



A generic software framework for role-based Organic Computing systems

Jan-Philipp Steghöfer, Florian Nafz, Hella Seebach, Frank Ortmeier, Wolfgang Reif



- Dedicated research effort in Germany
 - Priority program of the German Research Foundation
 - 18 teams working in different areas: sensor nets, software engineering, embedded systems, swarms, etc.
- Trying to coin a new term encompassing
 - Autonomic Computing
 - Bio-inspired algorithms
 - Self-Organizing and Self-Adaptive Systems

- Embedded, software intensive applications that are
 - particularly resistant against disturbances and component failures (w.r.t. functional correctness, safety, security)
 - adaptive to changing requirements and modified tasks
- Agent- and role-based systems
 - Each agent has several capabilities
 - Resources with different tasks are processed by the agents
 - Each task needs different processing steps
 - Processing steps are a given sequence of capabilities
 - Roles define which capabilities the agent applies

- Goal: Framework for the design and construction of highly reliable Organic Computing applications
 - Top-Down Design Methodology
 - Integrated Software Development Process
 - Formalization of self-x properties
 - Formal analysis and verification to enable behavioural guarantees
 - Extensible generic runtime environment (today)

SAVE ORCA – Adaptive Production Cell

Goal: Process workpieces in a given order!



Unprocessed
workpieces

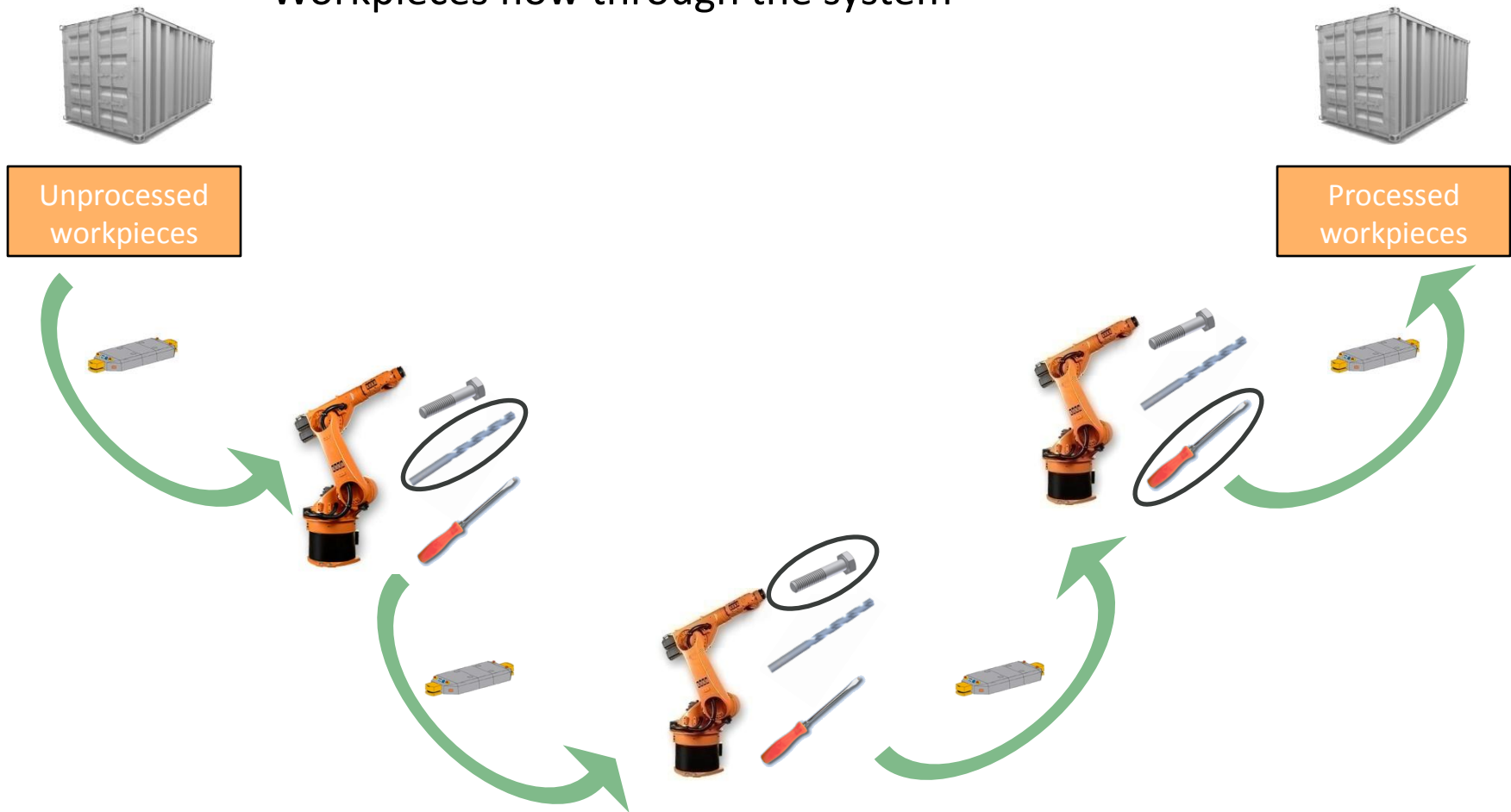


Processed
workpieces



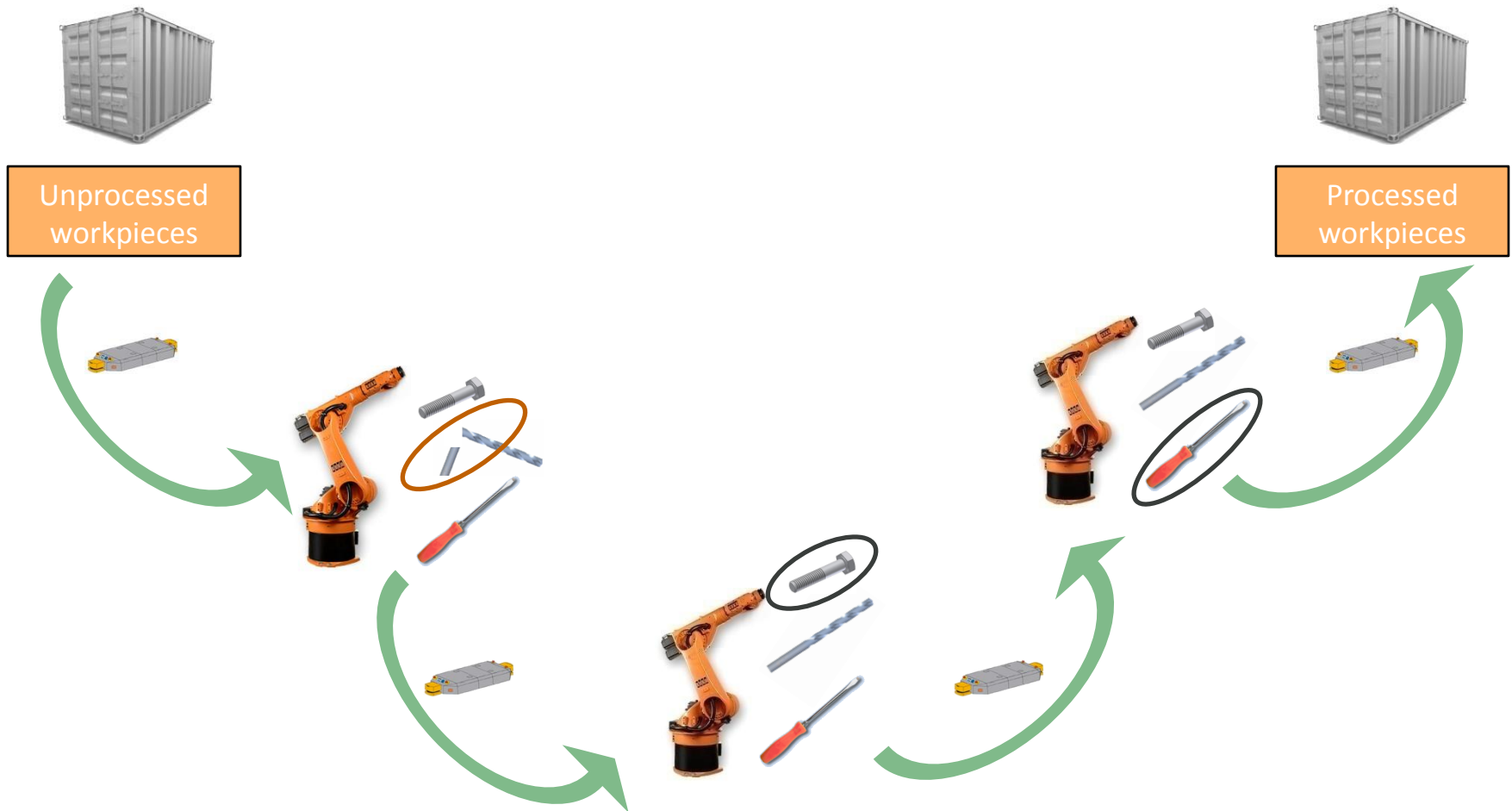
SAVE ORCA – Normal Operation

Normal Operation:
Workpieces flow through the system



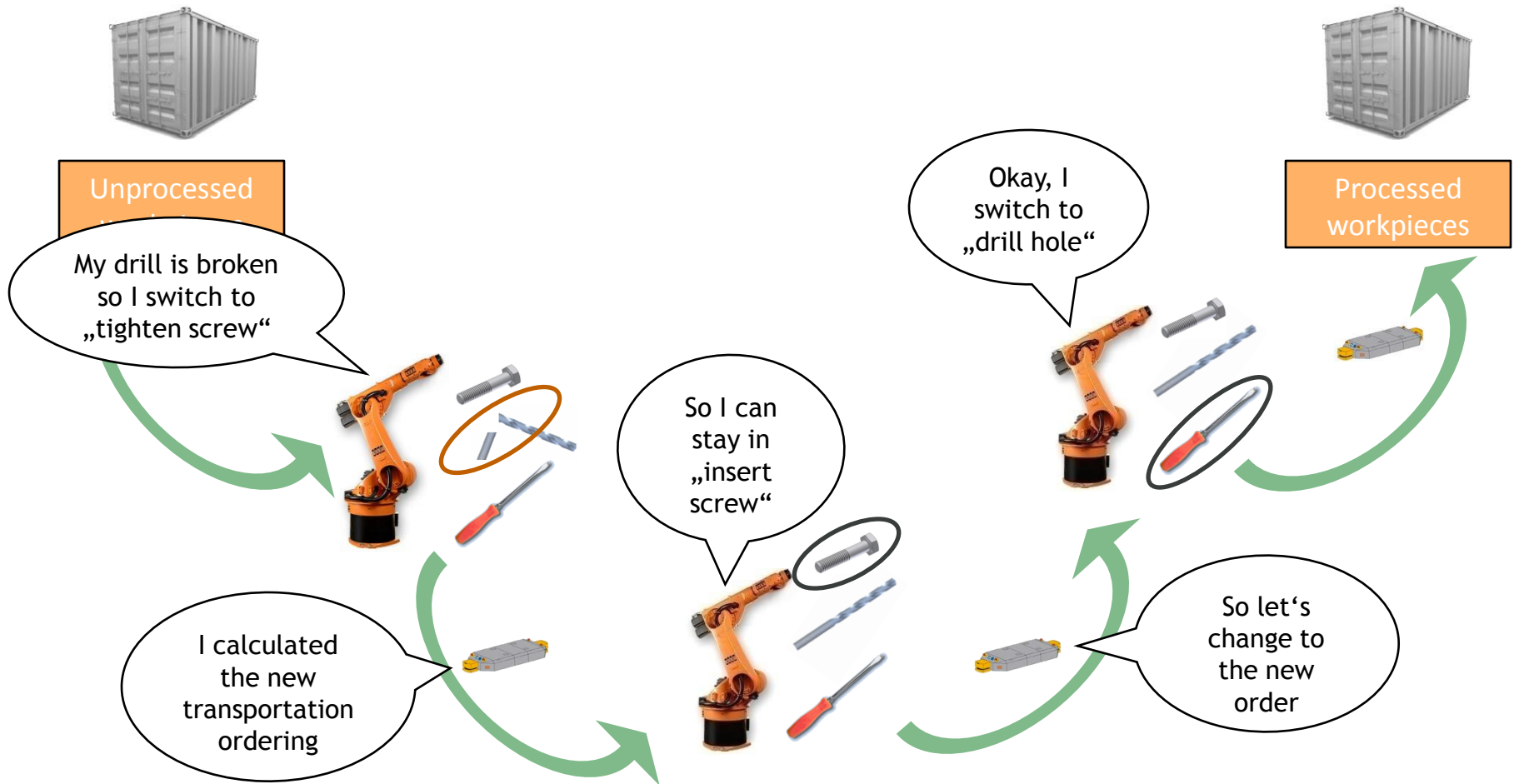
SAVE ORCA – Self-Healing

Self-Healing: Resistance to component failures



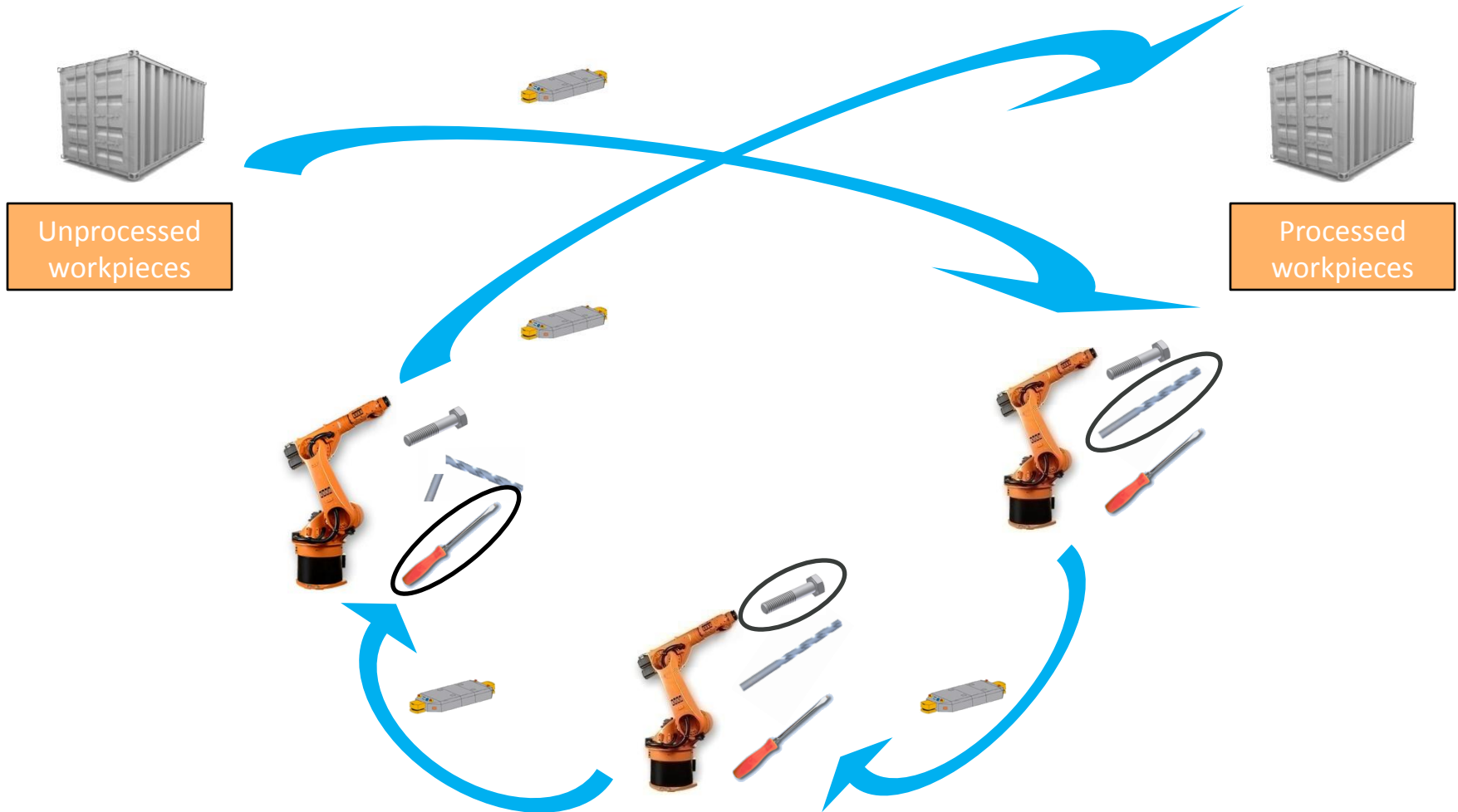
SAVE ORCA – Self-Healing

Self-Healing: Resistance to component failures



SAVE ORCA – Self-Healing

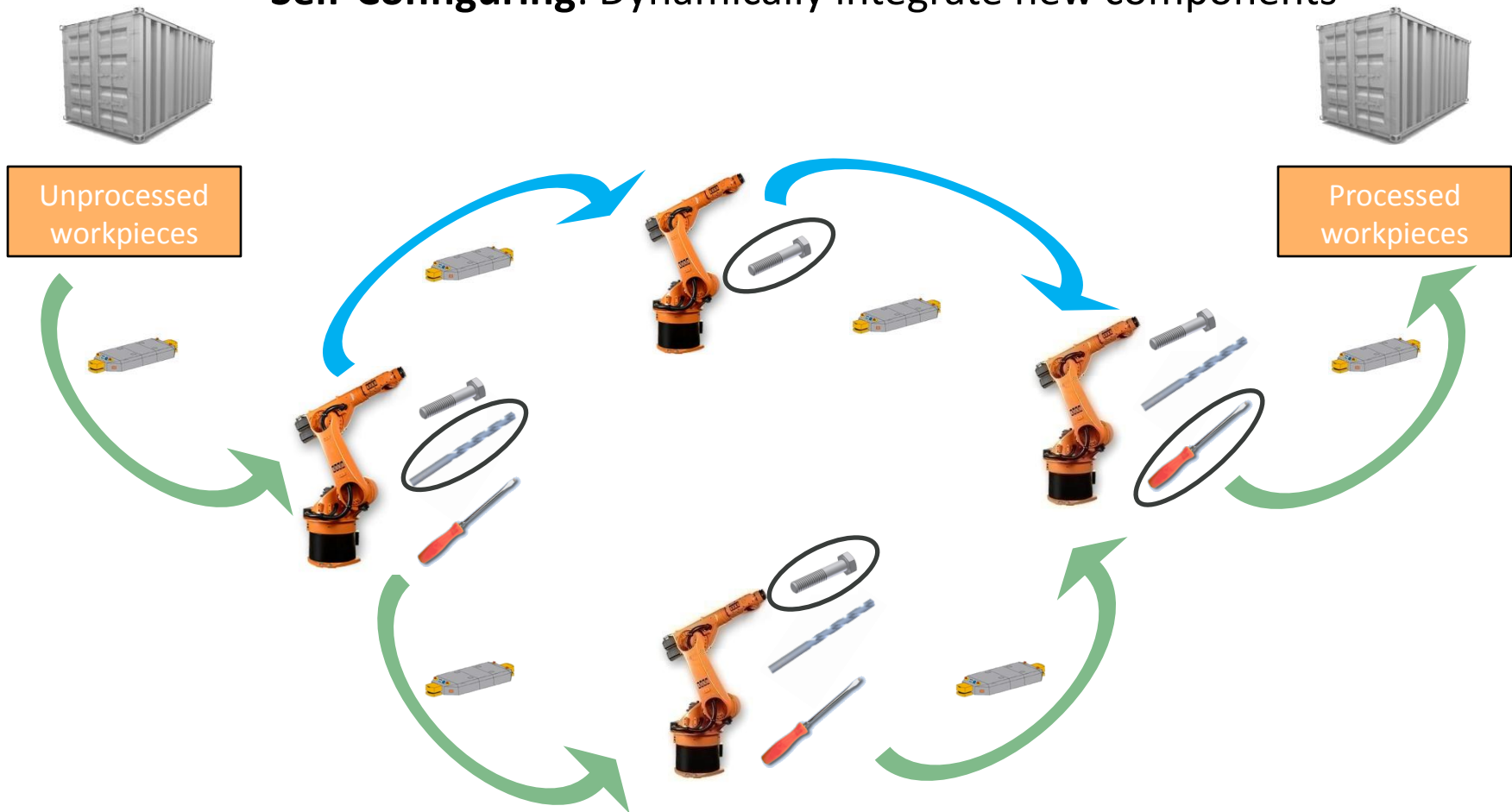
Self-Healing: Resistance to component failures



