Program Analysis for Industrial Automation Systems

Raoul Jetley, Principal Scientist, ABB Corporate Research
National Workshop on Formal Methods
17th October, 2014

Power and productivity
for a better world™

Program Analysis for Industrial Automation Systems
Outline

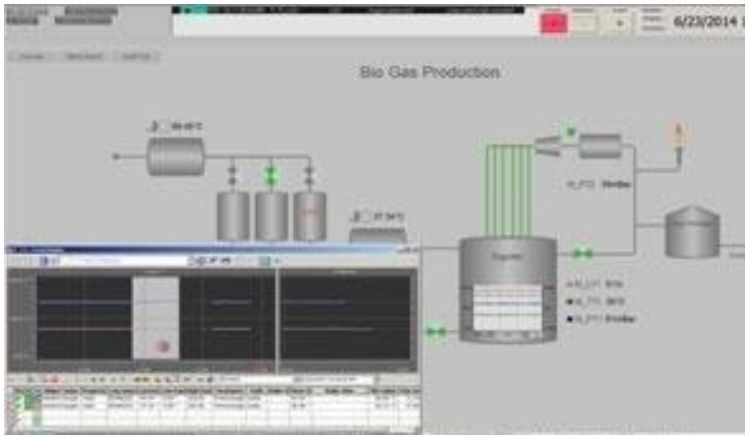
- Introduction to Industrial Automation systems and languages
- Challenges and problems to be addressed
- Tools for Program Analysis
  - Static Code Analysis for Structured Text and Function Block Diagrams
  - Change Impact Analysis for Automation Engineering System
- Summary

© ABB Group

Program Analysis for Industrial Automation Systems
Introduction

- Automation Engineering
  - Use of control systems for operating industrial systems
  - Typically used in industrial settings, manufacturing, power systems, oil & gas refineries, marine operations

- Objects in automation engineering system
  - Topology Data
  - Geometry
  - Kinematics
  - Human Machine Interface
  - Control Logic

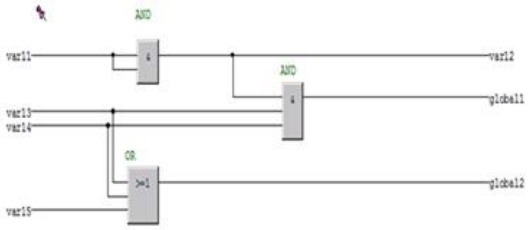


- Control logic comprises sequencing, behavior and control
  - Specified using (proprietary) domain-specific languages.
  - Consist of a mix of textual and graphical (fourth generation) languages

© ABB Group

Program Analysis for Industrial Automation Systems
Control System Languages

- Languages used – include textual and graphical languages

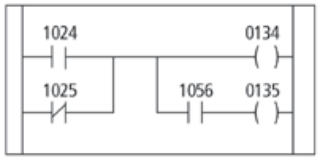


FBD

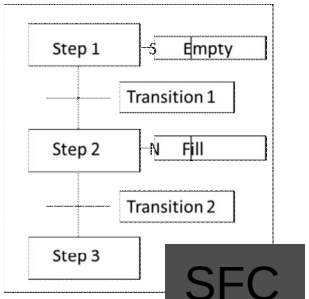
```
Sp_Temp + TempDeadband;
Sp_Temp - TempDeadband;

c Control *)
Setpoint - Temp_Hysteresis then
true;
```

ST



LD



SFC

```
LIMIT(
EN := COND,
IN := A,
MN := 5,
MX := 1,
ENO => TEMPL
)
ST B
```

IL

- Traditionally, defined using LL; over time more abstraction and additional constructs added
  - Five languages most popular – LL, IL, ST, FBD and SFC
  - Standards defined – IEC 61131-3, IEC 61499
  - PLCOpen, AutomationML efforts to standardize languages

- Need method to reason about correctness
  - Develop tools for program comprehension, automation of engineering effort

© ABB Group

Program Analysis for Industrial Automation Systems

What is Program Analysis?

