

Beremiz

*The Free and Open Source
IEC 61131-3
Automation IDE*

CANopen



Open Source for Open Standards

- Despite of open standards such as IEC 61131, PLCOpen and CanOpen, control engineers cannot easily transfer programs between vendor solutions.



- To this end, the Beremiz Project produce Free and Open Source software for automation :
 - Integrated Development Environment
 - Embedded runtime software
 - Automation, control and HMI software



What is Beremiz ?

- Presentation covers the 4 sub-projects Beremiz relies on :



CAN Festival

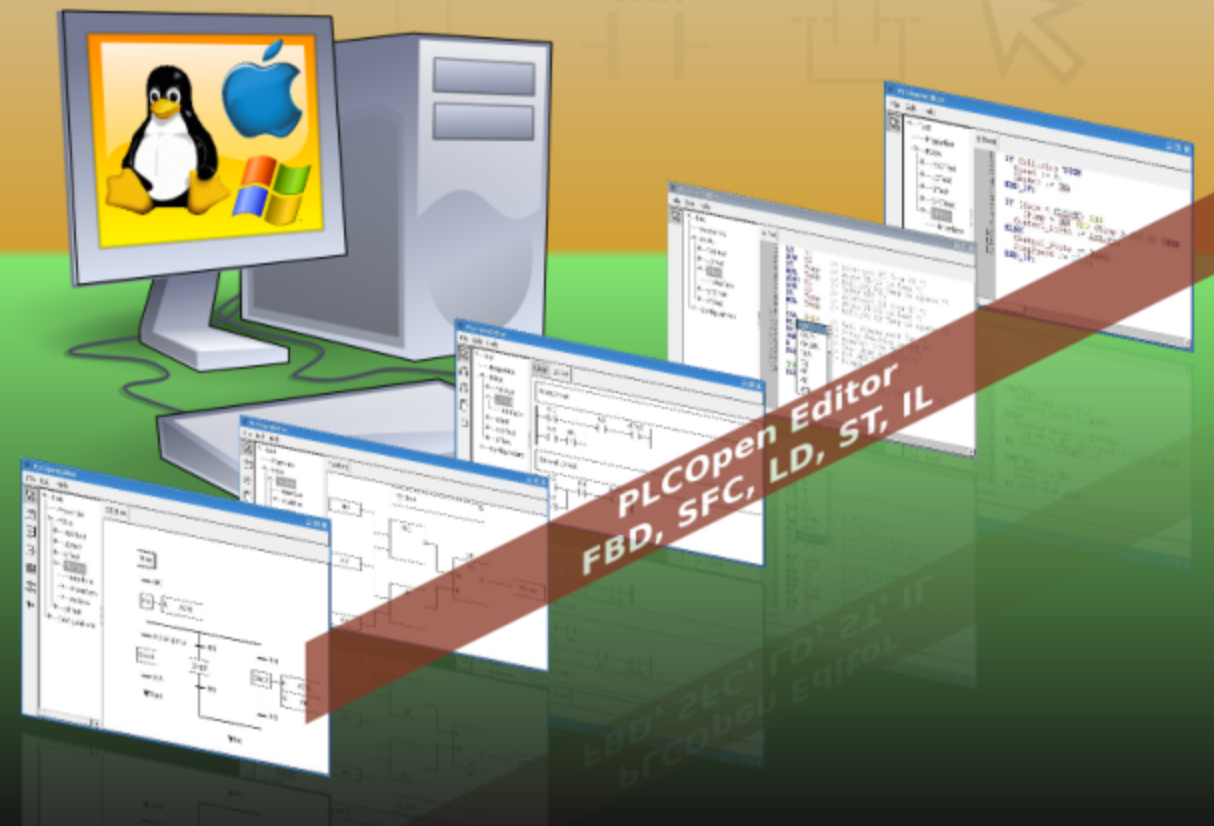


- 1 - The PLCOpen Editor
- 2 - The MatPLC's IEC compiler
- 3 - CanFestival
- 4 - SVGUI



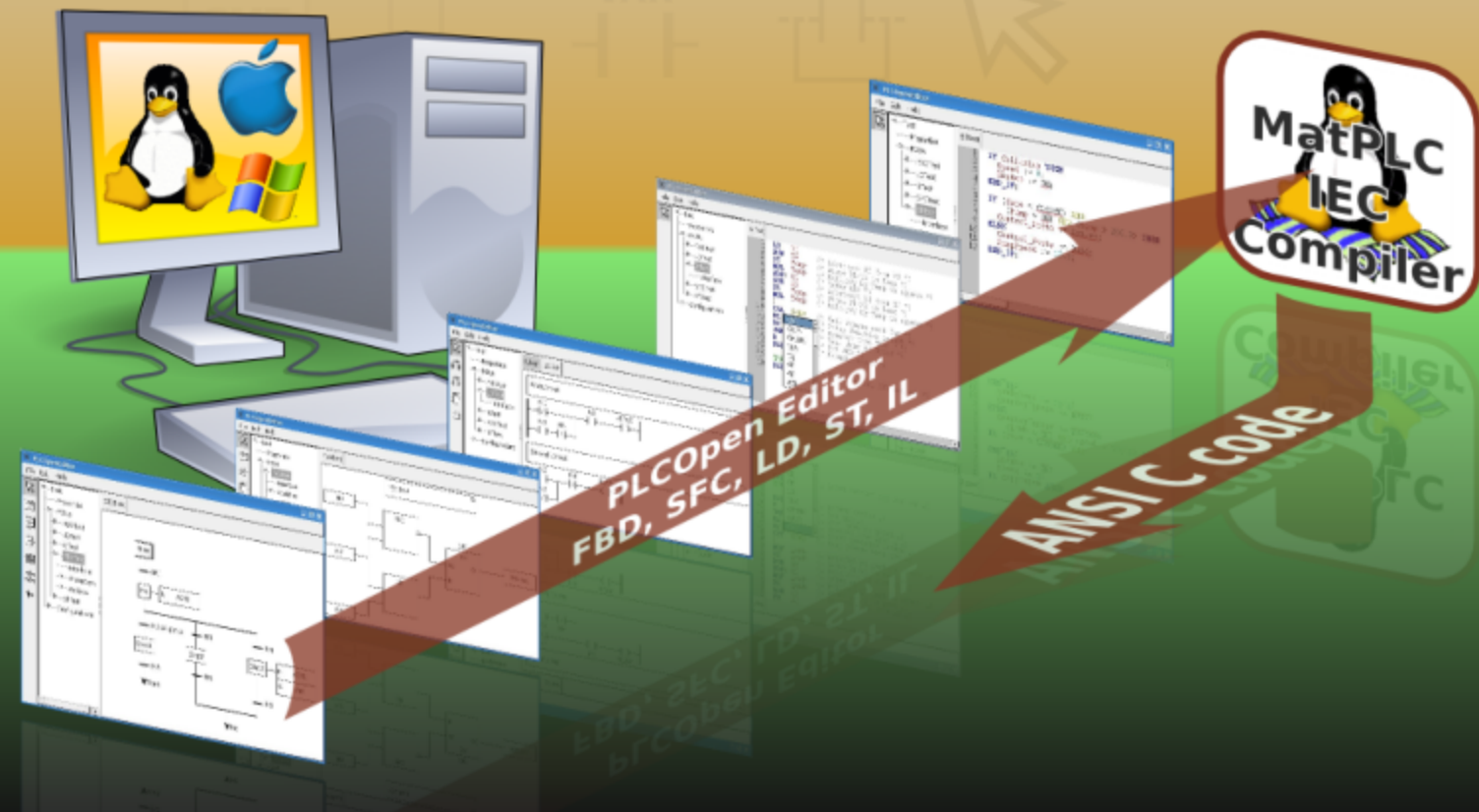
What is Beremiz ?

