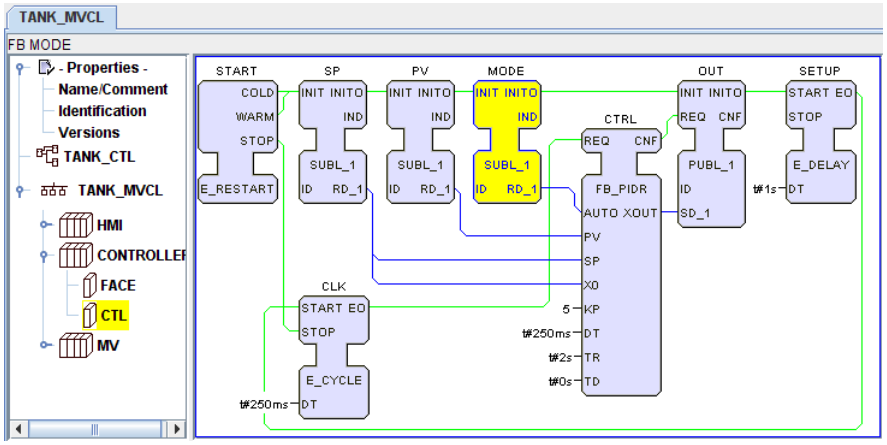
Modes: Basic Concepts

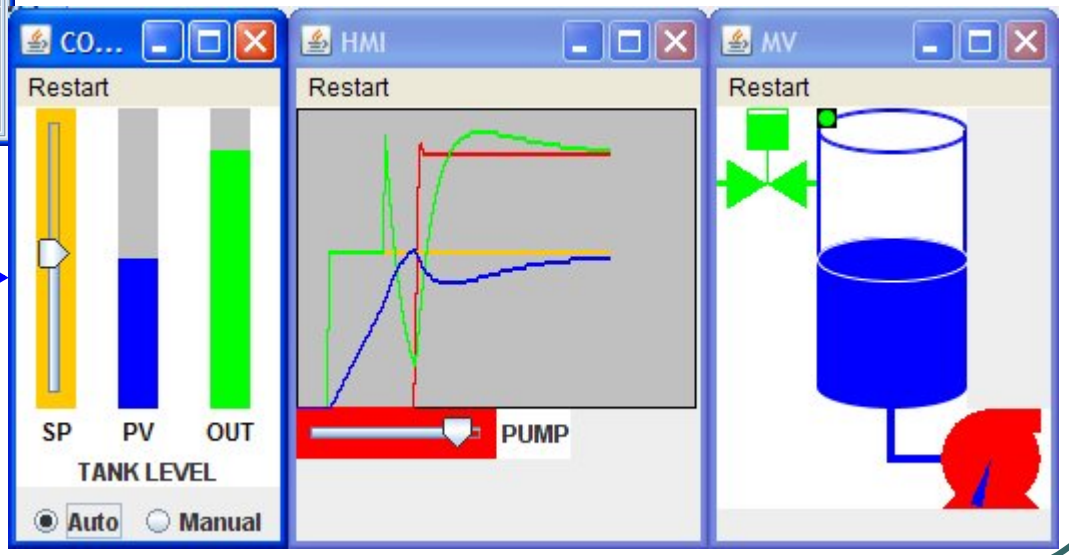
- ISA S88 “uses as examples three modes …. *Control modules* … will have *automatic* and *manual* modes, whereas a unit running *procedural control* would also have a *semi-automatic* mode.”
- “**A mode determines how equipment entities and procedural elements respond to commands and how they operate**.
- “In the case of procedural elements, the mode determines the way the procedure will progress and who can affect that progression.
- “In the case of a control module, such as an automatic block valve …, the mode determines the mechanism used to drive the valve position and who/what, such as another device or an operator, may manipulate it to change its state.
- “Equipment entities or procedural elements may change mode.
- “ **A change of mode in one [entity or element] may cause corresponding changes in other [entities or elements]**.”

Source: ANSI/ISA–88.01–1995, *Batch Control, Part 1: Models and Terminology*.