SAVE ORCA – Self-Configuring

Self-Healing: Resistance to component failures

Self-Configuring: Dynamically integrate new components

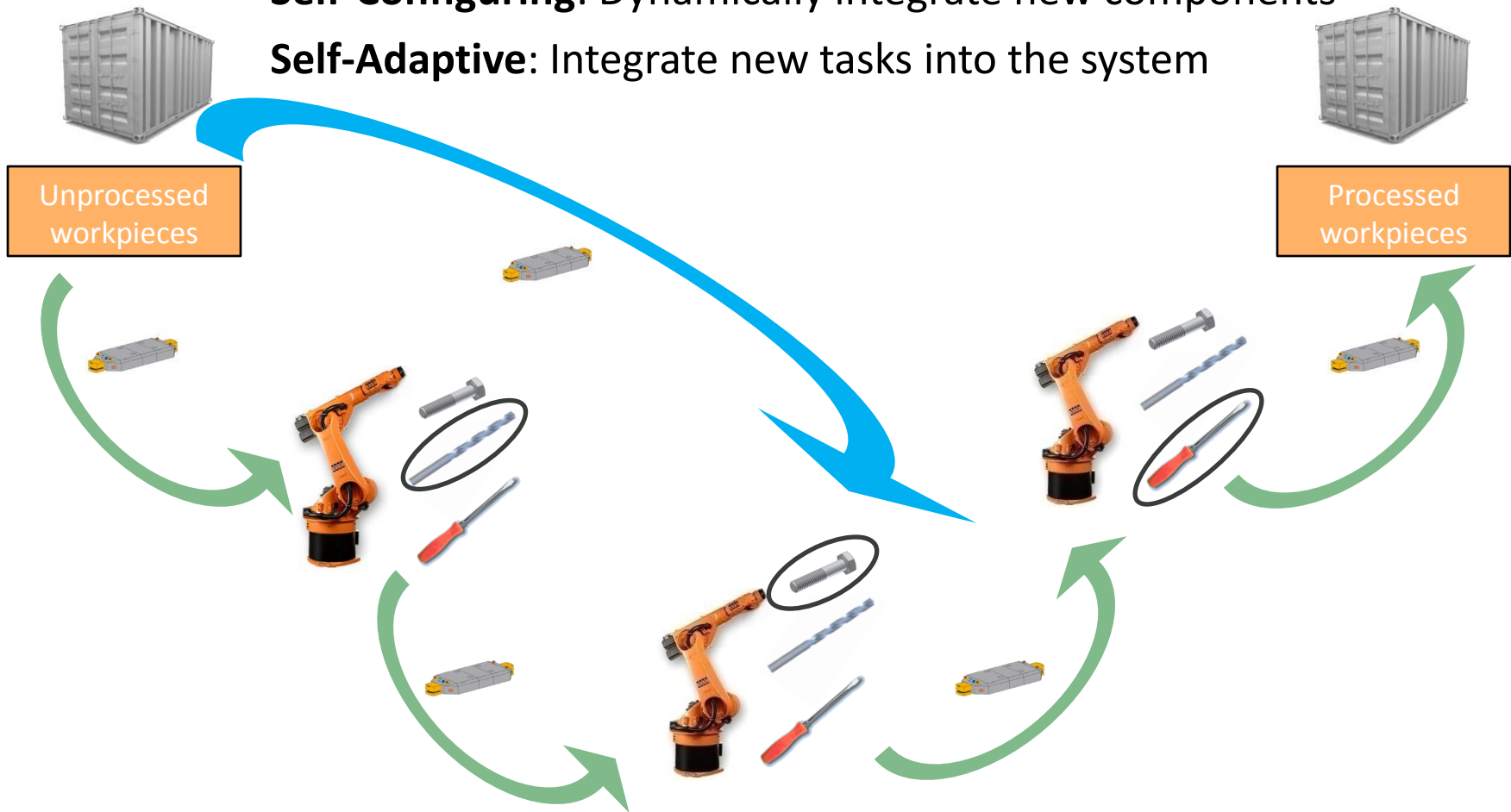


SAVE ORCA – Self-Adaptive

Self-Healing: Resistance to component failures

Self-Configuring: Dynamically integrate new components

Self-Adaptive: Integrate new tasks into the system



SAVE ORCA – Self-Optimizing

Self-Healing: Resistance to component failures

Self-Configuring: Dynamically integrate new components

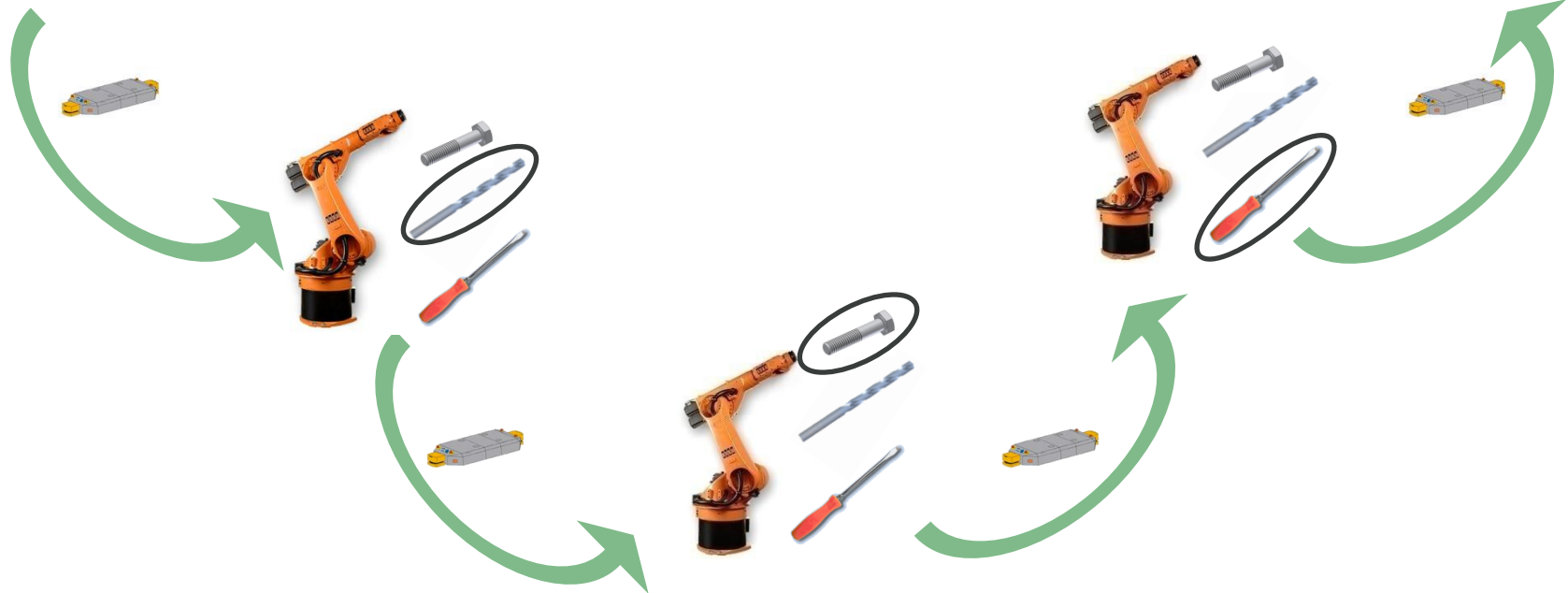
Self-Adaptive: Integrate new tasks into the system