1. Multi-platform **IDE** for **automation**



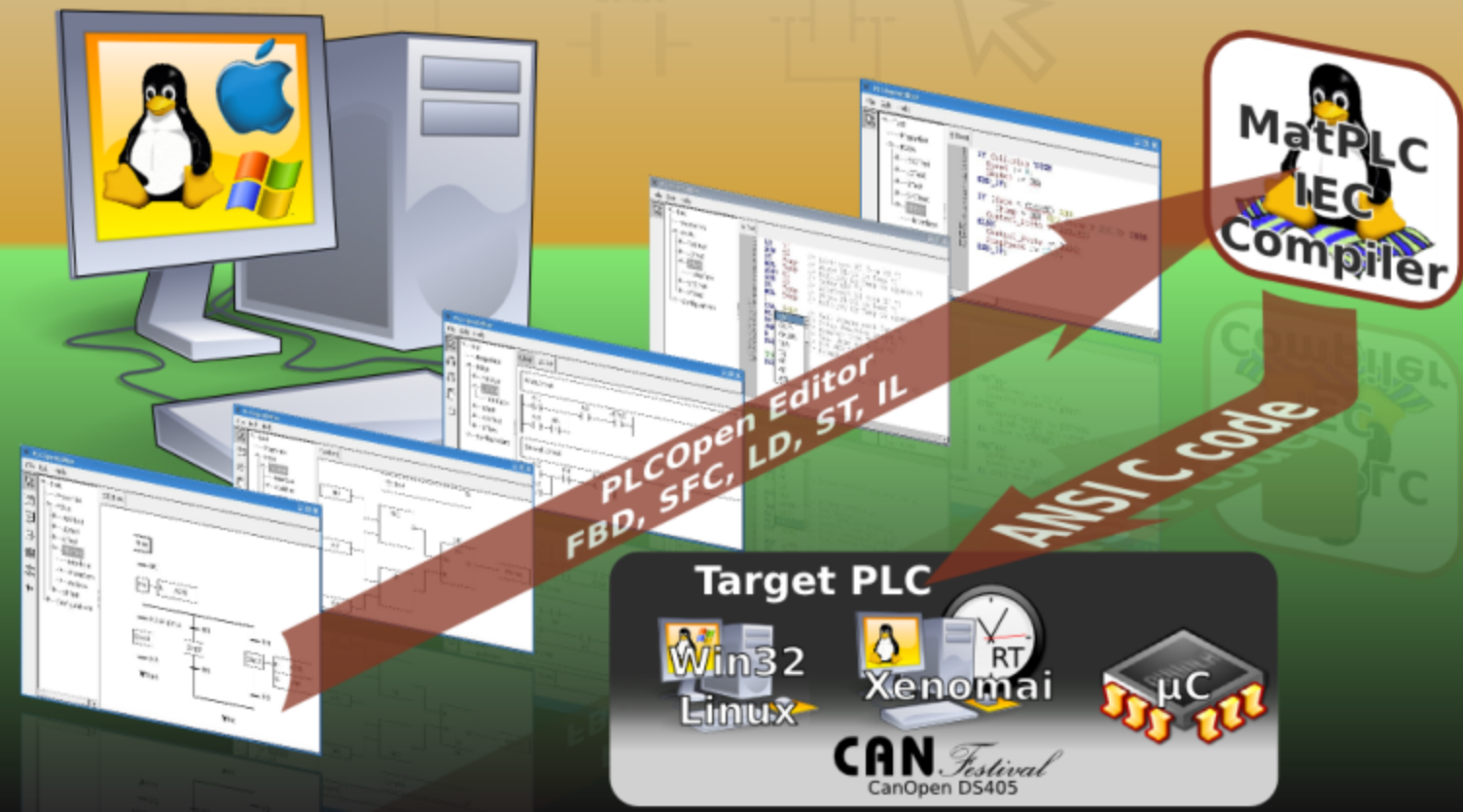
What is Beremiz ?

2. IEC 61131-3 compiler



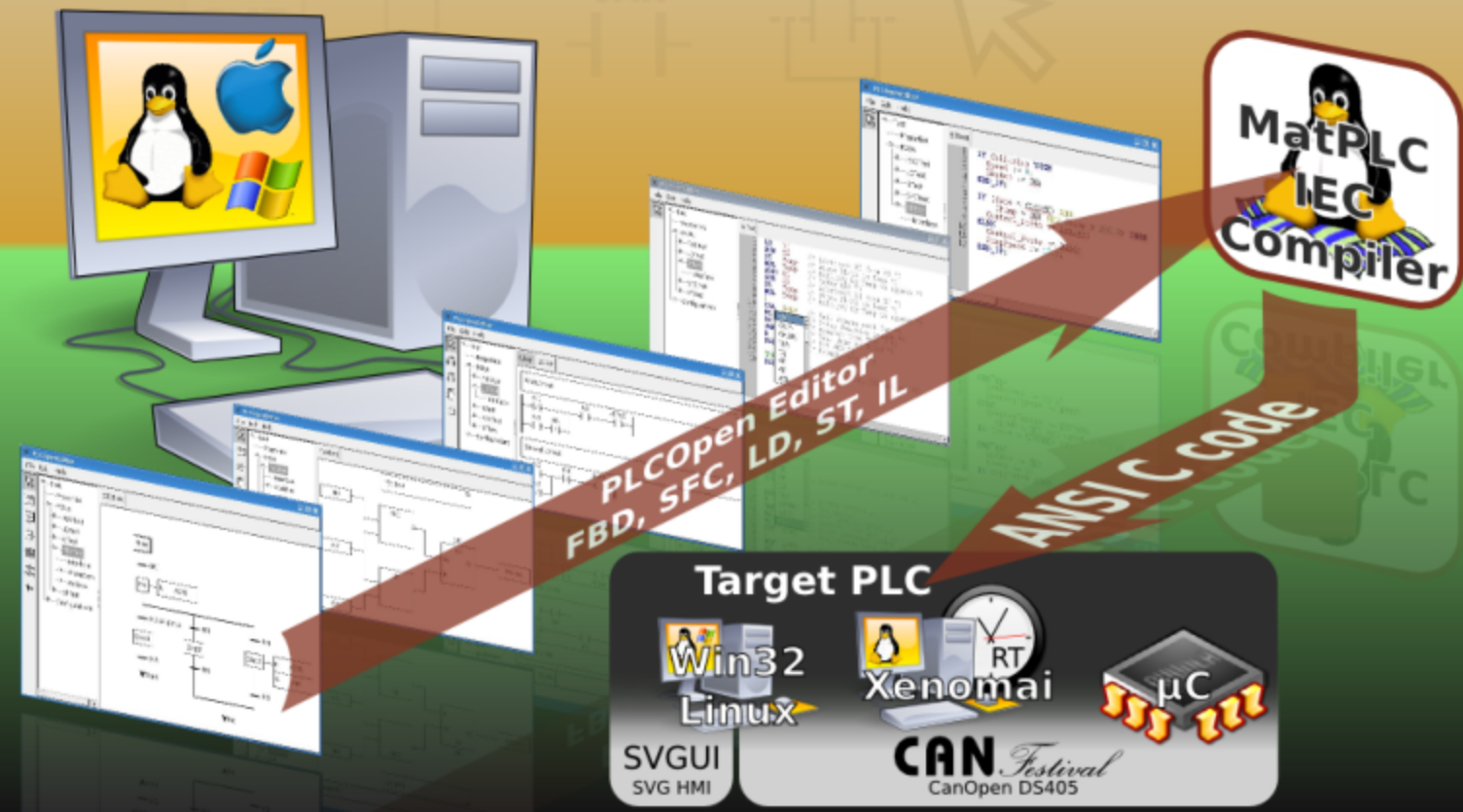
What is Beremiz ?