- Analyzing behavior of software
  - Used to detect errors & security vulnerabilities, improve code comprehension, automate testing, compute impact of change, collect metrics, temporal and spatial analyses
  - Can be both static & dynamic – we focus on static analysis methods

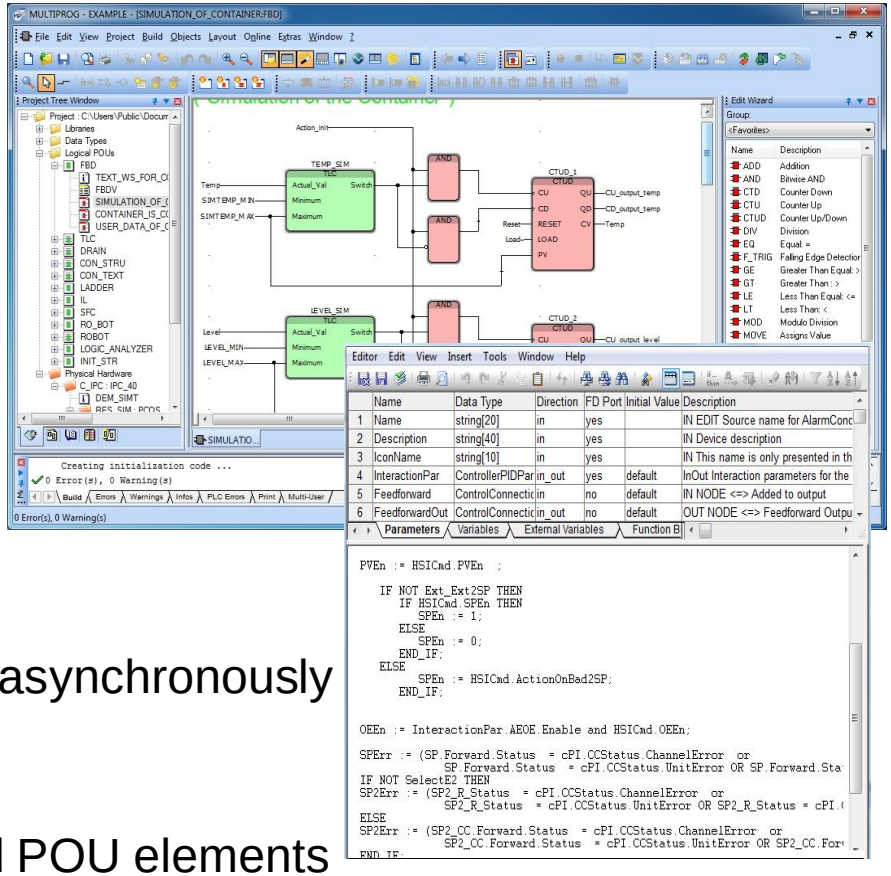
- Number of techniques and tools available for analysis of general purpose languages
  - Symbolic execution, Constraint solving, Program Slicing, Model checking, etc.

- Challenge
  - Need to adapt these techniques for automation engineering languages

© ABB Group

Program Analysis for Industrial Automation Systems
Domain Specific Challenges

- Programs defined using combination of text and diagrams
  - Interdependencies between different types of Program Organization Units (POUs)



- Differing semantics – e.g., Multiple I/O ports as opposed to only one output

- Tasks configured to execute applications
  - Can be run in parallel, synchronously or asynchronously

- Nesting of POUs
  - Rules defined for accessing parent, child POU elements

- Execution based on data flow rather than explicit control flow
  - Determined by data connections, positions of function blocks