Multicast Mode: An Example

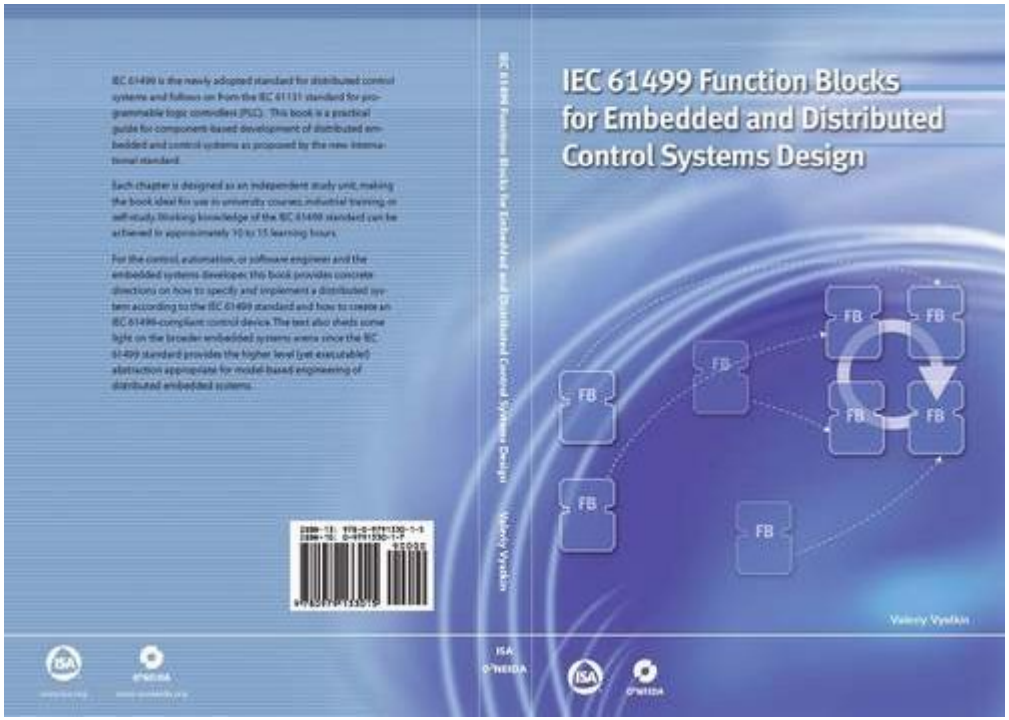






IEC 61499: Book

- ISBN 978-0-9792343-0-9
- Available at http://www.isa.org
- Learning exercises based on FBDK



Adding Value with IEC 61499

- Background
- Requirements
- Architecture
- Software Tools
- Design Patterns & Frameworks
- **Runtime Platforms**

IEC 61499-4: Rules for Compliance Profiles

- **Contents of compliance profiles**
  1. Scope
  2. Normative References
  3. Terms and Definitions
  4. Portability provisions
  5. Interoperability provisions
  6. Configurability provisions
  7. Test requirements
- **Annexes** (may include device configurability classes)
- **Example compliance profile**
  - See *http://www.holobloc.com/doc/ita/index.htm*
  - Originally developed for HMS feasibility demo (2000)

IEC 61499-4, Annex B: Example Compliance Classes

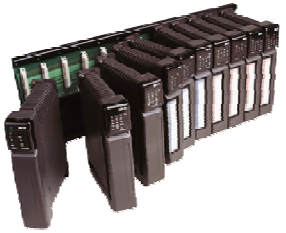
- **Class 0**: Simple Devices
  - Read & Write Parameter Values
  - Start & Stop FBs & Applications
  - Query for Supported Data Types & FB Types
- **Class 1**: Simple Programmable Devices
  - Class 0 plus:
    - Create & Delete FB instances & connections
    - Query FB instances & Applications
- **Class 2**: User Reprogrammable Devices
  - Class 1 plus:
    - Create & Delete Data Types & FB Types
    - Query for specific data types & instances of FB types

IEC 61499: Hardware Platforms

- HOLOCON
  - Compatible with FBDK
  - http://www.wrcakron.com/holocon.html

- Elsist NetMaster II
  - Has hosted FBRTin several laboratory applications
  - http://www.elsist.it/WebSite/Html/English/Products/Hardware/Netsyst/EnNetmasterII.php

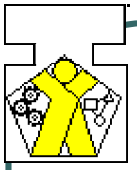
- Kingfisher PLUS+ RTU
  - Compatible with ISaGRAF
  - http://www.rtunet.com/products/kingfisher-plus/



- Beck IPC@CHIP@
  - Compatible with ISaGRAF
  - http://www.isagraf.com/pages/news/pr_beck_apr2007.htm

Conclusions

- IEC 61499 provides a standardized architecture for encapsulation, reuse and deployment of IP in automation and control systems.
- Compliance profiles, design patterns, frameworks and methodologies are rapidly maturing.
- Industrial adoption will depend on availability of commercially supported software tools and runtime platforms.



Copies of this presentation

Available at

http://www.holobloc.com

Copies of this presentation

Available at

http://www.holobloc.com