



**SMARTERAPP INTERFACE SPECIFICATION**  
for  
**ITEM SCORING CUSTOM OPERATORS**  
**American Institutes for Research**

Revision History

| Revision Description                            | Author/Modifier   | Date              |
|---|---|-------------------|
| 0.1 - First Draft Release                       | Jon Cohen<br>Balaji Kodeswaran<br>David Lopez de Quintana | November 17, 2014 |
| 0.2 – Added four additional EQ custom operators | Balaji Kodeswaran<br>David Lopez de Quintana              | November 19, 2014 |
|   |   |                   |
|   |   |                   |
|   |   |                   |



## Contents

|   |    |
|---|----|
| Introduction .....                        | 3  |
| CTRL Custom Operators .....               | 4  |
| GRID Item Custom Operators.....           | 7  |
| TABLE Item Custom Operators.....          | 13 |
| EQUATION Item Custom Operators.....       | 15 |
| Example Grid Rubric (Type = GRAPHIC)..... | 18 |
| Example Equation Rubric (Type = EQ) ..... | 33 |

## Introduction

### Overview

The SmarterApp items are divided into three categories based on how they are scored:

1. Not scored: WIT (wordlist item type) and TUT (tutorial) exist as resources to support scored items and so are not intended to be scored. They are not presented to students for scoring, and are not processed by scoring engines. No machine scoring rubrics are provided for these item types.
2. Human scored: SA (short answer), ER (extended response) and WER (writing extended response) items are intended to be human scored, so no machine scoring rubrics are provided for these item types.
3. Trivially scored: MC (multiple choice) and MS (multi-select) items are scored by comparing a student’s response with the “correct” response captured in the item XML itself.
4. Machine scored: EBSR (evidence-based selected response), HTQ (hot text – QTI) and MI (match interaction) items provide external scoring machine rubric files that are scored using QTI response processing described in an external rubric file.
5. Machine scored with custom operators: EQ (equation), GI (grid) and TI (table interaction) item types provide external scoring machine rubric files that are scored using QTI response processing that is extended by custom operators. QTI provides a means of extending default response processing via custom operators as a mechanism.
6. Probabilistically scored: ER and WER items containing essays can be scored with an open source essay scoring engine. However, unlike other machine scored items that use a scoring engine to deterministically score any compliant machine rubric, these items require a special machine rubric that is created by “training” the scoring engine with a training set of items that have been previously human scored. This type of item scoring is considered probabilistic as opposed to deterministic because (a) there must be a training process, and (b) the exact scores produced by this type scoring engine for a given student response cannot be predicted based on the contents of a rubric file, but is instead dependent on the training set used and the quality and accuracy of the human scores of the training set.

This document focuses on number 5 above, machine scored items with custom operators. It describes the custom operators used to score EQ, GI and TI item types. General QTI response processing is outside the scope of this document as this is well documented in various QTI standards (see reference 2).

### References

| Reference |  |
|-----------|--|
| 1         | SmarterApp Assessment Item Format Specification V0.80<br><a href="http://www.smarterapp.org/documents/SmarterApp_Assessment_Item_Format_Specification.pdf">http://www.smarterapp.org/documents/SmarterApp_Assessment_Item_Format_Specification.pdf</a> |
| 2         | IMS Question and Test Interoperability v2.1 Final Specification (31 August 2012)<br><a href="http://www.imsglobal.org/question/#version2.1">http://www.imsglobal.org/question/#version2.1</a>  |

## CTRL Custom Operators

This section details general custom operators that are not specific to Grid (GI), Table Interaction (TI) and Equation (EQ) Equation item types. All custom operators in this section are of type = CTRL.

### mapExpression

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | mapExpression   |
| Function Description | This applies the contained expressions to each element sequentially to each element of the collection identified in “container.” All expression must return a Boolean value. Note that the symbol “@” can be used by expressions to reference each element in the set |
| Return Type          | Container   |
| Return Description   | A container with the same basetype as the input container and an ordered cardinality  |

| Attributes | Type       | Description   |
|------------|------------|---|
| container  | Identifier | an identifier bound to a value with cardinality multiple or ordered |

### stringToFloat

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | stringToFloat   |
| Function Description | Parses a string and converts it to a floating point value                         |
| Return Type          | Float   |
| Return Description   | A numerical value based on parsing of the input string, or NaN if the parse fails |

| Attributes  | Type   | Description                             |
|-------------|--------|---|
| inputString | String | A string formatted as a numerical value |