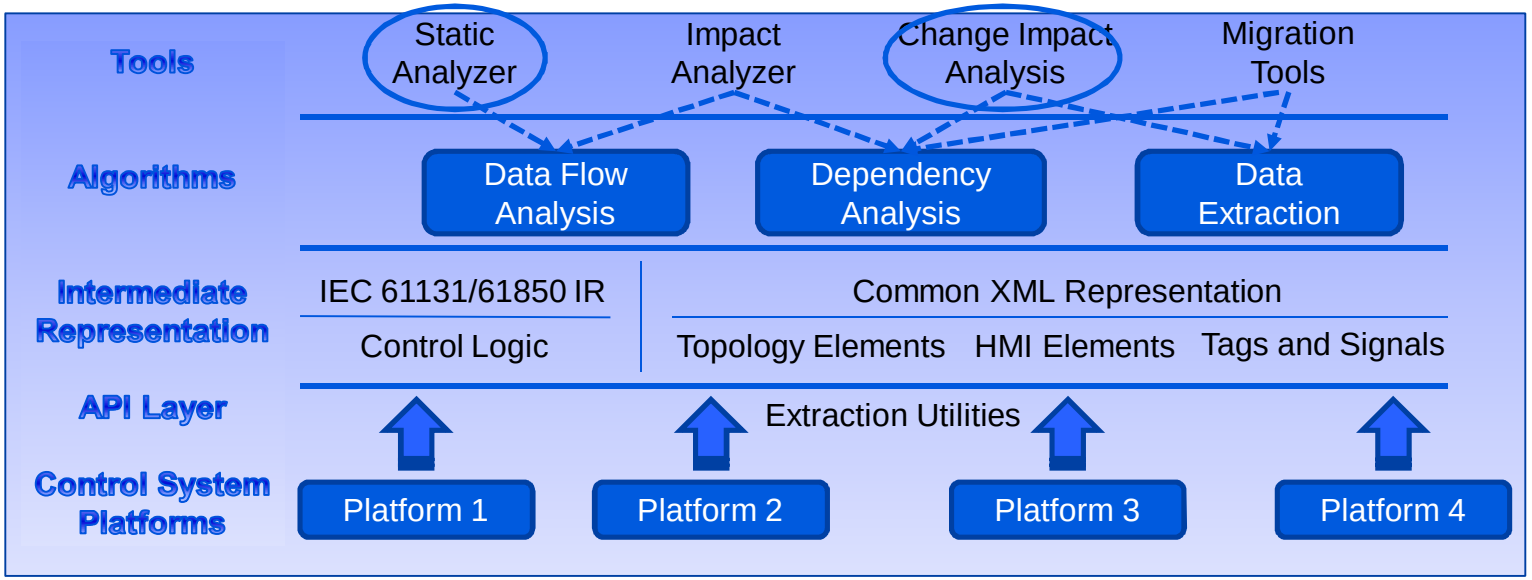
© ABB Group

Program Analysis for Industrial Automation Systems

Program Analysis Framework

- Objective
  - To enhance productivity of control system applications through a tool suite specifically targeted towards industrial automation platforms

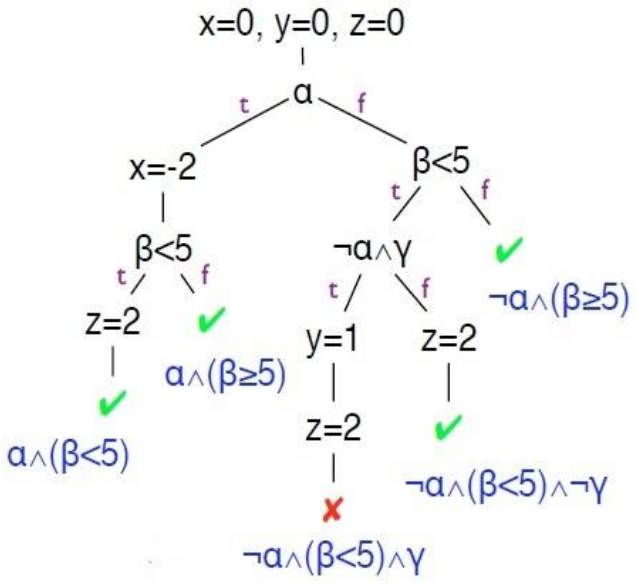
- Common framework for developing tools
  - Based on intermediate XML representation
  - Used for both control logic and other system elements



© ABB Group

Program Analysis for Industrial Automation Systems
Static Analysis Tool

- Overview
  - Automated tool to detect potential error conditions and coding compliance violations in Control system applications
  - Current focus on Structured Text & Function Block Diagrams; other languages to be targeted in future
- Benefits
  - Early detection of errors; Reduction of effort in testing, V&V activities
  - Conformance to guidelines, coding standards
- Analysis Method
  - Reason about all possible executions of a program
  - Use symbolic execution to determine if certain formulas are satisfiable



© ABB Group

Program Analysis for Industrial Automation Systems
Static Analysis Tool – Checks

- Types of checks – DFA-based, sematic checks, compliance checks
  - Can be pre-defined or user-defined
  - Consulted application engineers to collect list of rules

- Compliance rules/coding guidelines
  - All local variables should have a prefix "**LV**_"
  - All local variables should have the attribute cold_reset