3. CANOpen interface to physical I/O

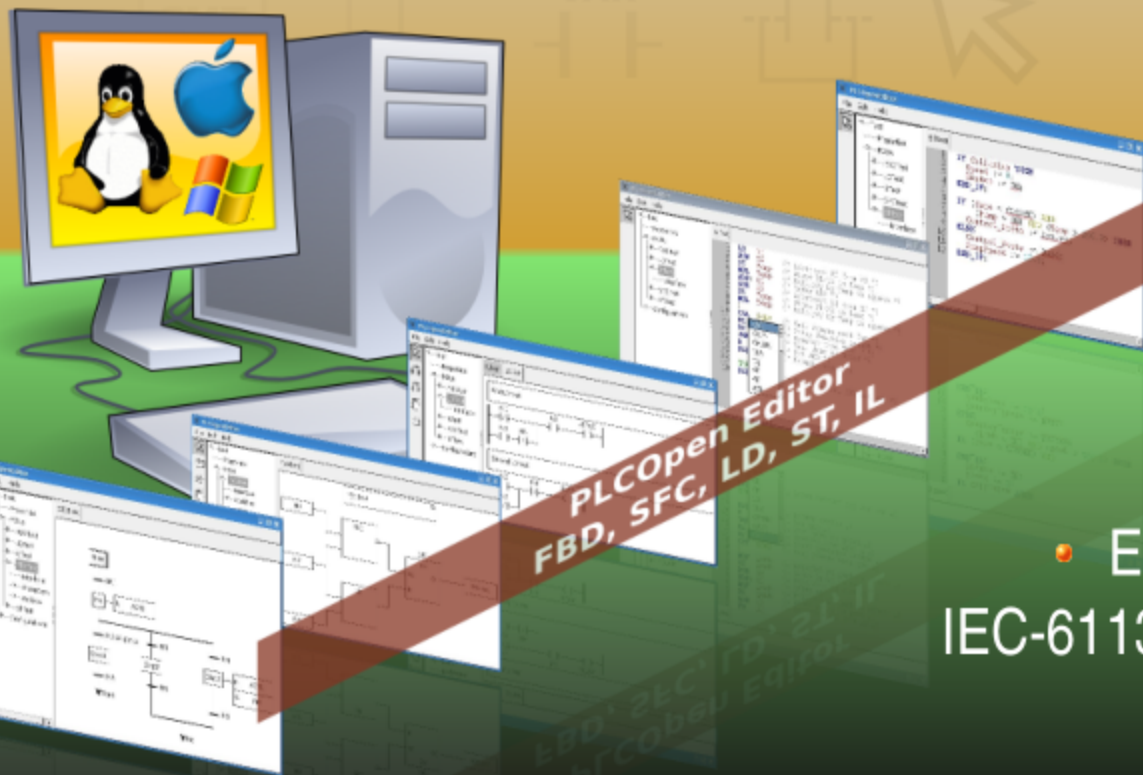


What is Beremiz ?

4. automated **HMI** tool, based on **SVG**



1. The *PLCOpen Editor*



- Edits all 5 of the IEC-61131-3 languages

1. The PLCOpen Editor

PLCOpenEditor

File Edit Help

Test

- Properties
- Functions
 - LDTest
 - ILTest
- Function Blocks
 - FBDTest
 - STTest
- Programs
- Configurations

LDTest | ILTest | FBDTest

| # | Name | Class | Type | Initial Value | Retain | Constant |
|---|------|--------|------|---------------|--------|----------|
| 1 | IN1 | Input | BOOL | false | No | No |
| 2 | IN2 | Input | BOOL | [4, [0, 1]] | No | No |
| 3 | IN3 | Input | BOOL | | No | No |
| 4 | OUT | Output | BOOL | | No | No |

Class Filter: All

^ Add
v Delete

Function Block Diagram - FBD

1. The PLCOpen Editor

PLCOpenEditor

File Edit Help

Test

Properties

Functions

- LDTest
- ILTest

Function Blocks

- FBDTest
- STTest

Programs

- SFCTest

Configurations

LDTest | ILTest | FBDTest | STTest | SFCTest

| # | Name | Class | Type | Location | Initial Value | Retain | Constant |
|---|------|-------|------|----------|---------------|--------|----------|
| 1 | IN1 | Input | BOOL | None | | No | No |
| 2 | IN2 | Input | BOOL | None | | No | No |
| 3 | IN3 | Input | BOOL | None | | No | No |
| 4 | IN4 | Input | BOOL | None | | No | No |
| 5 | IN5 | Input | BOOL | None | | No | No |

| N | ACT1 | IN5 |
|-------|--------------|-----|
| D | IN1 | |
| T#10s | | |
| P | IN2 := TRUE; | |

Class Filter: All

^ Add

v Delete

Sequential Function Chart - SFC

1. The PLCOpen Editor

The screenshot shows the PLCOpenEditor window with a menu bar (File, Edit, Help) and a tree view on the left containing 'Test', 'Properties', 'Functions' (with 'LDTest' and 'ILTest' selected), 'Function Blocks', 'Programs', and 'Configurations'. The main workspace displays a ladder logic diagram for the 'LDTest' function block. The diagram consists of a network with a coil labeled 'LDTest'. The coil is connected to a series of inputs: IN1 (normally open), IN2 (normally closed), and a parallel combination of IN3 (normally open) and IN4 (normally open). A 'Commentaire' text box is located above the diagram.

| # | Name | Class | Type | Initial Value | Retai | Return Type: |
|---|------|-------|------|---------------|-------|--------------|
| 1 | IN1 | Input | BOOL | | No | BOOL |
| 2 | IN2 | Input | BOOL | | No | |
| 3 | IN3 | Input | BOOL | | No | |
| 4 | IN4 | Input | BOOL | | No | |

Additional controls on the right side of the table include a 'Class Filter' set to 'All' and buttons for '^ Add' and 'v Delete'.

Ladder Diagram LD

1. The PLCOpen Editor

The screenshot shows the PLCOpen Editor interface. The main editing area contains the following Structured Text (ST) code:

```

1 IF Collision THEN
2   Speed := 0;
3   Brakes := TRUE;
4 END_IF;
5
6 IF (Gate = TRUE) AND
7   (Pump = TRUE) AND (Temp > 200.0) THEN
8   Control_State := TRUE;
9 ELSE
10  Control_State := FALSE;
11  PumpSpeed := 10.0;
12 END_IF;

```

Below the code editor is a table defining the variables used in the program:

| # | Name | Class | Type | Initial Value | Retain | Const |
|---|-----------|--------|------|---------------|--------|-------|
| 1 | Collision | Input | BOOL | | No | No |
| 2 | Gate | Input | BOOL | | No | No |
| 3 | Pump | Input | BOOL | | No | No |
| 4 | Temp | Input | REAL | | No | No |
| 5 | Speed | Output | INT | | No | No |
| 6 | PumpSpeed | Output | INT | | No | No |

On the right side of the table, there is a 'Class Filter' dropdown menu set to 'All' and two buttons: '^ Add' and 'v Delete'.

Structured Text - ST

1. The PLCOpen Editor

The screenshot shows the PLCOpenEditor window with a project named 'Test'. The left sidebar shows a tree view with 'Functions' expanded to 'ILTest'. The main editor displays a ladder logic program with the following instructions:

```

6      SUB  X2      (* Subtract X1 from X2 *)
7      ST   Temp   (* Store X1-X2 in Temp *)
8      MUL  Temp   (* Multiply by Temp to square *)
9
10     SQRT          (* Call Square root fun *)
11     ST   ILTest  (* Setup function result *)
12     GT   TMax    (* Greater than TMax ? *)
13     JMP  ERR     (* Yes, Jump to Error *)
14     S    ERROR   (* Set ERROR *)
15     RET          (* Normal return *)
16 ERR: RET        (* Error return, ENO not set *)
  
```

Below the editor is a table showing the instruction list:

| # | Name | Class | Type | Initial Value | Retain |
|---|------|-------|------|---------------|--------|
| 1 | X3 | Input | REAL | | No |
| 2 | Temp | Local | REAL | | No |
| 3 | X2 | Input | REAL | | No |
| 4 | Y1 | Input | REAL | | No |
| 5 | Y2 | Input | REAL | | No |
| 6 | TMax | Input | REAL | | No |

On the right side of the table, there are controls for 'Return Type' (set to REAL) and 'Class Filter' (set to All). There are also 'Add' and 'Delete' buttons.

Instruction List IL

1. The PLCOpen Editor

The screenshot shows the PLCOpen Editor interface. On the left is a tree view with the following structure:

- Test
 - Properties
 - Functions
 - LDTest
 - ILTest
 - Function Blocks
 - FBDTest
 - STTest
 - Programs
 - SFCTest
 - Configurations
 - ConfigTest
 - Resources
 - Resource
 - ConfigTest2

The main workspace contains several panels:

- Toolbar: CMT, START, STOP, ACT, STEP, and a downward arrow.
- Navigation: LDTest, ILTest, FBDTest, STTest, SFCTest, and ConfigTest-ResourceTest.
- Tasks Table:

| Name | Single | Interval | Priority |
|------|--------|----------|----------|
| Toto | | 1h100ms | 6 |
- Instances Table:

