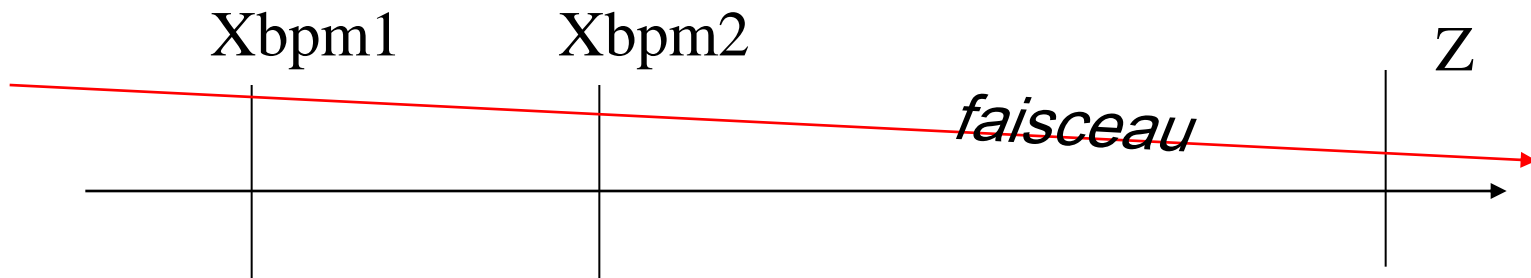


# **TangoParser**

## **Ou**

# **Device de calcul**

- Fournir de **nouveaux attributs** TANGO qui soient le résultat de calculs à partir **d'attributs déjà existants**
- Exemple :
  - ➡ Je calcule la position du faisceau au Point Z en fonction des mesures effectuées sur les xbp1 et xbp2 (mesures déjà disponibles dans TANGO)



- Les attributs à lire dans le système Tango sont décrits par la propriété **AttributeNames**

➔ *Exemple :*

```
p1,TDL-D13-1/DG/XBPM.1/verticalPosition1  
p2,TDL-D13-1/DG/XBPM.1/verticalPosition2  
p3,TDL-D13-1/DG/XBPM.2/verticalPosition1  
p4,TDL-D13-1/DG/XBPM.2/verticalPosition2
```

➔ *Explication :*

- ▶ La « *variable* » *p1* va maintenant contenir le résultat de la lecture de l'attribut *TDL-D13-1/DG/XBPM.1/verticalPosition1*

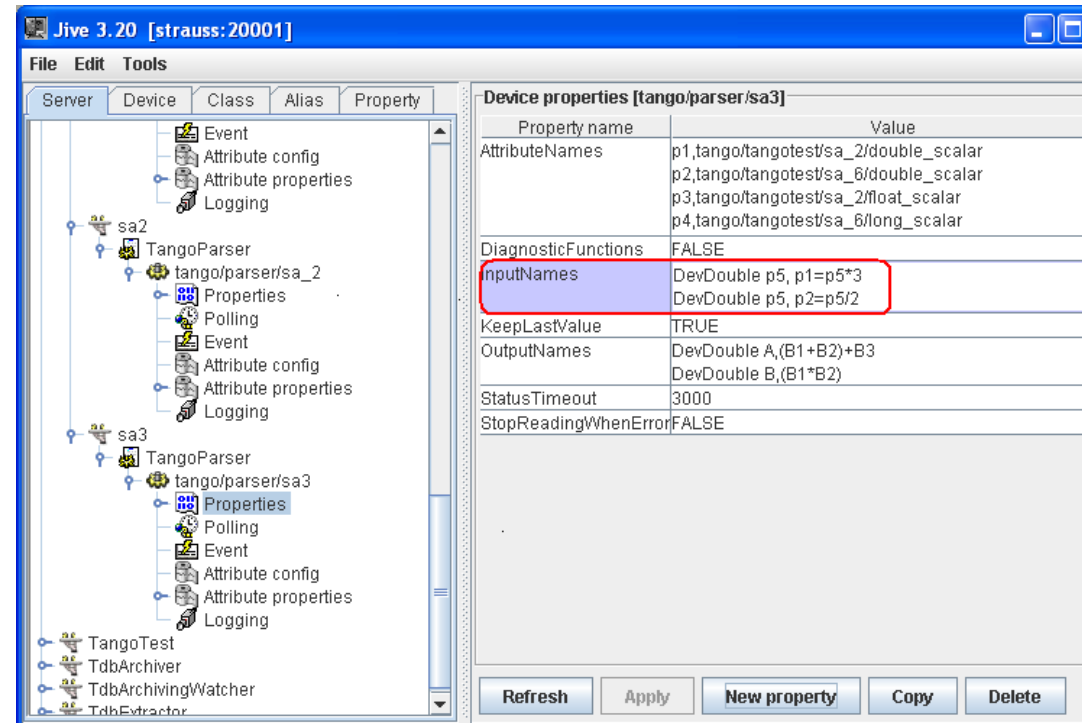
- Les nouveaux attributs dynamiques en écriture sont dans la propriété **InputNames**