- Runtime error checking
  - Divide by zero errors
  - Buffer overflow / underflow (Out of bounds)

- Best practices
  - Nesting level for loops should not be more than 3
  - Numeric variables used within a FOR loop for iteration counting should not be modified in the body of the loop

© ABB Group

Program Analysis for Industrial Automation Systems
Static Analysis Tool – Implementation

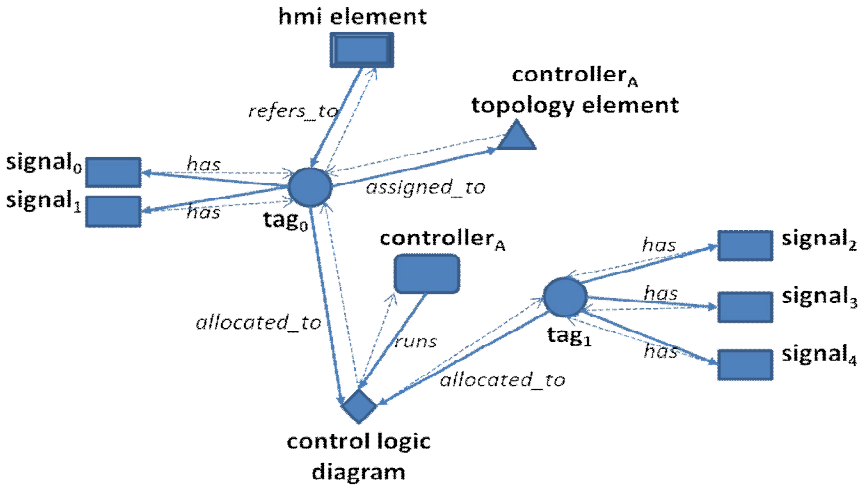
- Extract POUs
  - Extracted from automation engineering platform in XML format
  - Use APIs to extract code
- Parse extracted code
  - Generate AST from parser
  - Use parse tree to derive CFG (basic blocks and dependences)
- Perform Analyses using CFG
  - General analysis framework – extended for specific analyses like Live Variable Analysis, Reachability, Interval analysis
  - Instances of these used to detect invalid states
  - Simple semantic analyses can be performed on AST to detect unreachable code, compliance checks

© ABB Group

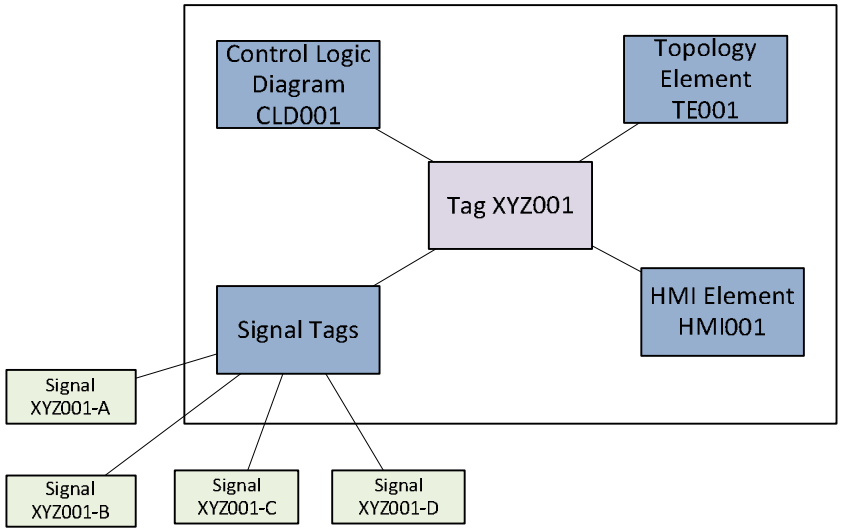
Program Analysis for Industrial Automation Systems

Change Impact Analysis

- Tool to deduce dependence between engineering system elements
  - Extract system data through API layer of different tools to build a graph G=(V, E)
  - Each vertex v ∈ V denotes an engineering system element
  - Each edge e ∈ E denotes a relation between vertices



Part of a Plant Automation Engineering System Element Graph



Data Dependencies between Automation Engineering Elements

- Analysis Method
  - Use graph based algorithms to provide dependency analysis, impact analysis, etc.
  - Develop visualization algorithms to show relations between elements to users

