TESTING & VERIFICATION OF PLC CODE FOR PROCESS CONTROL

B. Fernández#, E. Blanco, A. Merezhin, CERN, Geneva, Switzerland

ABSTRACT

Functional testing of PLC programs has been historically a challenging task for control systems engineers. This paper presents the **analysis** of different mechanisms for testing PLC programs developed within the **UNICOS** (UNified Industrial COntrol System) framework. The framework holds a **library of objects**, which are represented as Function Blocks in the PLC application. When a new object is added to the library or a correction of an existing one is required, exhaustive validation of the PLC code is needed. **Testing and formal verification** are two distinct approaches selected for eliminating failures of UNICOS objects. The **advantages and limitations** of both approaches are presented and illustrated with a case study, validating a specific UNICOS object.

UNICOS generalities

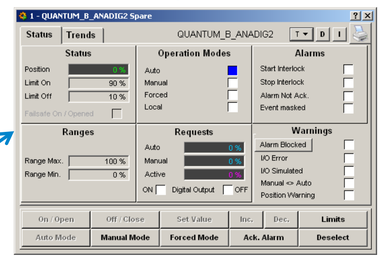
- Framework based on objects.
- Control engineers build a **model** of the process units using **UNICOS objects**.
- Objects represented at the **SCADA** and **Control** layer.