— Exemple :

DevDouble p5,  $p1=p5*3$   
DevDouble p5,  $p2=p5/2$

— Explication :

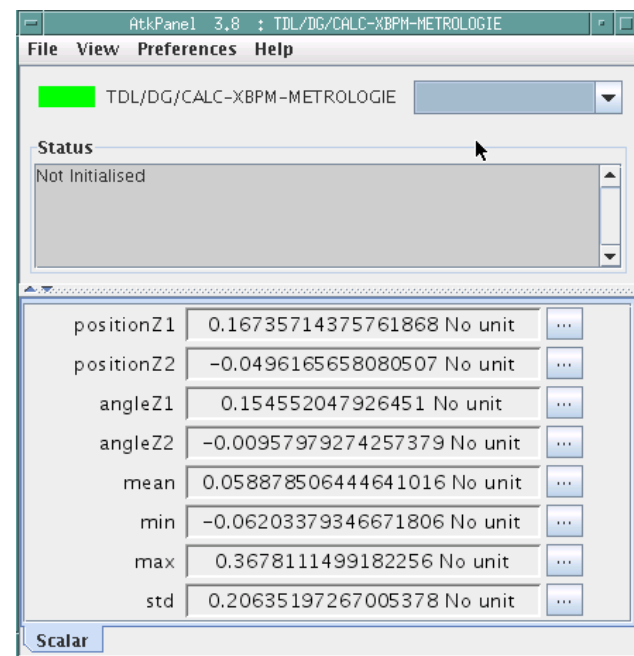
- Créer un nouveau attribut p5,
- A chaque fois qu'un client écrit dans l'attribut p5, le Parser écrit dans les attributs p1 et p2 les nouvelles valeurs calculées en fonction de p5 et selon les expressions définies dans InputNames.



- Les nouveaux attributs en lecture sont décrits par la propriété **OutputNames**

– *Exemple :*

```
DevDouble positionZ1, (p1+p3) / 2
DevDouble positionZ2, (p2+p4) / 2
DevDouble angleZ1, (p3-p1) / 3.030
DevDouble angleZ2, (p4-p2) / 3.030
```



– *Explication :*

- 4 nouveaux attributs de type *DevDouble* sont maintenant disponibles dans TANGO

- Les nouveaux attributs en écriture et en lecture sont décrits par la propriété **IONames**

– *Exemple :*

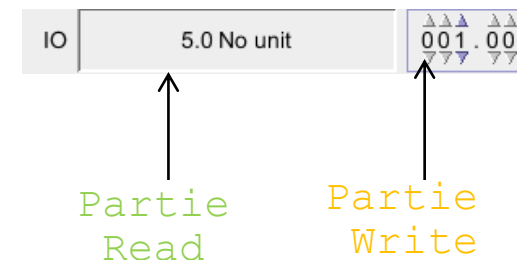
```
DevDouble IO;p1+p2;p1=IO+1;p2=IO+2
```



« ; » est le séparateur

– *Explication :*

- Un nouvel attribut est disponible. Sa partie read contiendra le résultat de « *p1+p2* » et lorsque l'on écrira dessus, le Parser écrira sur les attributs *p1* et *p2*.
- On ne peut définir qu'une seule expression (la première) pour la lecture. Et autant que souhaitées pour l'écriture.