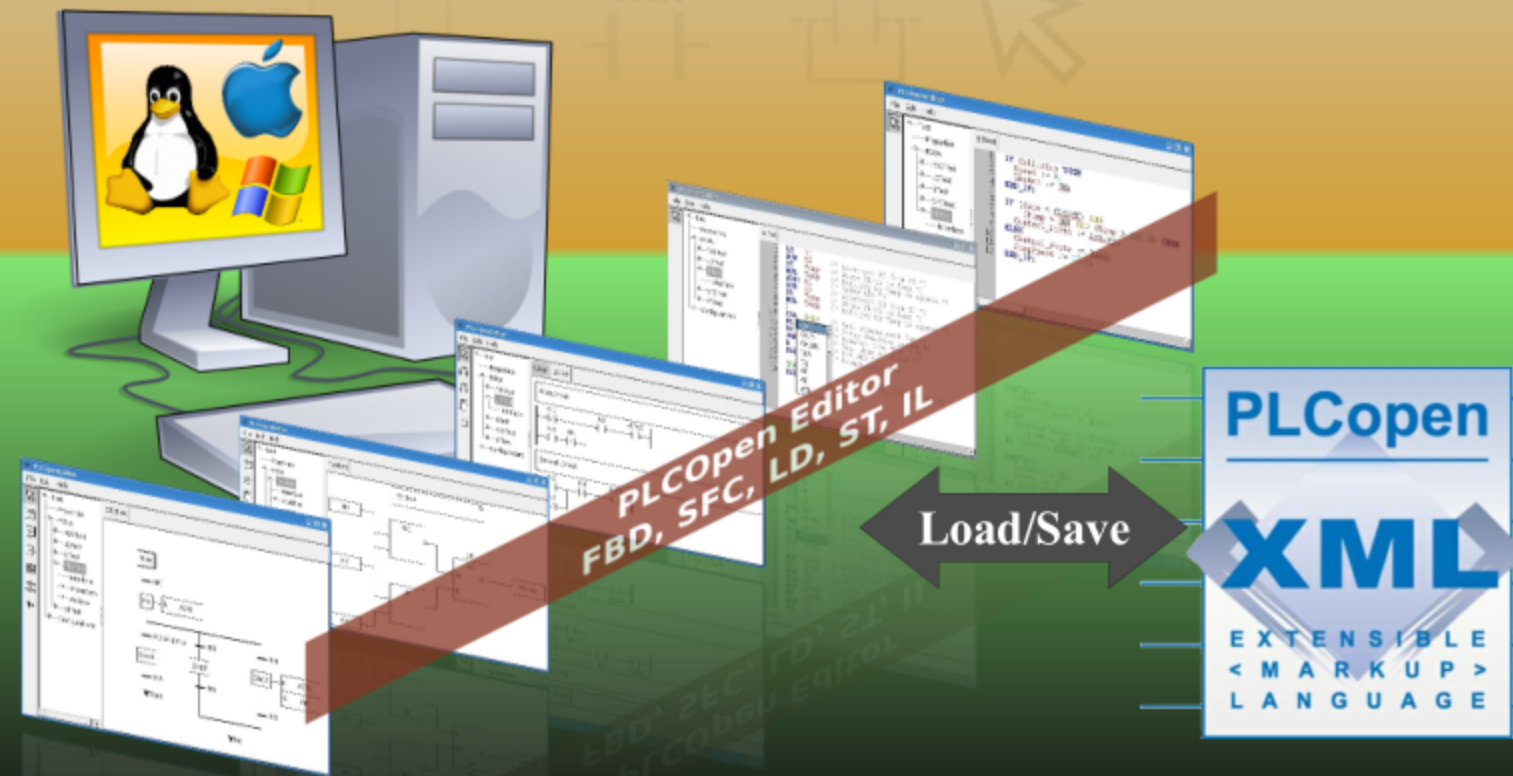
| Name | Type | Task |
|----------|---------|------|
| Program1 | SFCTest | Toto |
- Resources Table:

| # | Name | Class | Type | Location | Initial Value | Retain | Co |
|---|------|--------|------|----------|---------------|--------|----|
| 1 | Titi | Global | INT | M30 | | No | |
- Class Filter: All
- Buttons: Add, Delete, ^, v

Configurations, Resources and Tasks

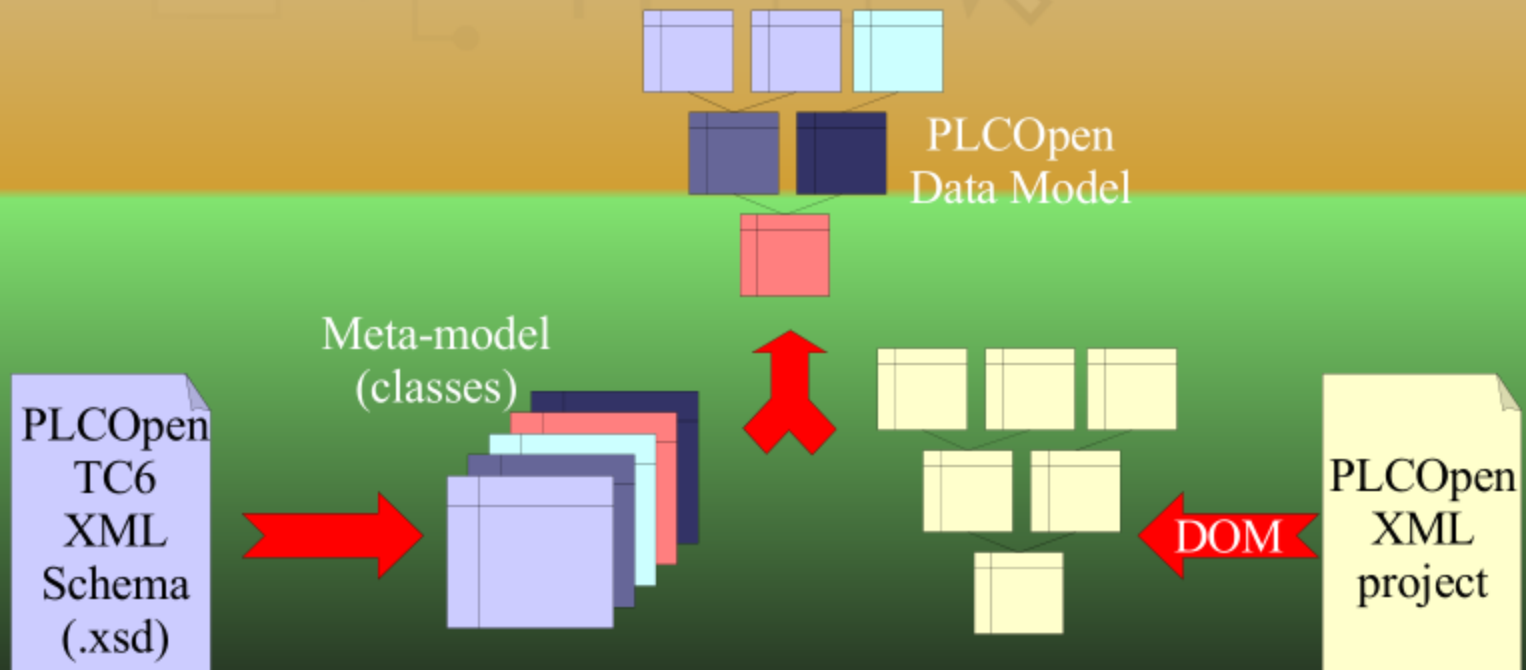
1. The *PLCOpen Editor*

- Saves and loads XML projects accordingly to TC6-XML



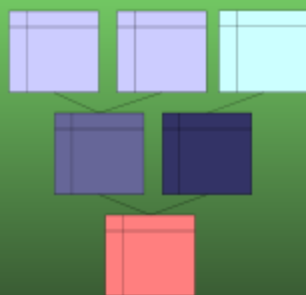
1. The PLCOpen Editor

- Data-model is based on TC6-XML XML Schema.



