A Communication Architecture to Observe and Partially Preserve Efficiency in Automated Production Systems

Source:

J. Wilch, B. Vogel-Heuser, Y. Hsieh and F. Cheng. "A Communication Architecture to Observe and Partially Preserve Efficiency in Automated Production Systems," in *18th International Conference on Automation Science and Engineering (CASE)*, IEEE, Aug. 2022.

18th International Conference on Automation Science and Engineering

August 20 – August 24, 2022, Mexico City, Mexico

Institute of Automation and Information Systems
TUM School of Engineering and Design
Technical University of Munich

Concept: Observation and Compensation in aPS

- **Observation**: Early detection of threats to OEE (availability)
  - Record process data; extract features, detect trends
- **Knowledge**: Relate and contextualize data from distributed sources
- **Compensation**: Trigger reactions to avoid availability threats

OEE =

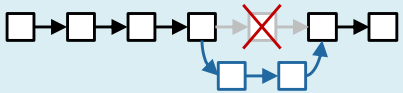
Availability

x Performance

x Quality

Process Observation
Detect risks to availability

Compensation
Behavior changes at runtime to maintain availability



Institute of Automation and Information Systems
TUM School of Engineering and Design
Technical University of Munich

Concept: Distributed Computation from Field to Cloud

aPS / PLC

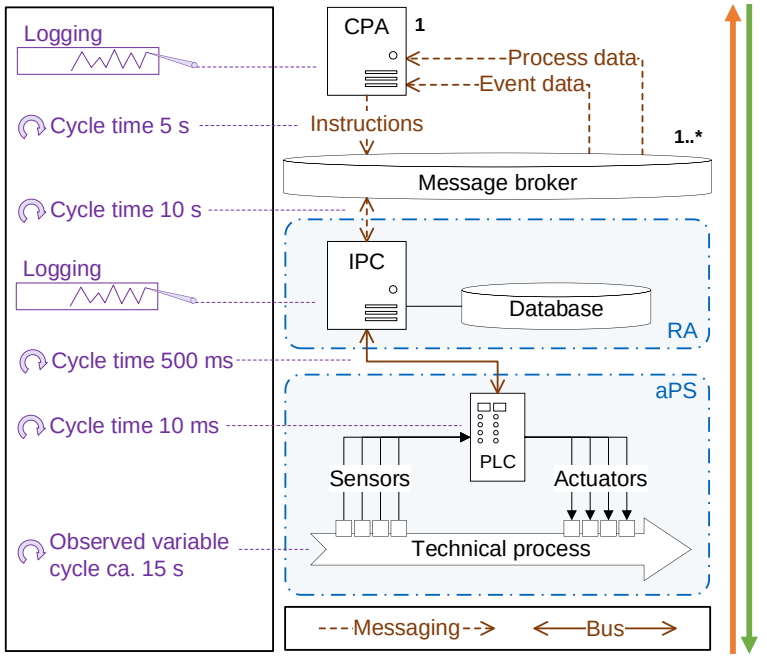
- Real-time execution of the application program
- Sampling of process values; stored in temporary variables
- Change behavior based on global variable value

Resource Agent (RA)

- Sampling PLC's process value variables
- Transmit values to buffered message queue
- Receive and execute changes to PLC's global variables

Cyber-physical Agent (CPA)

- Communicate with <u>one or many</u> RAs via message queues
- Detect value degradations and select suitable instruction



Institute of Automation and Information Systems
TUM School of Engineering and Design
Technical University of Munich