- On peut créer de nouvelles variables (non définies dans AttributeNames) dans **InputNames** ou **IONames**:

- Exemple en créant les variables « aux1 » et « coef »:*

```
nbbin;aux1;aux1=nbbin
```

```
detcRW;10*(coef^(100/aux1));coef=detcRW
```

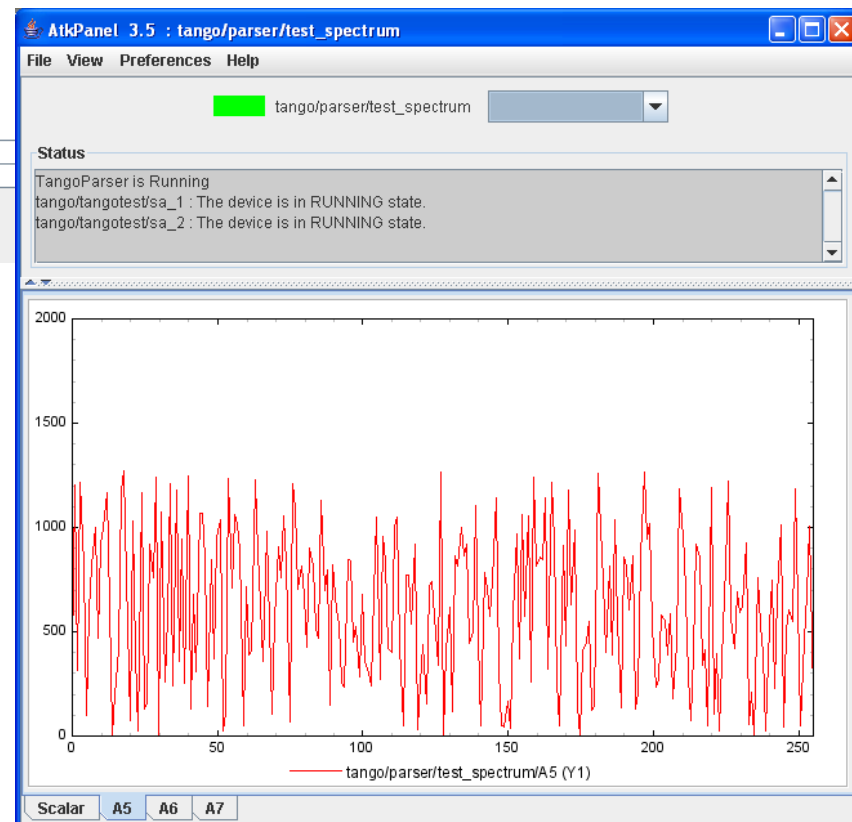
nbbin	10.0 No unit	010.00
detcRW	9.765625E7 No unit	005.00

- Explication :*

- 2 attributs en Read/Write
- Chaque écriture sur detcRW ou nbbin, modifie la partie read de detcRW.
- On peut réutiliser les variables créées dans **OuputNames**.

Device properties [tango/parser/test_spectrum]	
Property name	Value
AttributeNames	B1,tango/tangotest/sa_1/State B2,tango/tangotest/sa_2/double_spectrum_ro
DiagnosticFunctions	FALSE
InputNames	
KeepLastValue	TRUE
OutputNames	DevBoolean A1,B1>5  B1<11 DevBoolean A2,B1>11 DevState A3,B1 DevDouble A4,ysum(B2) SPECTRUM DevDouble A5,B2*5 SPECTRUM DevDouble A6,B2+B2 SPECTRUM DevDouble A7,fit(B2)
StatusTimeout	3000
StopReadingWhenError	FALSE

- Un nouvel attribut A5 est créé et est le résultat de la multiplication de B2 par un scalaire
  - B2 est un attribut spectrum READ-ONLY relu sur un Device Tango
- Un nouvel attribut A6 est la somme de 2 attributs spectrum





■ **Formats:** SCALAR, SPECTRUM.