Self-Optimizing: Improve configuration fitness



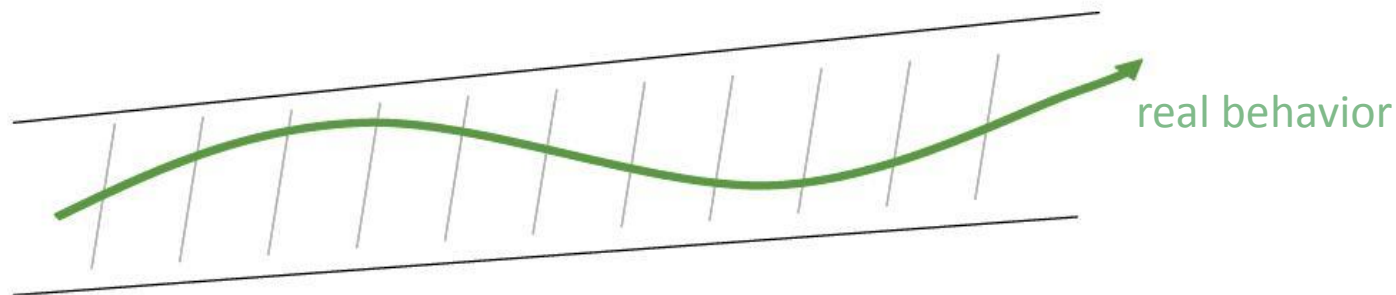
Unprocessed
workpieces

Processed
workpieces

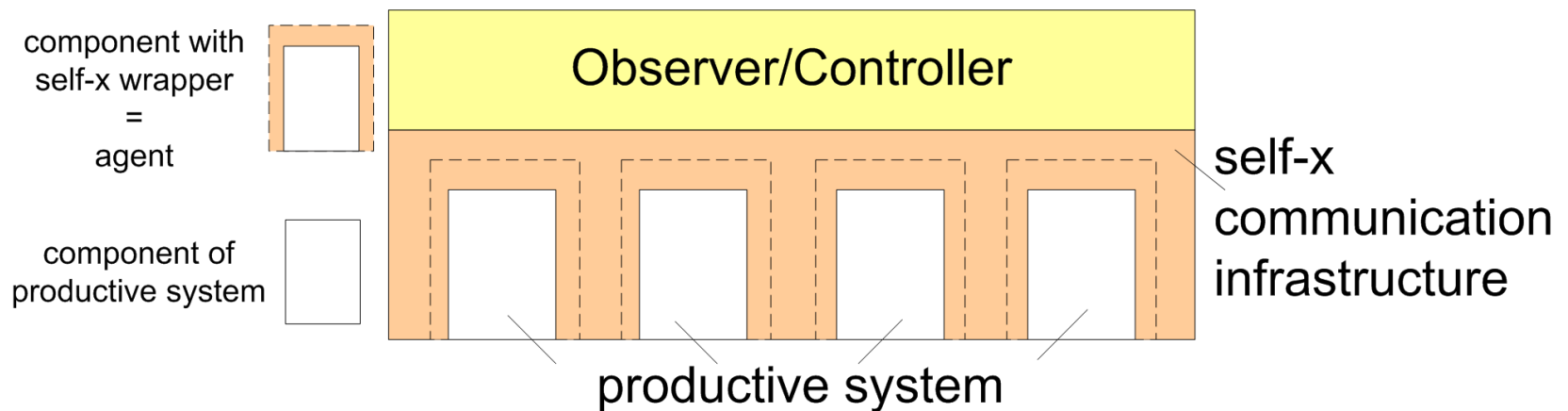


SAVE ORCA – Restore Invariant Approach

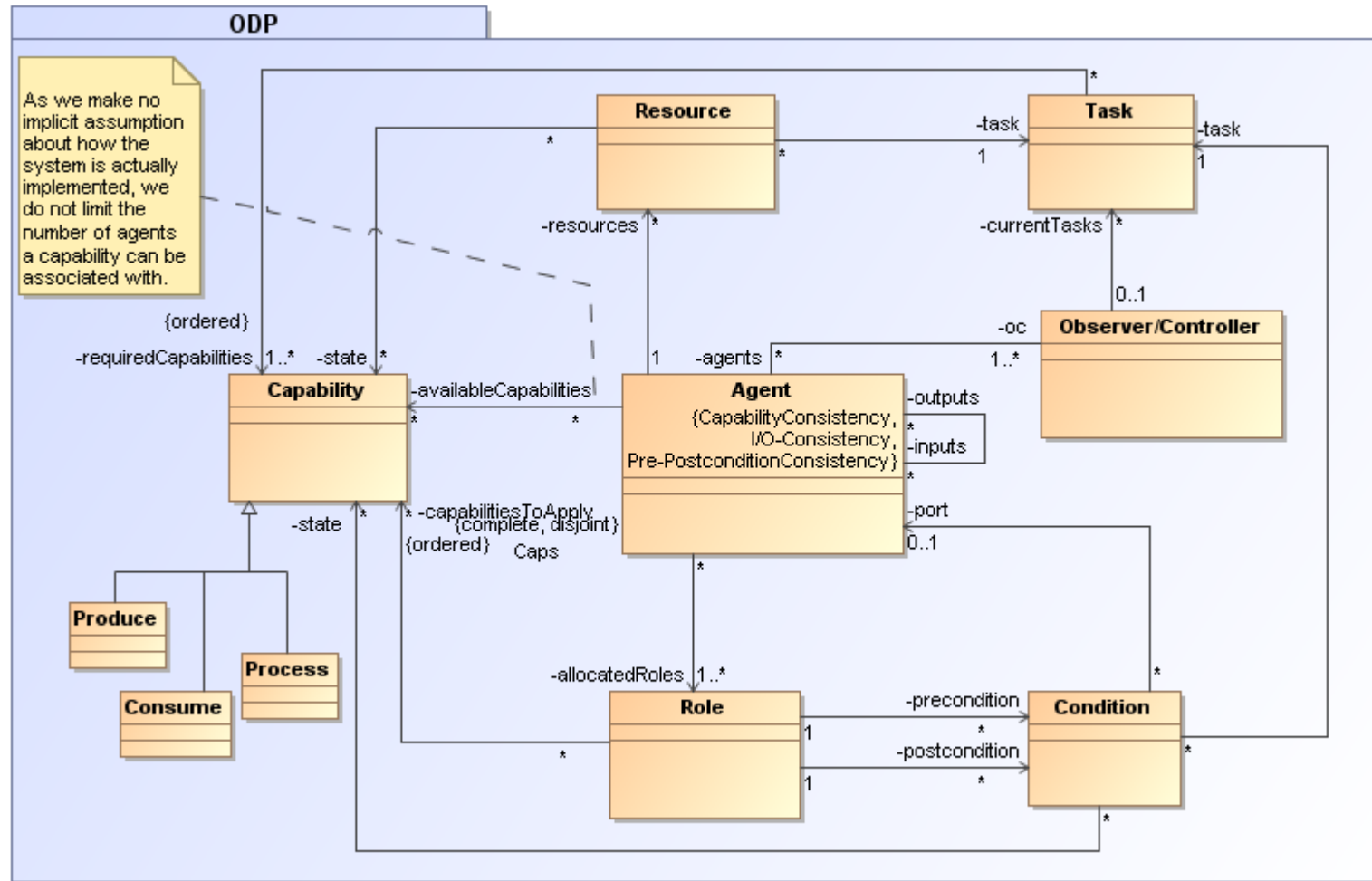
- Formally model a system's intended behaviour with invariants