UNICOS LIBRARY

UNICOS objects

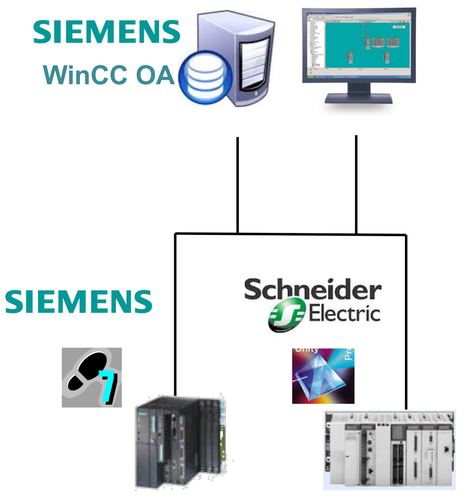
UNICOS object Design

UNICOS object in WinCC OA



ST PLC code

```
…
IF c THEN
   s1;
ELSE
   s2;
…
```

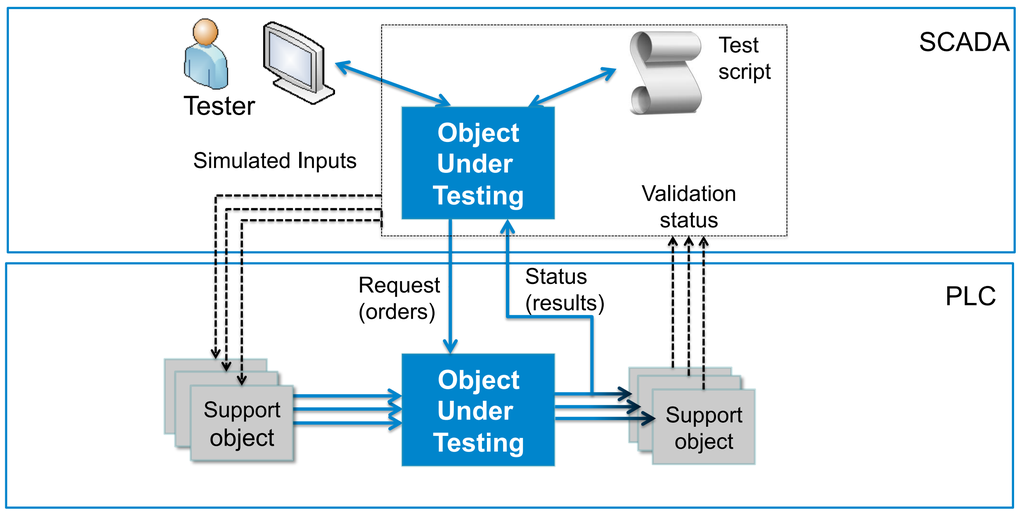


Goal

Analyse different approaches to **test and verify UNICOS PLC objects**

Testing

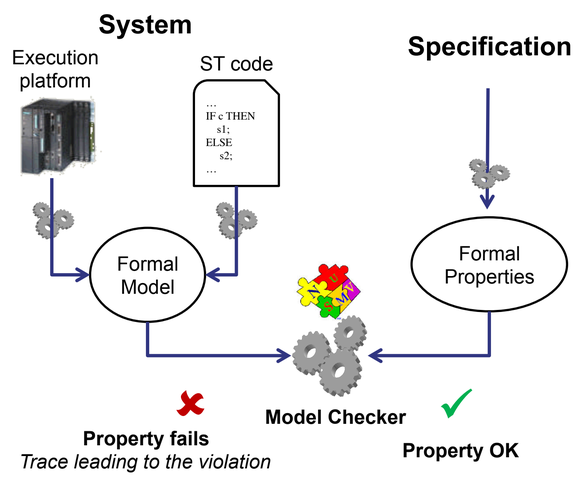
- Applied to a **real system.**
- Based on a **test catalog.**
- Manual testing can be **replaced** by automatic testing.
- **Automatic** testing with WinCC OA Control language scripts:
  - Send orders to PLC.
  - Compare retrieved result in WinCC OA.
  - Report results.

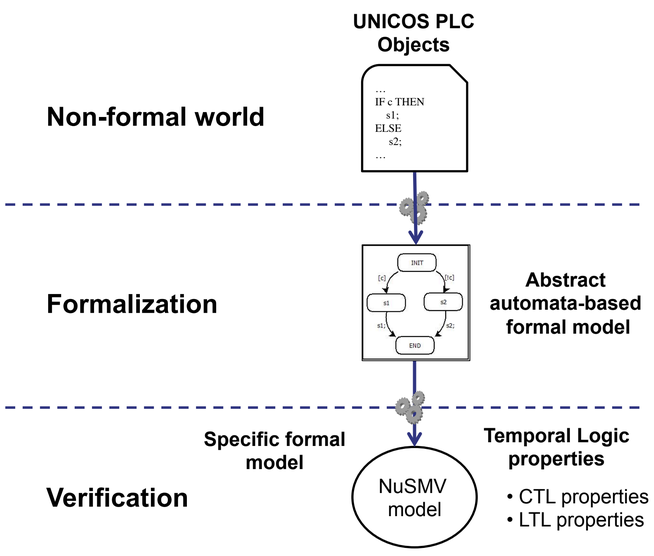


Formal Verification

Concepts

- **Model Checking** uses semi-algorithms to check that a global **model** (representing the whole **system**) **meets the requirements.**
- **System Model:** Petri Nets, Timed Automata, Hybrid Automata, etc.
- **Formal Properties:** Temporal Logic





Procedure

- Formalization of the System.
- Formalization of the Specification Requirements.

Contributions

- Methodology: Transformation rules.
- Automatic generation Tool:
  ST → Abstract Automata → Specific model

emf @ECLIPSE MODELING FRAMEWORK Xtext

Analysis & Conclusions

	Pros ☺	Cons ☹
Automatic Testing	Testing the **real system** Technology is available Reduce human errors **Reusable** for different PLC	Sophisticated maintenance High price for **new test case** Black box testing Difficult to find the **source of the problem**
Model Checking	**Explores all the combinations** Earlier bug detection Avoid human errors Complexity hidden by the generation tools **Counter-examples** to find the source problem	Verification of a **system model** Need of automatic generation tools Need to prove the transformations State space explosion Applying **abstraction** techniques is not trivial

- PLC software development lacks of modern software engineering best practices such as unit test or daily builds**.**
- Testing and formal verification are **complementary** not exclusive.
- Both **formal verification** and **automatic testing** can be integrated in the development process of PLC code in order to **detect and correct bugs before the deployment**.

THPPC080

bfernand@cern.ch

Industrial Controls Engineering Department

CERN EN Engineering Department