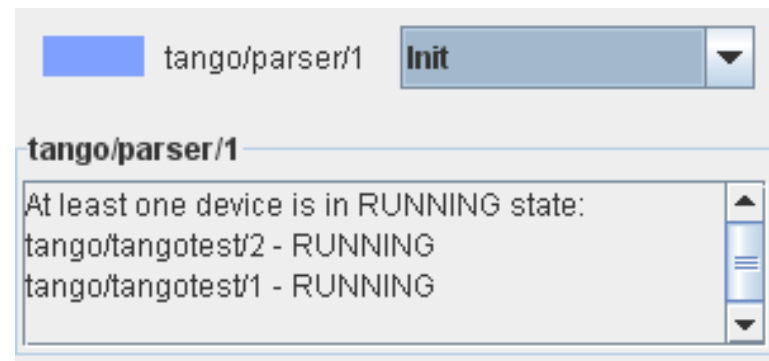
➡ Optionnel. Valeur par défaut: SCALAR.

➡ Seulement pour OutputNames.

■ **Types:** DevBoolean, DevShort, DevLong, DevFloat, DevDouble, DevUShort, DevULong, DevString, State, DevUChar, DevLong64, DevULong64

➡ Optionnel. Valeur par défaut: DevDouble

- Le parser indique maintenant le State de chacun des Devices utilisés pour les calculs
- L'état du parser prend le state avec la priorité la plus élevée. (Voir propriété : PriorityList)



■ Un attribut « Errors », donne la liste des erreurs.

➡ La 1ere colonne indique l'attribut en erreur

➡ La 2eme colonne indique le message d'erreur.

tango/parser/1

**tango/parser/1**

ERROR present, check error attribute for more details  
 At least one device is in RUNNING state:  
 tango/tangotest/2 - RUNNING  
 tango/tangotest/1 - RUNNING

	0	
0	testRW2	INIT Error Level 0: - desc: double_scala attribute not found - origin: MultiAttribute::get_attr_by_name - reason: API_AttrNotFound - severi
1	testRWbool	INIT Error Level 0: - desc: double_scala attribute not found - origin: MultiAttribute::get_attr_by_name - reason: API_AttrNotFound - severi
2	hello2	Error Level 0: - desc: org.nfunk.jep.ParseException: Could not evaluate v: variable not set - origin: [org.nfunk.jep.EvaluatorVisitor.visit(Ur
3	testRW3	INIT Error Level 0: - desc: double_scala attribute not found - origin: MultiAttribute::get_attr_by_name - reason: API_AttrNotFound - severi

Scalar errors

- Autres propriétés :
  - ScanMode:
    - TRUE : lecture des attributs et de l'état de façon synchrone
    - FALSE : lecture de façon asynchrone
  - PeriodRefresh
    - Délai de rafraîchissement des attributs et état en mode asynchrone en ms.
  - DiagnosticFunctions
    - TRUE : le parser ajoute le calcul des fonctions MIN, MAX, MEAN, STD sur les AttributesNames
  - MovingState
    - indique au ScanServer l'état auquel il doit s'attendre lorsque le device est en « moving »
  - PriorityList
    - Liste des états selon leur priorité. Cela permet de composer les états des différents devices appelés et de remonter le plus important.

Property name	Value
AttributeNames	spec,tango/tangotest1/double_spectrum A1,tango/tangotest1/short_scalar_w B1,tango/tangotest1/short_spectrum C1,tango/tangotest1/string_scalar A2,tango/tangotest2/short_scalar_w B2,tango/tangotest2/short_scalar C2,tango/tangotest2/string_scalar D1,tango/tangotest2/boolean_scalar vx,tango/tangotest1/double_scalar
DiagnosticFunctions	TRUE
inputNames	DevDouble Y,A1=Y+1 DevDouble Y,B2=Y+2
IONames	DevDouble IO,A1+A2,A1=IO+1,A2=IO+2
OutputNames	
PeriodRefresh	3000
PriorityList	UNKNOWN,3 FAULT,2 ALARM,1
ScanMode	FALSE

