

Pascal Verdier & Andy Goetz - Software Group - ESRF. 6 Rue Jules Horowitz, BP 220, 38043 Grenoble, France http://www.esrf.eu /

TANGO is a distributed object oriented control system toolkit based on CORBA.It has been initially developed at ESRF.It is now also developed and used by Soleil, Elettra, Alba, Desy and many labs.Tango objects are representations of devices and support methods, attributes, properties...



History

Development diagram

This code generator (designed at the beginning of TANGO ~1999) was based on java parsing methods. It can generate Tango classes in C++, java or python



To follow many evolutions and features of TANGO control system, it was more and more difficult to maintain compatibility.

The code generator has to be re-designed using a new and efficient parsing technology.

It appears that openArchitectureWare is a good candidate.

The Parsing Technology

openArchitectureWare (oAW) is a modular MDA/MDD generator framework implemented in java(TM).

It support parsing of arbitrary models, and a language family to check and transform models as well as generate code based on them. Supporting editors are based on Eclipse platform. oAW has strong support for EMF (Eclipse Model Framework) based model.

The Model

The TANGO class model is defined with Eclipse using xtext grammar. oAW will generate a java API with classes and methods for commands, attributes, properties,...

The TANGO class code is defined by templates with Eclipse, using Xpand language. This template can contain protected zones for programmer code.

TANGO model with Xtext

Code templates with Xpand

Compatibility

There are more than 500 device classes already developed with the old code generator (225 on sourceforge, 150 on ESRF repository, many classes on each institute).

When the new code generator will be available, it will have to be able to read and translate these classes from the old model to the new one. It will also have to parse the programmers' added code, and to be able to insert it in the new generated code.