© ABB Group

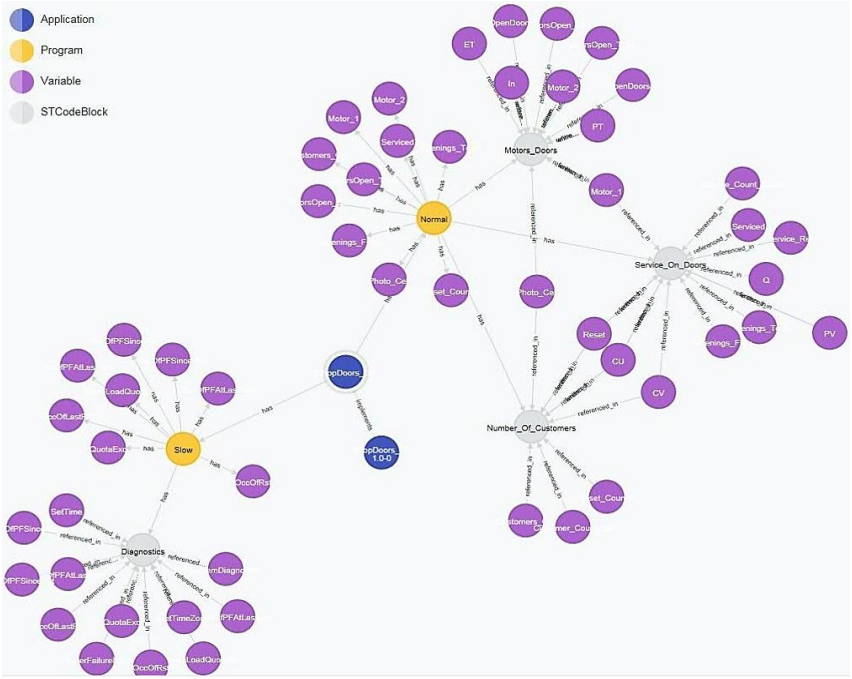
Program Analysis for Industrial Automation Systems
Change Impact Analysis – Representation

- Computing the Graph
  - G = (V, E), where V (vertices) is a set of entities and E (edges) is a set of relations
  - $V = C \cup S \cup A \cup T \cup H$
    - Where, C is the set of controller hosts, S is the set of signals, A is the set of control applications, T is the set of tags, H is the set of HMI elements
  - The dependence relation set R can be defined as $R: V \times V$, such that:
    - t “has” s, iff $t \in T$, $s \in S$, and the tag t has the signal s associated with it
    - t “assigned_to” c, iff $t \in T$, $c \in C$, and the tag t is assigned to the controller host c.
    - t “allocated_to” a, iff $t \in T$, $a \in A$, and tag t is allocated to the control application a
    - h “refers_to” t, iff $h \in H$, $t \in T$, and the HMI element h refers to the tag t.
    - a “executes_on” c, iff $a \in A$, $c \in C$, and the application a executes on controller c
- Searching the Graph
  - Use Iterative deepening depth-first search to collect all nodes that can be reached from (or precede) the specified element

© ABB Group

Program Analysis for Industrial Automation Systems
Change Impact Analysis – Implementation

- Graph Storage
  - Uses neo4j, allowing for portable deployment across platforms
  - Requires that a neo4j server is running on the machine
- Graph Visualization
  - Implemented using Graph# API
  - Allows user to interactively navigate through engineering system graph
  - What-if analysis performed to determine change impact for a selected element
- Currently implemented at POU level
  - Can analyze code within POUs for end-to-end analysis
  - Provide tool support by integrating with engineering platform



© ABB Group

Program Analysis for Industrial Automation Systems
Summary

- Traditional program analysis techniques can be adapted for Industrial Automation systems
  - Examples covered – static analysis, change impact analysis
  - Can be extended to include more formal analysis – model checking, automated test case generation, etc.
- Challenges include addressing problems specific to the domain
- Research already being performed in academia, need to work with industry to provide integrated solutions and tools

© ABB Group

Program Analysis for Industrial Automation Systems
Contact

- Raoul Jetley

  raoul.jetley@in.abb.com

  Principal Scientist at ABB Corporate Research, Bangalore

  Formerly Research Scientist and Laboratory Leader at US FDA

  PhD from NC State University (2006)

© ABB Group

Power and productivity
for a better world™