Affichage de  
l'expression de  
l'attribut dans le  
champ description

**Attribute property editor**

Identification

Device: tango/parser/1

Attribute: testRW2

**83.34602205026455 No** 000.00

Properties

Label: testRW2

Minimum value: Not specified Maximum value: Not specified

Minimum alarm: Not specified Maximum alarm: Not specified

Format: Unit: No unit

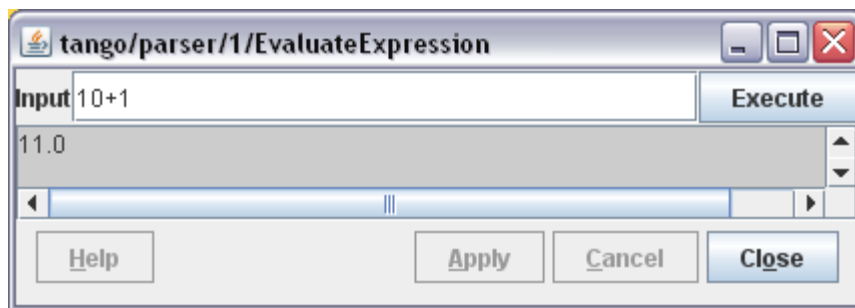
Min. warning: Not specified Max. warning: Not specified

Delta t(ms): Not specified Delta Val: Not specified

expression read: a+b  
expressions write: [a=testRW2, b=testRW2\*2]

Apply change Information Dismiss

- La commande EvaluateExpression retourne l'évaluation de l'expression passée en argument d'entrée.





# Les fonctions mathématiques disponibles: Scalaires : Opérateurs

Power	$\wedge$
Boolean Not	!
Unary Plus, Unary Minus	+x, -x
Dot product, cross product	., ^^
Modulus	%
Division	/
Multiplication	*
Addition, Subtraction	+, -
Less or Equal, More or Equal	<=, >=
Less Than, Greater Than	<, >
Not Equal, Equal	!=, ==
Boolean And	&&
Boolean Or	
Assignment	=

# Les fonctions mathématiques disponibles: Scalaires: Fonctions

## Trigonometric Functions

All functions accept arguments of the `Double` and `Complex` only accepts `Double` arguments.

Description	Function Name
Sine	<code>sin(x)</code>
Cosine	<code>cos(x)</code>
Tangent	<code>tan(x)</code>
Arc Sine <sup>2</sup>	<code>asin(x)</code>
Arc Cosine <sup>2</sup>	<code>acos(x)</code>
Arc Tangent	<code>atan(x)</code>
Arc Tan with 2 parameters	<code>atan2(y, x)</code>
Secant	<code>sec(x)</code>
Cosecant	<code>cosec(x)</code>
Co-tangent	<code>cot(x)</code>
Hyperbolic Sine	<code>sinh(x)</code>
Hyperbolic Cosine	<code>cosh(x)</code>
Hyperbolic Tangent	<code>tanh(x)</code>
Inverse Hyperbolic Sine	<code>asinh(x)</code>
Inverse Hyperbolic Cosine <sup>1</sup>	<code>acosh(x)</code>
Inverse Hyperbolic Tangent <sup>1</sup>	<code>atanh(x)</code>

# Les fonctions mathématiques disponibles: Scalaires: Fonctions

## Log and Exponential Functions

All functions accept arguments of the Double and Complex

Description	Function Name
Natural Logarithm <sup>1</sup>	<code>ln(x)</code>
Logarithm base 10 <sup>1</sup>	<code>log(x)</code>
Logarithm base 2 <sup>1</sup>	<code>lg(x)</code>
Exponential ( $e^x$ )	<code>exp(x)</code>
Power <sup>1</sup>	<code>pow(x)</code>

## Statistical Functions

All functions accept either a vector (e.g. `min([1,2,3])`) or `min(1,2,3)`.

Description	Function Name
Average	<code>avg(x1,x2,x3,...)</code>
Minimum	<code>min(x1,x2,x3,...)</code>
Maximum	<code>max(x1,x2,x3,...)</code>

## Rounding Functions

Description	Function Name
Round	<code>round(x)</code> , <code>round(x, p)</code>
Floor	<code>floor(x)</code>
Ceiling	<code>ceil(x)</code>

# Les fonctions mathématiques disponibles: Scalaires: Fonctions

## Miscellaneous Functions

Description	Function Name
If	if(cond, trueval, falseval)
Str (convert number to string)	str(x)
Absolute Value / Magnitude	abs(x)
Random number (between 0 and 1)	rand()
Modulus	mod(x, y) = x % y
Square Root <sup>1</sup>	sqrt(x)
Sum	sum(x, y, ...)
Binomial coefficients	binom(n, i)
Signum (-1, 0, 1 depending on sign of argument)	signum(x)