Edit Source Refactor Navigate Search Project Bun Window Help									e Edit Navigate Search Project Bun Window Heip							
3• 🖩 🗁] 🏇• O• Q₂•] 😕 🛱 O•] 🧶 😂 🔗•] ∮]+ ∮]+ ∮]+ ∮]+ ↓ ↔ ↔																
Package Explorer 🔀	- 0 1	pogo.xtxt 🛿 🚺	Main.xpt 🔀 attributes.xpt	CppUtil.ext Commands.xpt	🖸 cppDevice.xpt 🛛 🔂 initia	alize.xpt [»] 6	- 0	4	■ Package Explorer 🛛 🗖 🗖	I pogo.xtxt 🔂 Main.xpt	🔂 attributes.xpt 🛿	CppUtil.ext	🔂 commands.xpt	C cppDevice.xpt	initialize.xpt	
E 😫	© ▼ 	"class (c	Poperties:" lassProperties+=Property)	<pre>// Class property list *</pre>					달 😨 🔽	<pre>«IMPORT pogo» «EXTENSION fr::esrf::ta «EXTENSION fr::esrf::ta</pre>	:ango::pogo::cpp::_C :ango::pogo::cpp::_C	ppConstants» ppUtil»				
⊽ ∰ src		"devic	ePoperties:"	// Device Property list			6	5	▶ 🥵 src							
🕨 🗄 fr.esrf.tango.pogo		((evicerropercies+=rropercy)+			_		👂 🛋 JRE System Library [linux]	«DEFINE cppFile FOR Po	goDeviceClass»					
👂 🌐 fr.esrf.tango.pogo.parser		" comma	nds:"	<pre>// Device command list</pre>					Plug-in Dependencies	<pre>«IF attributes.size > ///////////////////////////////////</pre>	θ»					
🗢 🌐 bodo		((onmands+=conmand)+						▽ 🥵 src-gen	/**						
Argument.java		"attri	butes:"	<pre>// Device attribute list</pre>					🕨 🌐 fr.esrf.tango.pogo	* method : «na	ame»::read_attr_har	dware				
Attribute.java		(6	(() IDulest=Allibule)+						🕨 🔠 fr.esrf.tango.pogo.parser	* description : Ha	rdware acquisition	for attributes.				
BooleanArray.java	=	"state	s: *	<pre>// Device state list</pre>					🕨 🌐 fr.esrf.tango.pogo.resource	*/						
BooleanArrayType.java		-}-;	(ates+=state)*						debugMeWithAntlrWorks.g	void «name»::read_attr	_hardware(vector <lo< td=""><td>ng> &attr_list)</td><td></td><td></td><td></td></lo<>	ng> &attr_list)				
BooleanType.java									Dep META-INF	{					n .	
CharArrayType.java		11							📆 build.properties	«separator1()»	mame»::read_attr_na	rdware(vector <td< td=""><td>ng> wattr_tist) er</td><td>rtering < << en</td><td>ac;</td></td<>	ng> wattr_tist) er	rtering < << en	ac;	
ClassDescription.java		// Class info	rmation						📄 plugin.properties	«PROTECT CSTART '/	* CEND '*/' ID are	aProtectedID("re	ad_attr_hardware")	35		
Command.java		⊖ClassDescripti	.on :						🔂 plugin.xml	«ENDPROTECT»						
Comments.java		description	n = STRING // Class d	description					> 🥵 fr.esrf.tango.pogo.editor	«separator1()»						
ConstString.java		title sourcePath	= STRING // Short d = STRING // Files l	lescription Location					7 🤔 fr.esrf.tango.pogo.generator	3						
ConstStringType.java		"inheritan	ces: "						JRE System Library [J2SE-1.5]	«FOREACH attributes AS	; att»					
Description.java		(inher Language	<pre>itances+=Inheritance)*</pre>	<pre>// inheritance class definitions ae to generate</pre>					Plug-in Dependencies	<pre>«separator2()» /**</pre>						
DevintType.java		generate	= STRING // File(s)	to generate (code, makefile,)					▽ 🥭 src	* Read «att.name» at	tribute					
DoubleArrayType.java		comments	= Comments;						Fr.esrf.tango.pogo	<pre>* «att.description.co *</pre>	comments(" * ")»					
DoubleStringArrayType	.java	11							Fr.esrf.tango.pogo.cpp	* Attr type: «att.a	ttType» «att.attTyp	eDimentions()»				
DoubleType.java		// Inheritanc	e Definition						fr.esrf.tango.pogo.cpp.device	* Data type: «att.da */	lataType.cppType().t	oString()»				
DoubleVectorType.java		⊖Inheritance:							Fr.esrf.tango.pogo.cpp.dserver	«separator2()»						
EncodedType.java		classname sourcePath	= STRING						META-INF	<pre>void «name»::read_«att {</pre>	name»(Tango::Attri	bute &attr)				
EncodesType.java		//	- STRANG,						🗟 build.properties	DEBUG_STREAM << *«	name»::read_«att.na	me»(Tango::Attri	bute &attr) enter:	ing * << endl;		
FloatArrayType.java		// Comment De	finition						🖻 🔂 fr.esrf.tango.pogo.ui	«separatorl()» «PROTECT_CSTART_'/	*' CEND '*/' TD pro	tectedID(att.	read") »			
FloatType.java		⊖Comments:								, norect conner ,	(Ling / 10 pro					
FloatVectorType.java		commandsTa	ble = STRING;							attr.set_value(att «ENDPROTECT»	r_watt.name>_readwa	tt.attSetValueSi	ze()»);			
Import.java		11								«separator1()»						
Inheritance.java		// State Defi	nition							<pre>}«IF att.rwTtype.conta. "concreter2()"</pre>	ins("WRITE")»					
InheritedFrom.java		State:								/**						
InputArg.java		name	= ID							* Write «att.name» a	ttribute values to	hardware.				
IntArrayType.java		descriptio	m = STRING;							* Attr type: «att.a	ttType» «att.attTyp	eDimentions()»				
IntType.java		//								* Data type: «att.d	lataType.cppType().t	oString()»				
IntVectorType.java		// Property D	erinition							«separator2()»						
LongArrayType.java	-	Property:					•			<pre>void «name»::write_«at</pre>	t.name»(Tango::WAtt	ribute &attr)				
III		4		III			>			•						
•						8	@ 🔂 🖪		D°		Writa	ble Insert	58 : 27			

The model and the templates can be exported in a jar file and interfaced with any java application (e.g. GUI)

Software Group – ESRF, 6 rue Jules Horowitz, BP220, 38043 GRENOBLE CEDEX, FRANCE Tel +33 (0)4 76 88 20 00 - Fax +33 (0)4 76 88 20 20

Conclusions

openArchitectureWare is very powerful code generator tool for object oriented control system.

The development of the device class code generator for TANGO has started few months ago. Everything has not been yet tested, but the first results are very encouraging in term of code readability, and for further evolutions and maintenance.

> openArchitectureWare (oAW) Modular MDA/MDD Generator Framework http://www.openarchitectureware.org/