- Invariants establish corridor of acceptable behaviour
- Whenever system leaves corridor, reconfigure to restore invariants



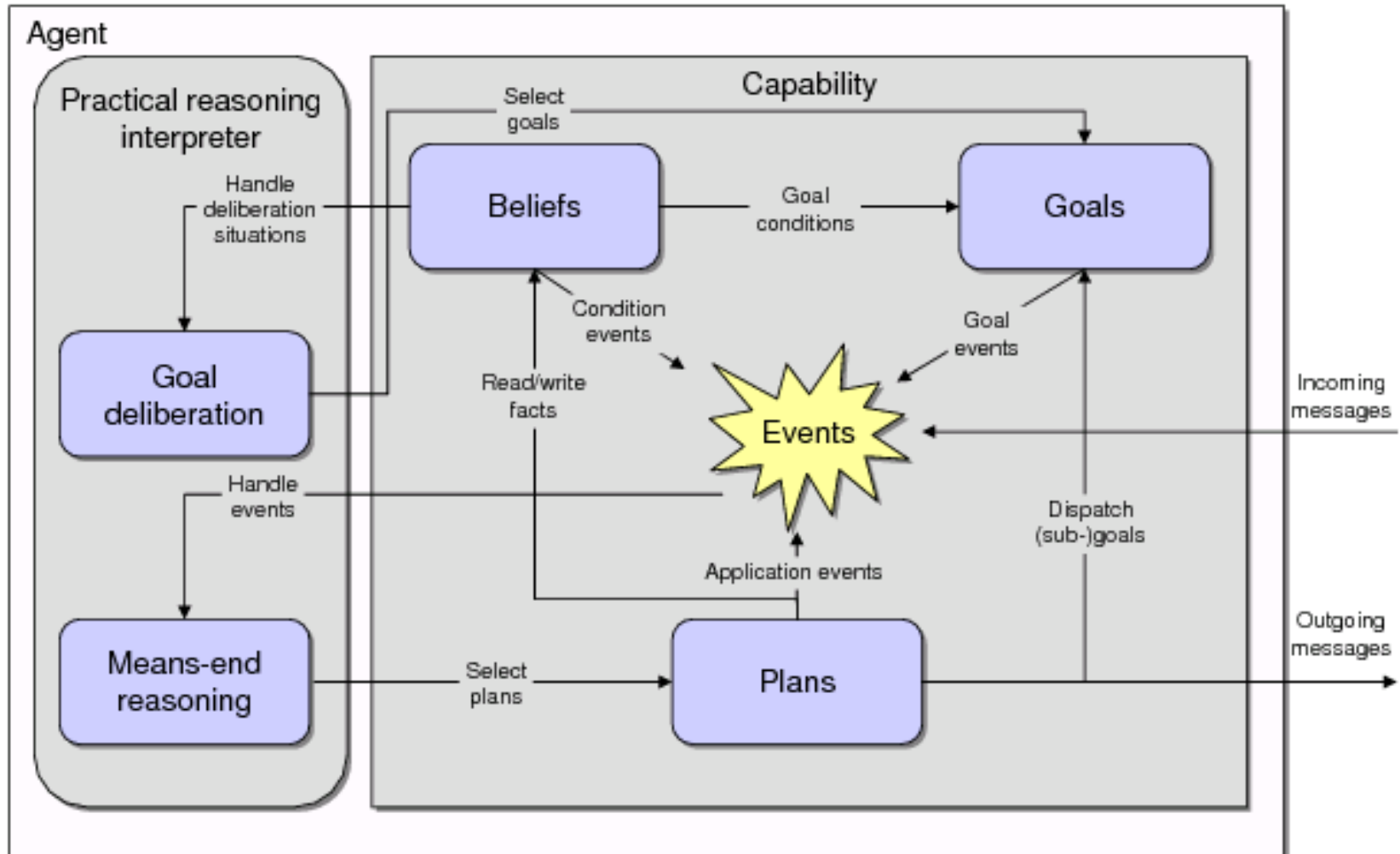
- Invariants are observed at runtime by centralized or distributed Observer/Controller
- Reconfiguration can be formulated as a constraint satisfaction problem