## Complex Functions

Description	Function Name	
Real Component	re(c)	<a href="#">R</a>
Imaginary Component	im(c)	<a href="#">I</a>
Complex Modulus (Absolute Value)	cmod(c)	<a href="#">A</a>
Argument (Angle of complex value, in radians)	arg(c)	<a href="#">A</a>
Complex conjugate	conj(c)	<a href="#">C</a>
Complex, constructs a complex number from real and imaginary parts	complex(x, y)	<a href="#">C</a>
Polar, constructs a complex number from modulus and argument	polar(r, theta)	<a href="#">P</a>

## String Functions

The following string functions are not included in the standard configuration, however they can be added by loading the [StringFunctionSet](#) component. To add them, use `jep.setComponent(new StringFunctionSet())` before parsing. Individual functions can also be added using `jep.addFunction()`.

Description	Function Name
Left	<code>left(str, len)</code>
Right	<code>right(str, len)</code>
Middle	<code>mid(str, start, len)</code>
Substring	<code>substr(str, start, [end])</code>
Lower Case	<code>lower(str)</code>
Upper Case	<code>upper(str)</code>
Length	<code>len(str)</code>
Trim	<code>trim(str)</code>

## Optional Functions

These functions are not included in the standard configuration, but can be added using `jep.addFunction()`.

In `com.singularsys.jep.misc.functions`:

Class	Description
<a href="#">LogTwoArg</a>	Two argument log function where second argument is the base.
<a href="#">Remainder</a>	Calculates the remainder and quotient of the arguments. Constructors allow different conventions for remainder to be used.
<a href="#">RoundSF</a>	Rounds arguments to a specific number of significant figures.
<a href="#">ToBase</a>	Converts numbers to a string in a given base.
<a href="#">FromBase</a>	Converts a string in a given base to numbers.
<a href="#">Switch</a>	A switch statement. Returns the value of the argument based on the value of the first argument.
<a href="#">SwitchDefault</a>	A switch statement with a default value as final argument.
<a href="#">Case</a>	A case statement, first argument is test condition, following arguments are in pairs with a test value and corresponding result.
<a href="#">IsNull</a>	Tests if the argument is null.
<a href="#">IsNaN</a>	Tests if the argument is NaN.
<a href="#">IsInfinite</a>	Tests if the argument is infinite.
<a href="#">IsType</a>	Test if the argument is of the type specified in the constructor. E.g. <code>isDouble(x)</code>

# Les fonctions mathématiques disponibles: Spectrum: Fonctions

Name	Description	Examples
length	Finds the length of a vector or the number of elements in a matrix or tensor.	<code>length(5)==1, length([1,2,3])==3, length([ [1,2], [3,4] ])==4</code>
size	Finds the size of a vector, matrix or tensor.	<code>size(5)==1, size([1,2,3])==3, size([ [1,2], [3,4], [5,6] ])==[2,3]</code>
det	The determinant of a square matrix.	<code>det([ [1,2], [3,4] ])==-2</code>
trace	The trace of a square matrix.	<code>trace([ [1,2], [3,4] ])==5</code>
vsum	The sum of elements in a vector, matrix or tensor.	<code>vsum([1,2,3])==6, vsum([ [1,2], [3,4] ])==10</code>
trans	The transpose of a matrix.	<code>trans([ [1,2], [3,4] ])==[ [1,3], [2,4] ]</code>
getdiag	Extracts the diagonal from a square matrix.	<code>getdiag([ [1,2], [3,4] ])==[1,4]</code>
diag	Generates a square matrix with a given diagonal.	<code>diag([1,2,3])==[ [1,0,0], [0,2,0], [0,0,3] ]</code>