1. The *PLCOpen Editor*

- PLCOpen editor has built-in export filter that convert graphical languages to their equivalent textual form
(FB, LD, SFC)=>(ST, IL, SFC)

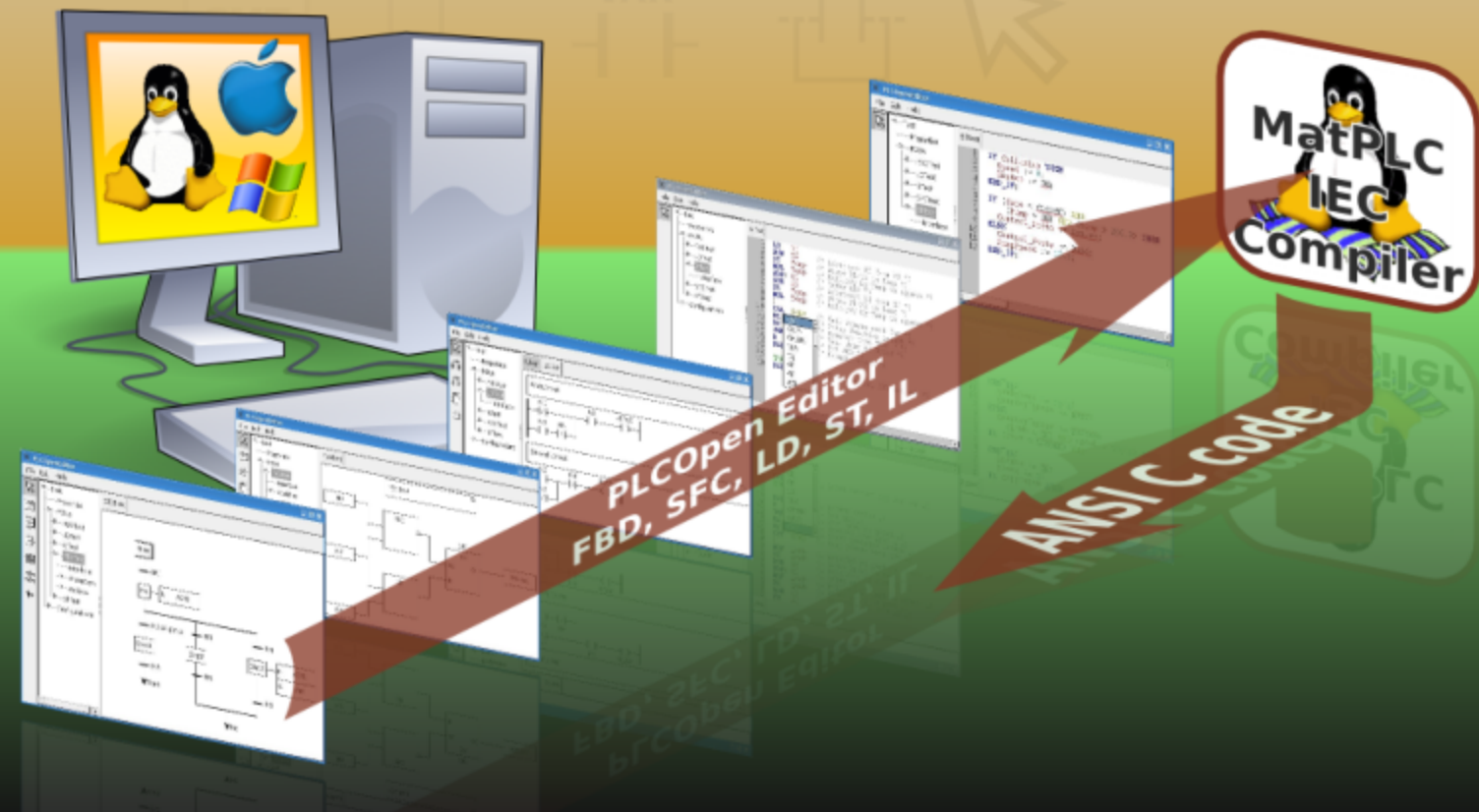


PLCOpen
Data Model



IEC-61131-3
Textual
Languages
(ST, IL, SFC)

2. The IEC to ANSI-C compiler



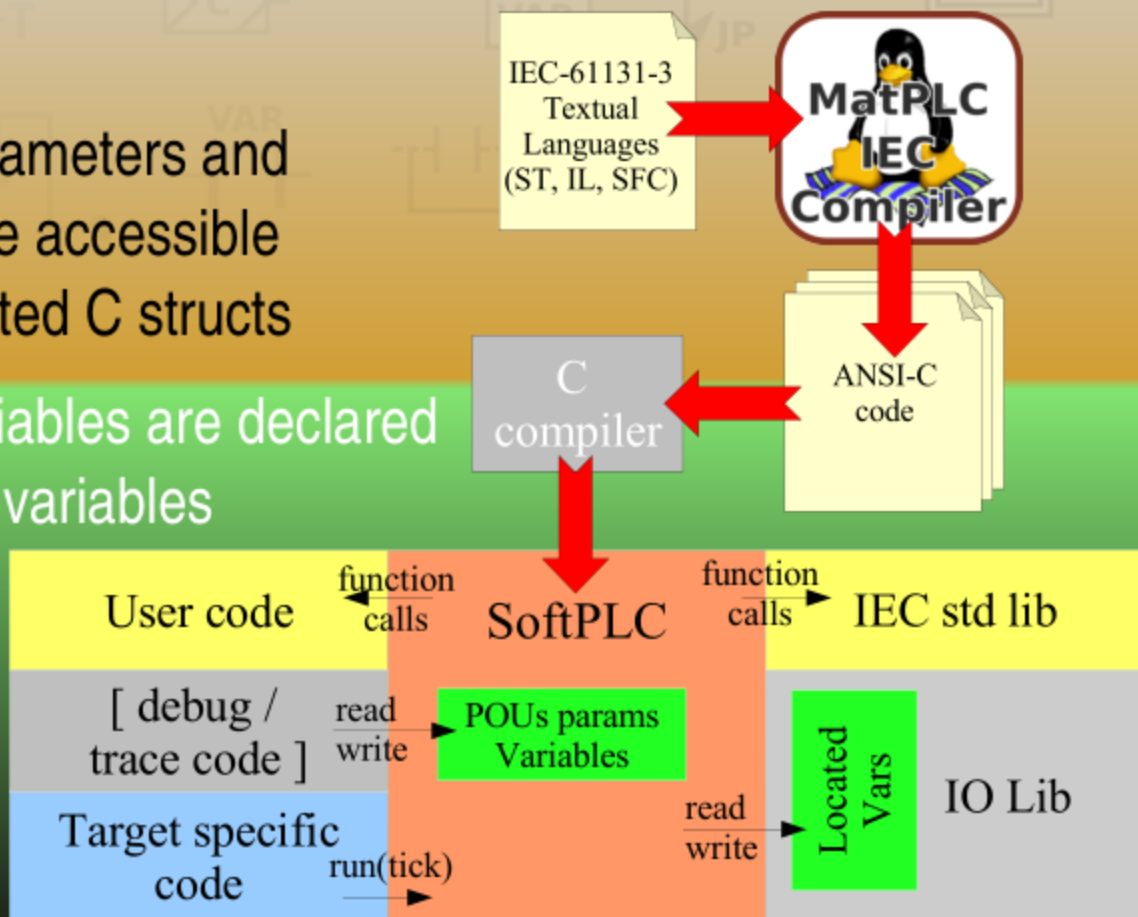
2. *The IEC to ANSI-C compiler*

- Project started in 2002 by Mario de Sousa (U-Porto)
- Compiles ST/IL/SFC code into ANSI-C code.