SAVE ORCA – Organic Design Pattern



Jadex BDI Agent System



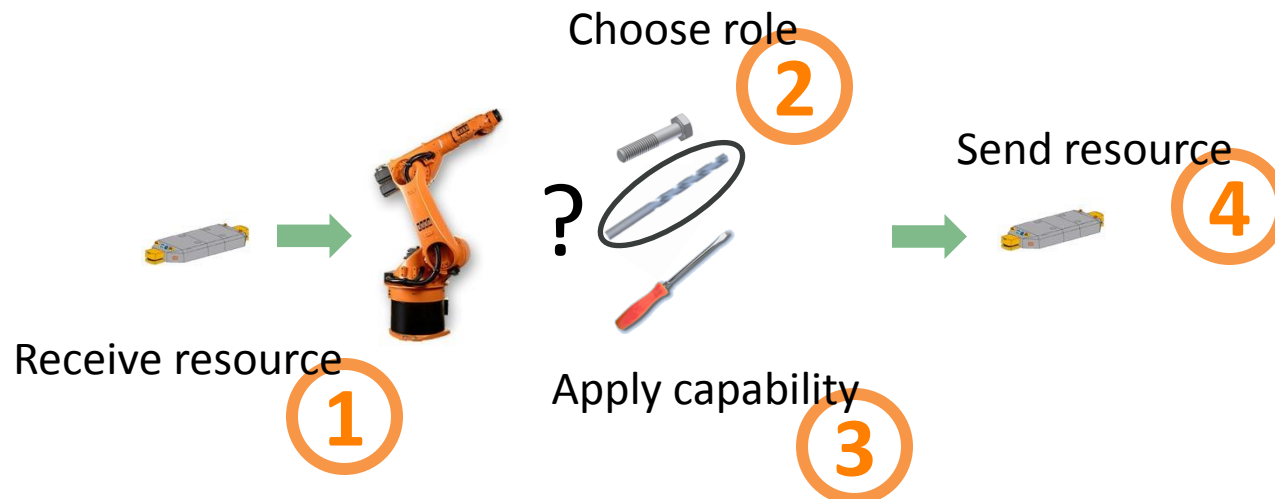
Graphic from [Jadex User Guide]

- ODP concepts can be mapped to Jadex concepts
- Generic infrastructure is provided as Jadex capabilities:
 - Communication
 - Role selection and execution
 - Reconfiguration
 - Data models and messages
- Domain and application-specific extension points

ODP	Jadex
Agent	Agent
Observer/Controller	Agent
Capability	Plan
Role	Belief
Condition	Belief
Resource	Belief
Task	
State	

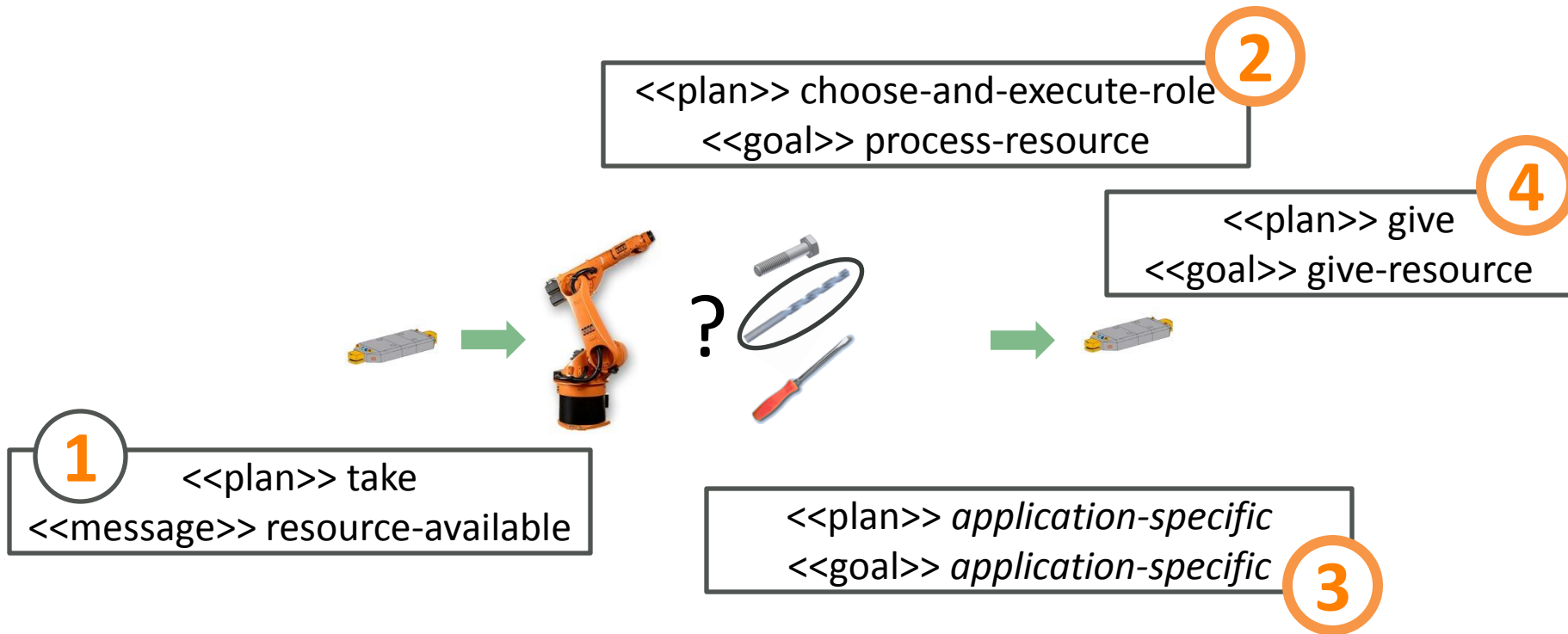
Generic role execution

Role: Precondition (Input agent, resource state)
Capability to apply
Postcondition (new resource state, output agent)