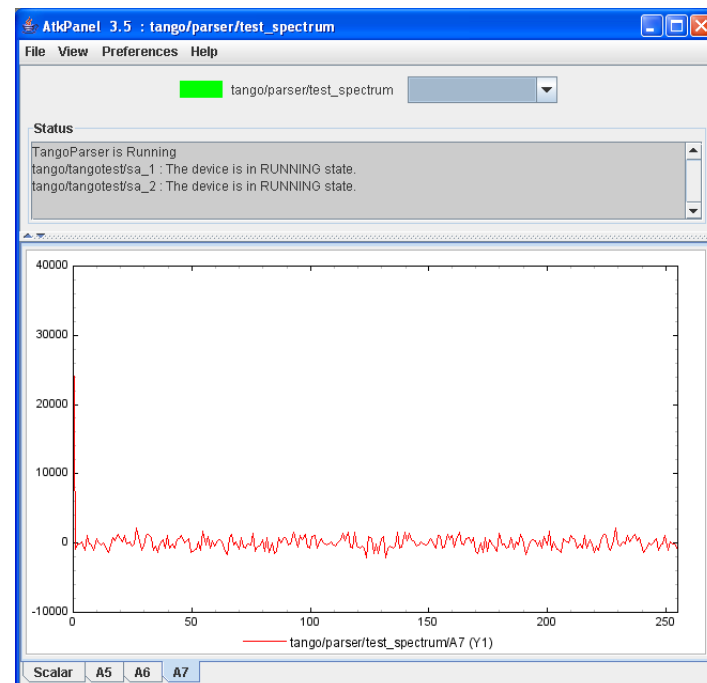
GenMat	Generates vectors and matrixes	<p>First argument specifies size of the vector (3) and matrixes ([2,2]). Second argument is formula for each element, constants (1), functions (rand()), Third argument (if present) is list of variables used in formula. For vectors a single variable is specified whos value runs from 1 to number of elements. For matrixes a two variable list ([ii,jj]) is specified whos value are the column and row indicies.</p> <p><code>GenMat(3,1) -&gt; [1,1,1]</code>  <code>GenMat(3,ii,ii) -&gt; [1,2,3]</code>  <code>GenMat(3,rand()) -&gt; [0.343,0.974,0.567]</code>  <code>GenMat([2,2],ii+jj,[ii,jj]) -&gt; [[2,3],[3,4]]</code></p>
ele	Extracts an element from a vector, matrix or tensor.	<code>ele([1,2,3],2)==2, ele([ [1,2], [3,4] ], [1,2])==2</code>

The Map function applies a function to each element of a vector or matrix



Device properties [tango/parser/test_spectrum]	
Property name	Value
AttributeNames	B1,tango/tangotest/sa_1/State B2,tango/tangotest/sa_2/double_spectrum_ro
DiagnosticFunctions	FALSE
InputNames	
KeepLastValue	TRUE
OutputNames	DevBoolean A1,B1>5  B1<11 DevBoolean A2,B1>11 DevState A3,B1 DevDouble A4,ysum(B2) SPECTRUM DevDouble A5,B2*5 SPECTRUM DevDouble A6,B2+B2 SPECTRUM DevDouble A7,fft(B2)
StatusTimeout	3000
StopReadingWhenError	FALSE

- Un nouvel attribut A7 est crée et est le résultat de la **Transformée de Fourier** de l'attribut spectrum B2



## ■ AttributeNames:

➔ B2,tango/tangotest/sa\_2/double\_spectrum\_ro

## ■ OutputNames:

➔ SPECTRUM DevDouble A1,vlog(B2)

▶ Chaque point du spectrum A1 contient le log du point de même indice du spectrum B2

➔ SPECTRUM DevDouble A2,vcos(B2)

▶ Chaque point du spectrum A2 contient le cos du point de même indice du spectrum B2

➔ SPECTRUM DevDouble A3,vsin(B2)

▶ Chaque point du spectrum A3 contient le sin du point de même indice du spectrum B2

➔ SPECTRUM DevDouble A4,add(B2,5)

▶ Chaque point du spectrum A4 contient la somme du scalaire de valeur 5 et du point de même indice du spectrum B2

➔ SPECTRUM DevDouble A5,sub(B2,3)

▶ Chaque point du spectrum A5 contient la différence du point de même indice du spectrum B2 et du scalaire de valeur 3

- Introduire un nouveau sensor pour les Scans tel que :
  - ➡ le rapport  $I/I_0$
  - ➡ Le rapport  $\log(I/I_0)$
  - ➡ etc ..
- Définir de nouvelles conditions d'alarme sur le système de contrôle

## ■ Scalaires :

➡ <http://www.singularsys.com/jep/doc/html/index.html>

## ■ Spectrum

➡ <http://www.singsurf.org/djep/html/djep/VectorJep.html>