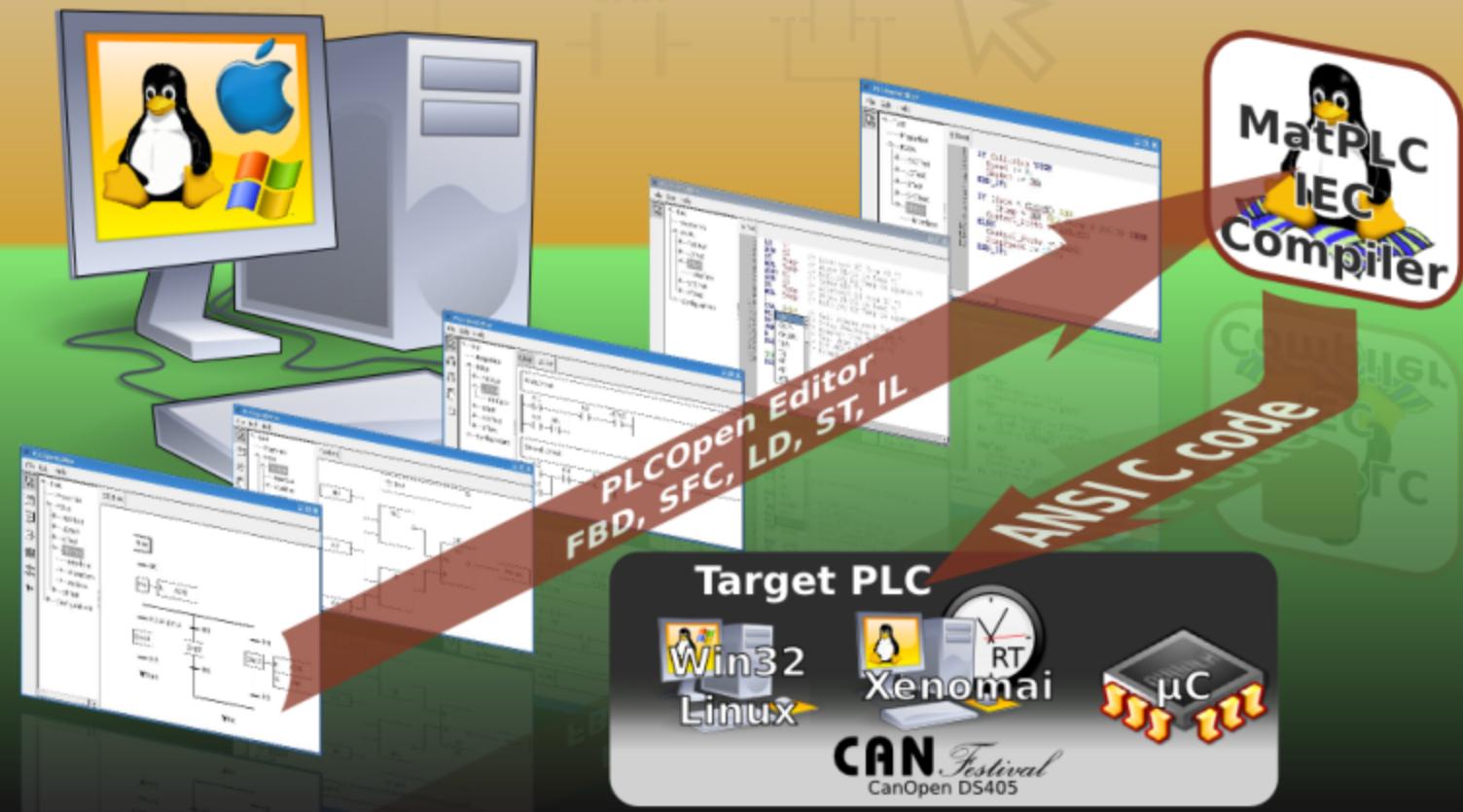


2. The IEC to ANSI-C compiler

- All POU parameters and variables are accessible through nested C structs
- Located variables are declared as extern C variables



3. CanFestival CanOpen stack



3. CanFestival CanOpen stack

- Started in 2001 by Edouard TISSERANT
- Runs on any target, with or without OS, fully ANSI-C
- For Beremiz, CanFestival provides :
 - A point & click GUI for CANOpen I/O
 - An I/O library
 - A HAL library

CAN *Festival*



| | | |
|---------------------------------|---------|------------------------|
| User code | SoftPLC | IEC std lib |
| [debug / trace code] | | IO Lib: CanFestival |
| Target specific: CanFestival | | |



3. CanFestival CanOpen stack

- Network Topology Editor maps CANOpen process variables to IEC Located Variables with simple drag & drop
- User just provides vendor EDS file for each node

The screenshot shows the 'Networkedit - TestNodeList' application window. The main area displays a list of CANOpen objects for a node (0x05 TestNode):

- 0x1800-0x19FF Transmit PDO Parameters
- 0x1A00-0x1BFF Transmit PDO Mapping (selected)
- 0x1C00-0x1FFF Other Communication Parameters
- 0x2000-0x5FFF Manufacturer Specific

The selected object (0x1A00-0x1BFF) is expanded to show its subindex details:

- 0x1A00 Transmit PDO 1 Mapping
- 0x1A01 Transmit PDO 2 Mapping (selected)
- 0x1A02 Transmit PDO 3 Mapping

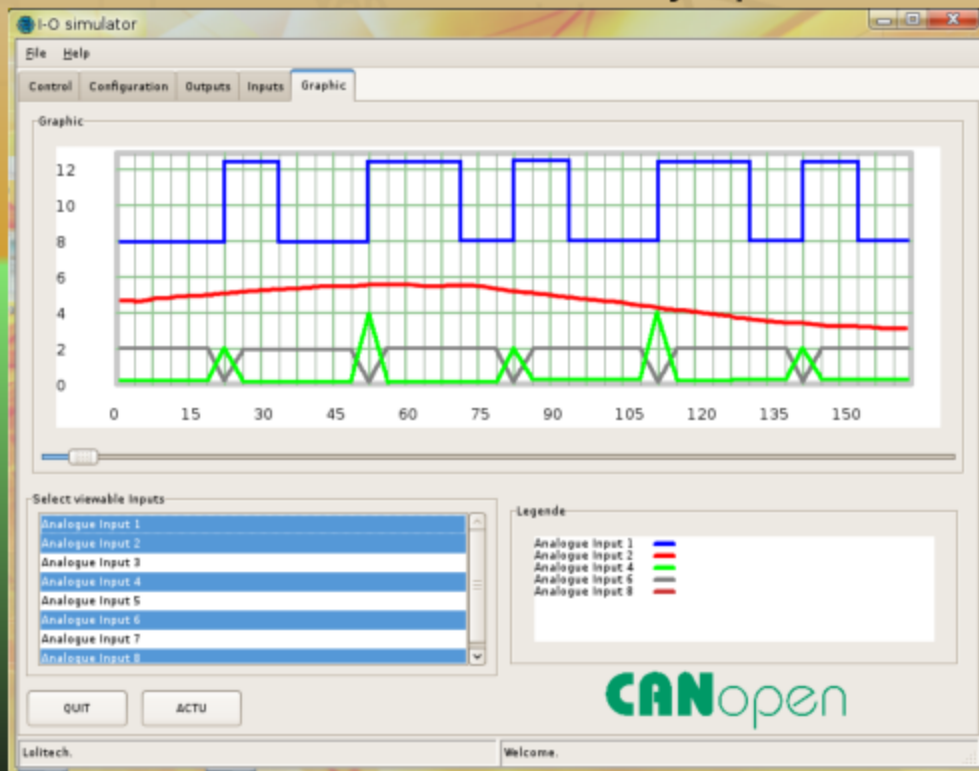
Below the list, there is a table of subindex data:

| subindex | name | type | value | access |
|----------|---------------------------------------|------------|------------------|------------|
| 0x00 | Number of Entries | UNSIGNED8 | 4 | Read/Write |
| 0x01 | PDO 2 Mapping for a process data vari | UNSIGNED32 | Analogue Input 1 | Read/Write |
| 0x02 | PDO 2 Mapping for a process data vari | UNSIGNED32 | Analogue Input 2 | Read/Write |
| 0x03 | PDO 2 Mapping for a process data vari | UNSIGNED32 | Analogue Input 3 | Read/Write |
| 0x04 | PDO 2 Mapping for a process data vari | UNSIGNED32 | Analogue Input 4 | Read/Write |

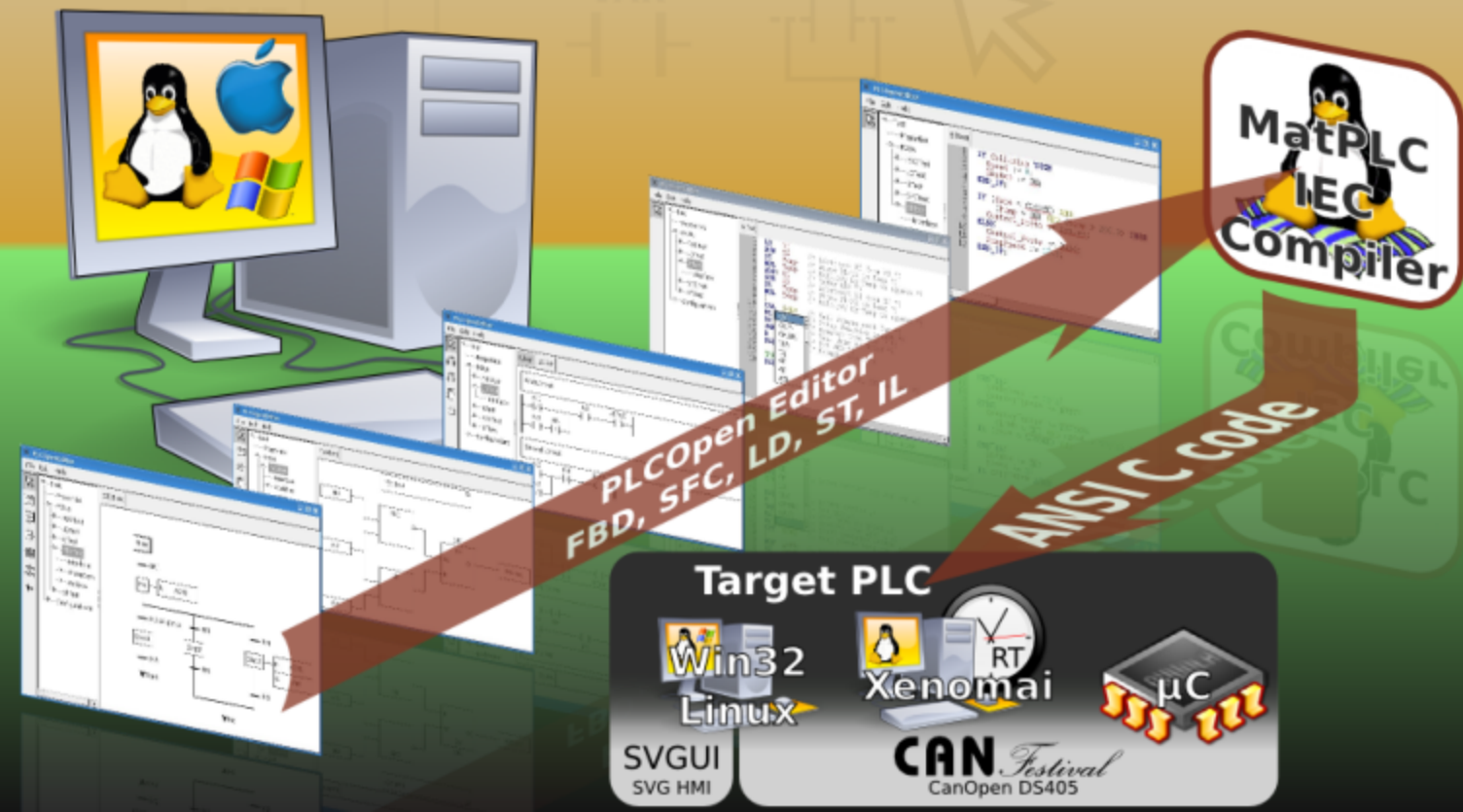
At the bottom of the window, the status bar shows: Index: 0x1A01 Subindex: 0x00 Transmit PDO 2 Mapping: Optional entry of struct REC possibly defined 512 times.