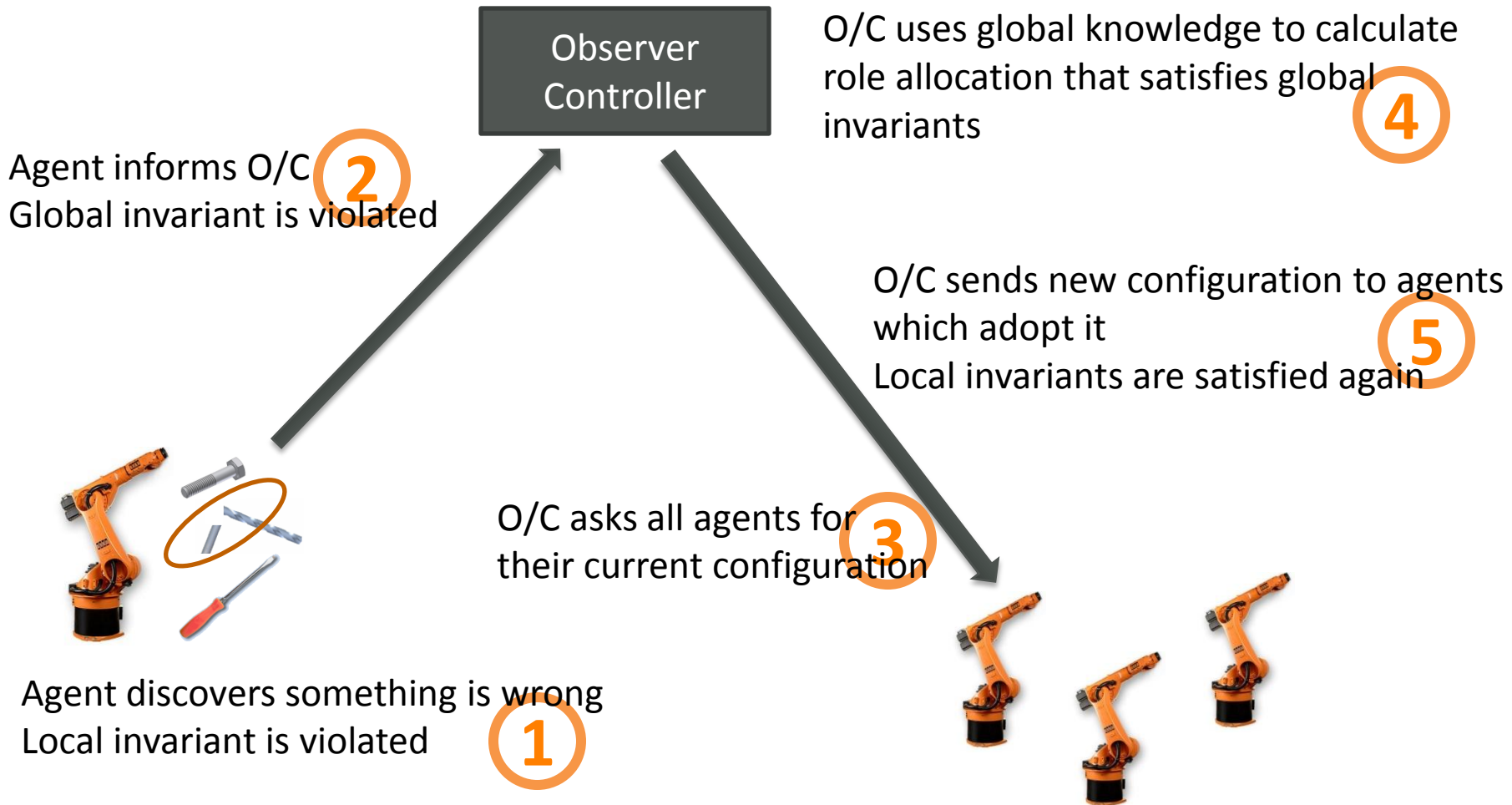


Generic role execution

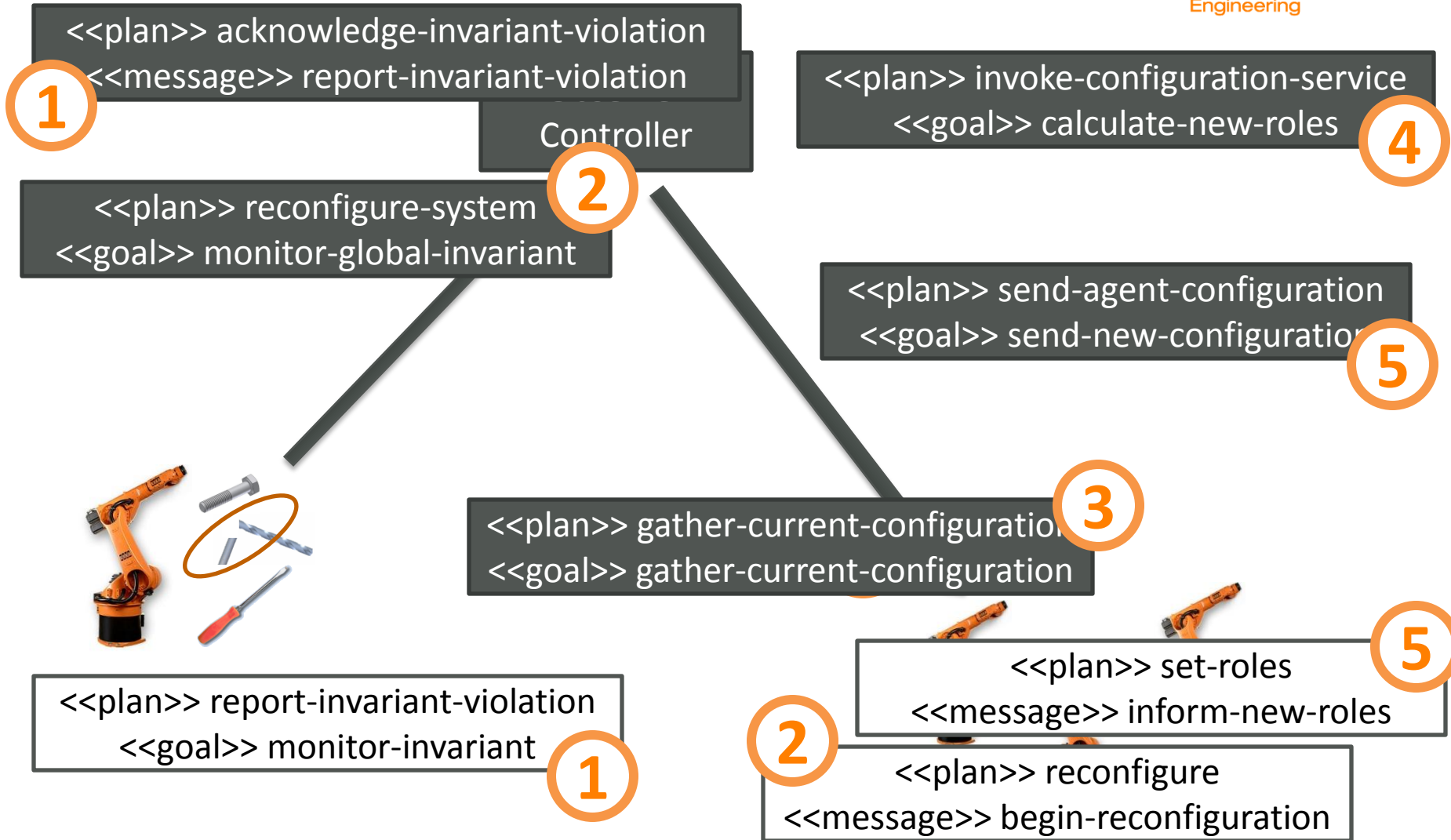
Role: Precondition (Input agent, resource state)
Capability to apply
Postcondition (new resource state, output agent)



ODP's reconfiguration process



ODP reconfiguration process



- **Application of capabilities:** What should the agents do with the workpieces
- **Domain-specific invariants:** Specify the behavioural corridor of the application domain
- **Reconfiguration algorithm:** Optimize the algorithm for the domain or application
- **Resource Flow:** Adapt the framework to specific resource handling requirements

- SAVE ORCA provides guideline and process for reliable, secure Organic Computing systems
- ODP Runtime Environment provides the means for short design-implementation turnaround times
- Many features implemented in a generic fashion
- Clearly defined extension points
- Built on top of a reliable and well-documented platform