Evaluation: Communication Architecture Prototype

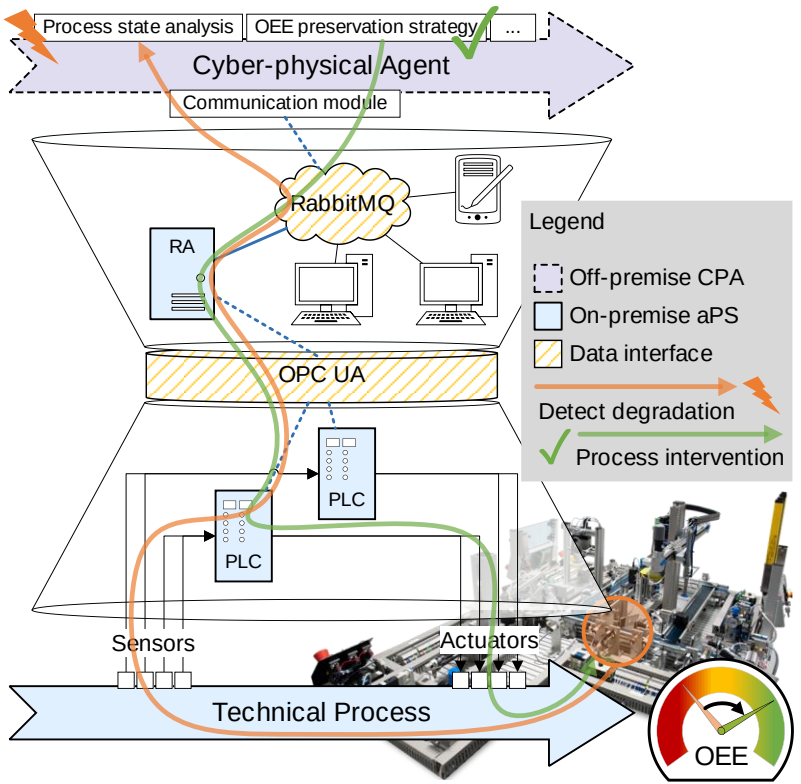
Communication Protocols

- PLC *(Beckhoff CX9020; 1 core, 1 GHz)* ⇔ RA *(1 core, 1.86 GHz)* via OPC UA; PLC as OPC UA server
- RA ⇔ CPA *(4 cores, 1.3 GHz)* via RabbitMQ over AMQP

Cycle Times

- PLC samples process every 10 ms
- RA ⇔ PLC every 500 ms
- RA ⇔ message broker every 10 s
- CPA ⇔ message broker every 5 s

➢ Worst-case reaction time = 31.02 s
(compute times at each level are negligible)



Institute of Automation and Information Systems
TUM School of Engineering and Design
Technical University of Munich

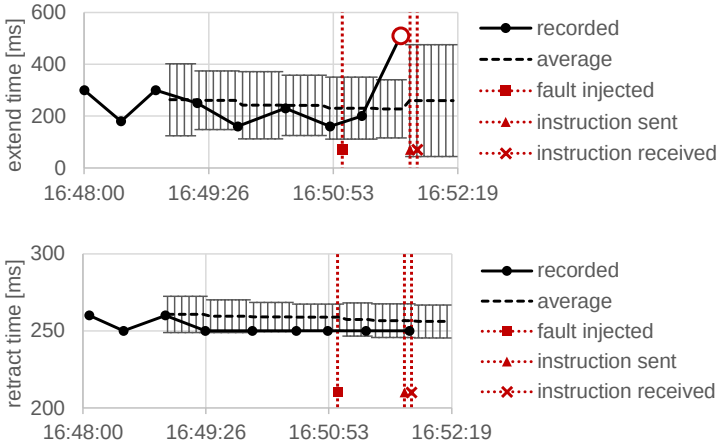
Evaluation: Investigation of Runtime Behavior

- Get timestamped extend/retract times (PLC → RA → CPA)
- Compute running average and standard deviation (CPA)
- Trigger compensation, if average ± two standard deviations is exceeded (CPA → RA → PLC)

- A **fault can be injected manually** (cylinder delay)

Timely Behavior

- Timestamps based on synchronized clocks (PLC, RA, CPA)
- Sampling delay causes shifts between process value changes and CPA reactions



Summary and Outlook

Distribution of Computations over the Hierarchy

- Currently, only vertical distribution (1 PLC, 1 RA, 1 CPA)
- Future work: larger horizontal setups (n PLCs with n RAs)

Detection of Threats to Efficiency

- Currently, simple observation of average and standard deviation
- Future work: Remaining Useful Life, closed-loop control, etc.

Injection of Compensation Strategies

- Currently, a preconfigured strategy is activated on demand
- Future work: Generate strategies based on knowledge of the PLC's behavior capabilities