3. CanFestival CanOpen stack

- CanFestival's virtual IO block GUI lets users simulate and stimulate SoftPLC without any specific hardware

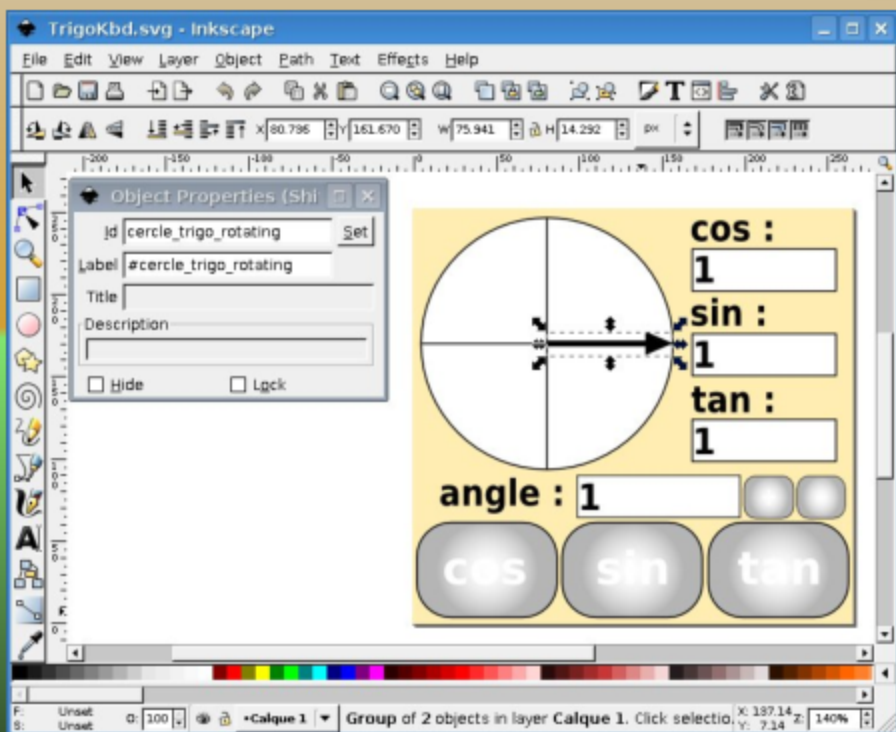


4. SVGUI : The SVG HMI toolkit

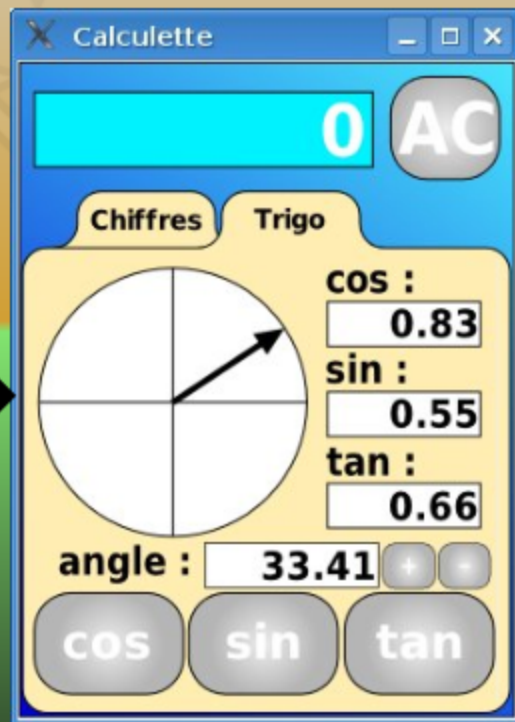


4. SVGUI : The SVG HMI toolkit

- SVGUI lets user “draw” and “skin” GUIs



INKSCAPE
(drawing program)



HMI

4. SVGUI : The SVG HMI toolkit

Def Editor

File Edit

- window
 - boutonAC
 - nombre
 - cercle_trigo
 - panel
 - scrollbar
 - text3

| attribute | value |
|---------------|---------------|
| background_id | boutonAC_back |
| unselected_id | |
| selected_id | boutonAC_on |
| name | boutonAC |

boutonAC

SVGUI_Button

```

graph LR
    En[En] --- B1[BOOL]
    Show[Show] --- B2[BOOL]
    Toggle[Toggle] --- B3[BOOL]
    Eno[Eno] --- B4[BOOL]
    Visible[Visible] --- B5[BOOL]
    State[State] --- B6[BOOL]
  
```

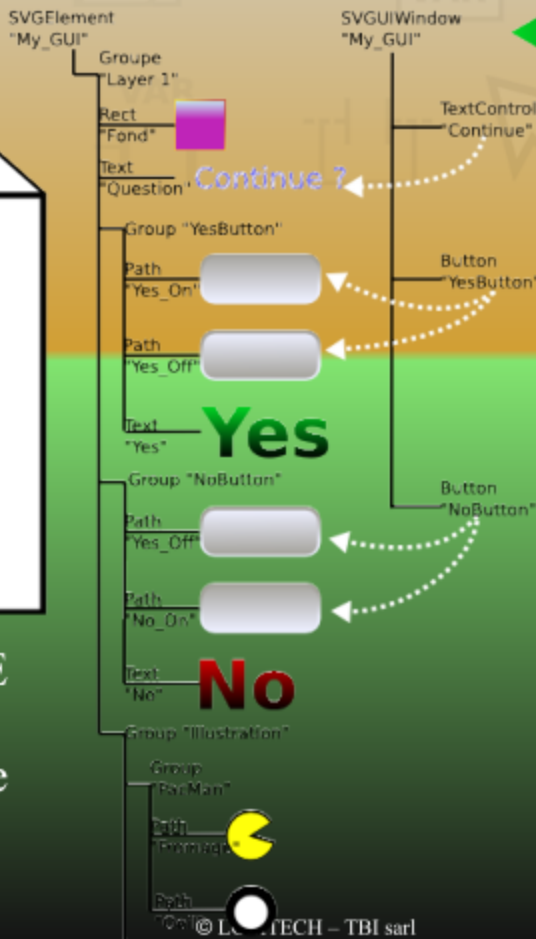
Pick up an element in the SVG View to set the value

