



Pogo Roadmap proposal

Damien Lacoste

September 15 2021

Pogo roadmap

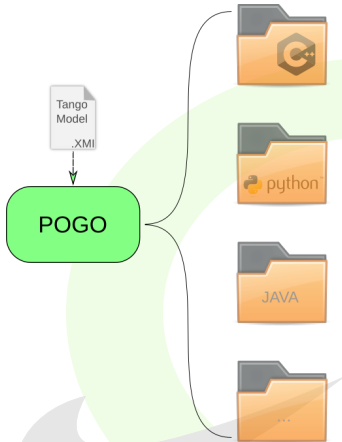
Pogo

Roadmap proposal

Coming next...


controls

POGO - Program Obviously used to Generate tango Objects



- ▶ Tango code generator written in java swing.
- ▶ Use a xmi file to describe tango class and device server model.
- ▶ Support device servers with several classes.
- ▶ GUI to create and manage tango class and device server model.
- ▶ Support generators for different languages and uses:
 - ▶ C++ code generator.
 - ▶ java code generator.
 - ▶ Python code generator, in 2 flavors.
 - ▶ HTML documentation generator.
- ▶ Run with java 11+.

Pogo roadmap



Pogo
Roadmap proposal
Coming next...

controls

- ▶ C++ generator code reorganization
 - ▶ Closer to Tango model.
 - ▶ Strong cmake integration.
 - ▶ Build libraries and binaries.
- ▶ CI/CD.
 - ▶ Clear release procedure.
 - ▶ Use of gitlab CI/CD pipelines to test.
- ▶ Get rid of "protected region" paradigm.
- ▶ Documentation.
- ▶ And maybe more...

controls

C++ code generator refactoring

- ▶ All files in the same directory.
- ▶ Differences between device server with one or many classes.
- ▶ Old cmake support.
- ▶ No possibility to build a library.

```
± tree
.
├── ClassFactory.cpp
├── CMakeLists.txt
├── main.cpp
├── Makefile
├── TestClass.cpp
├── TestClass.h
├── Test.cpp
├── Test.h
├── TestStateMachine.cpp
└── Test.xml

0 directories, 10 files
```

C++ code generator refactoring

```
± tree
.
├── classes
│   └── Test
│       ├── CMakeLists.txt
│       ├── include
│       │   ├── TestClass.h
│       │   └── Test.h
│       └── src
│           ├── TestClass.cpp
│           ├── Test.cpp
│           └── TestStateMachine.cpp
├── CMakeLists.txt
├── doc
├── servers
│   └── Test
│       ├── CMakeLists.txt
│       └── src
│           ├── ClassFactory.cpp
│           └── main.cpp
├── test
└── Test.xmi

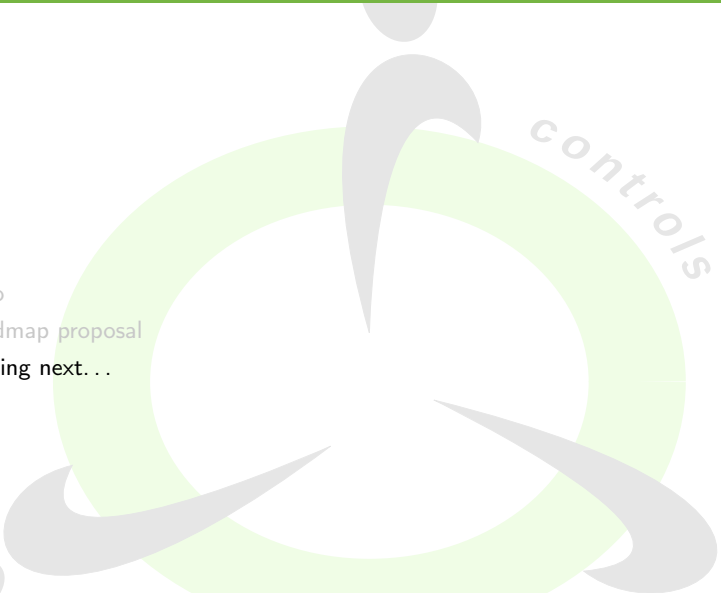
9 directories, 11 files
```

- ▶ Clear separation between headers and source files.
- ▶ We can see the tango model structure of the classes and device servers.
- ▶ All projects are managed the same, whatever the number of classes they ship.
- ▶ Modern cmake integration with targets, and support for cmake installation.
- ▶ Full control about what to build, may it be a library, binaries, linking to static or shared library, activating support for part of the classes or not through the use of consistent cmake flags.

- ▶ C++ generator code reorganization
 - ▶ Closer to Tango model.
 - ▶ Strong cmake integration.
 - ▶ Build libraries and binaries.
- ▶ CI/CD.
 - ▶ Clear release procedure.
 - ▶ Use of gitlab CI/CD pipelines to test.
- ▶ Get rid of "protected region" paradigm.
- ▶ Documentation.
- ▶ And maybe more...

controls

Pogo roadmap




Pogo
Roadmap proposal
Coming next...

controls

A few ideas

- ▶ More powerful command line interface.
- ▶ Support other input formats.
- ▶ Rename `fr.esrf.*` java classes to `org.tango.*`.
- ▶ Move away from xtend/xttext technology.
- ▶ Pogo rewrite ?

tango controls



Any Questions ?
Thanks !



Documentation

Project repository

<https://isocpp.org>

<https://www.python.org>