4. SVGUI : The SVG HMI toolkit

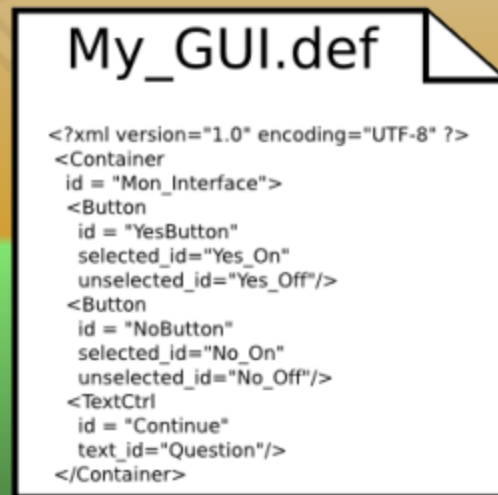
wxSVG



Edit with INKSCAPE
(SVG based drawing
program, OpenSource
and multi-platform)

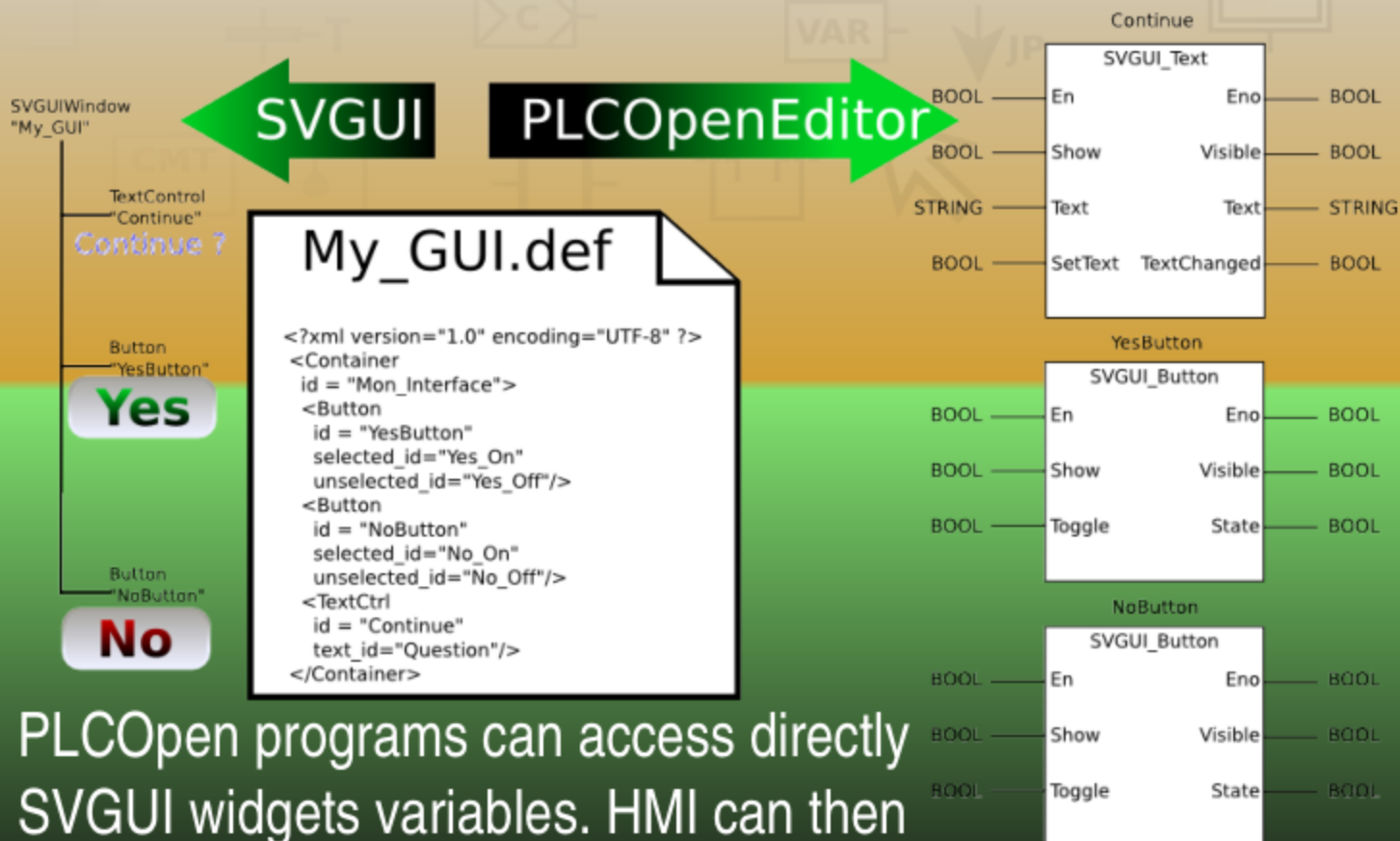


SVGUI



Edit with DefEditor
(comes with SVGUI)

4. SVGUI : The SVG HMI toolkit



What next ?

- Thanks to Open Standards, Beremiz will bring automation to everyone's use
- Control engineer, researchers and hobbyists will share automation the same way free software community shares source code. Public repositories will appear
- Beremiz will be used for teaching, implying long term adoption of PLCOpen, IEC-61131, CanOpen and SVG.
- Automation “vendor lock-in” will be a user choice

Current project status



- On the road to stability :
 - PLCOpenEditor , IEC to C and SVGUI's are experimental, they prove the concept, but need improvements
 - 2.5 man-years needed to provide a production release



- Time to join and contribute :
 - Source is available on public CVS
 - Sponsoring, donation, patronage or funding are needed
 - Shared automation repositories need your blocks

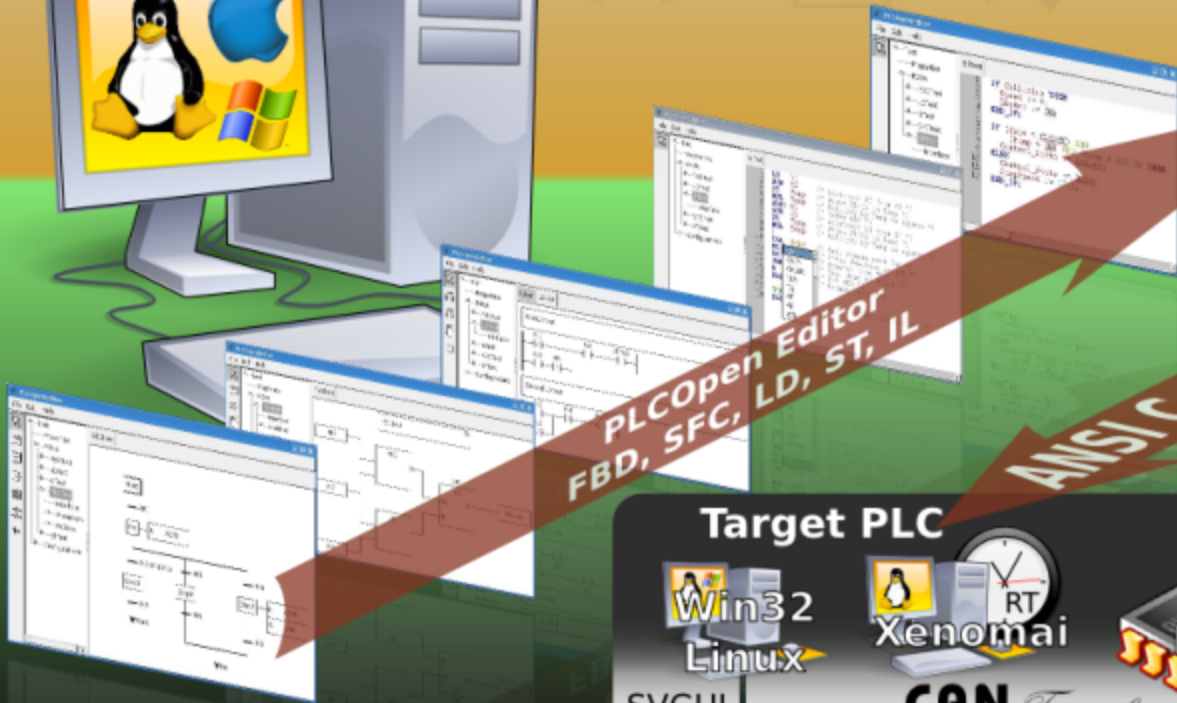
Behind the project



- Lolitech is a French company, held in 2005
- Our business model is based on Free Software
- Our goal is to bring Free Software to Industry
- We provide early support for the project, and manage community contributions
- Once official, the Beremiz foundation will drive, finance and represent the project

Beremiz

Open Source Software for Automation



PLCOpen Editor
FBD, SFC, LD, ST, IL

ANSI C code

Target PLC



SVGUI
SVG HMI

CAN Festival
CanOpen DS405

XML RPC

Python

LOLITech