### COUNTDOUBLEINRANGE

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | COUNTDOUBLEINRANGE  |
| Function Description | Takes a collection or comma-separated list of numeric values and counts the number of values within the specified range |
| Return Type          | Integer   |

| Custom Operator    | Description                                  |
|--------------------|--|
| Return Description | The number of doubles in the specified range |

| Attributes | Type                 | Description   |
|------------|----------------------|---|
| List       | String or identifier | either: 1) a list of values consisting of parseable numeric values or numeric variable identifiers prefaced with a "\$", or 2) an identifier bound to a variable with basetype float or integer |
| Min        | Float                | minimum value included in the range   |
| Max        | Float                | maximum value included in the range   |

## COUNTBOOL

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | COUNTBOOL  |
| Function Description | Takes a collection or list of Boolean values and counts the number that are either true or false |
| Return Type          | Integer  |
| Return Description   | The number of values that are specified  |

| Attributes   | Type                 | Description   |
|--------------|----------------------|---|
| List         | String or identifier | either: 1) a list of values consisting of parseable Boolean values or Boolean identifiers prefaced with a "\$", or 2) an identifier bound to a variable with basetype Boolean |
| booleanValue | Boolean              | true or false. The value to be matched.   |

## MAXINT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | MAXINT  |
| Function Description | Returns the maximum value of a set of integers                            |
| Return Type          | Integer   |
| Return Description   | An integer representing the maximum value in the set if integers provided |

| Attributes | Type | Description |
|------------|------|-------------|
|------------|------|-------------|

| Attributes | Type                 | Description  |
|------------|----------------------|--|
| List       | string or identifier | either: 1) a list of values consisting of parseable integer values or integer variable identifiers prefaced with a "\$", or 2) an identifier bound to a variable with basetype integer |

## MININT

| Custom Operator      | Description                                  |
|----------------------|--|
| Function Name        | MININT                                       |
| Function Description | Returns the minimum of a set of integers     |
| Return Type          | Integer                                      |
| Return Description   | The minimum value of a set of integer values |

| Attributes | Type                 | Description  |
|------------|----------------------|--|
| List       | string or identifier | either: 1) a list of values consisting of parseable integer values or integer variable identifiers prefaced with a "\$", or 2) an identifier bound to a variable with basetype integer |

## GRID Item Custom Operators

This section describes the custom operators used in response processing of SmarterApp Grid (GI) item type rubrics. All custom operators in this section are of type = GRAPHIC.

### PREPROCESSRESPONSE

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | PREPROCESSRESPONSE  |
| Function Description | Translates graphic responses to a collection of strings representing the objects in the set |
| Return Type          | basetype  |
| Return Description   | A basetype “string” with cardinality “ordered”  |

| Attributes | Type       | Description  |
|------------|------------|--|
| Response   | identifier | names the original XML response to be preprocessed |

### COUNTSIDES

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | COUNTSIDES   |
| Function Description | Counts the number of line segments that comprise the object  |
| Return Type          | Integer  |
| Return Description   | an integer representing the number of sides of the object. If the object passed in was not a graphic object or it was not an object comprised of line segments, zero is returned |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier bound to a string representing a graphic object |

### GETPOINT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETPOINT  |
| Function Description | Gets a point object corresponding to the location of a point or atomic object in the response space |
| Return Type          | Point   |

| Custom Operator    | Description   |
|--------------------|---|
| Return Description | A point in the response space, or null if an inappropriate object was passed in |

| Attributes | Type       | Description   |
|------------|------------|---|
| object     | Identifier | Identifier bound to a string representing a point or an atomic object |
| pointIndex | Integer    | The index of the specified point                                      |

### INTERSECTSREGION

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | INTERSECTSREGION   |
| Function Description | Checks whether an object intersects a rectangular region. For usability reasons intersection is defined as within 5 pixels |
| Return Type          | Boolean  |
| Return Description   | Boolean indicating whether the object intersected the region   |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier bound to a string representing a graphic object |
| topY       | Integer    | top of the region  |
| leftX      | Integer    | leftmost part of the region                                |
| bottomY    | Integer    | bottom of the region                                       |
| rightX     | Integer    | rightmost part of the region                               |

### ISREGIONSELECTED

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | ISREGIONSELECTED  |
| Function Description | Determines if a region is selected                      |
| Return Type          | Boolean   |
| Return Description   | Returns true if the region is selected, false otherwise |

| Attributes | Type | Description |
|------------|------|-------------|
|------------|------|-------------|



| Attributes | Type       | Description  |
|------------|------------|--|
| region     | Identifier | Identifier bound to a string representing a region |

### GETSELECTEDREGIONSCOUNT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETSELECTEDREGIONSCOUNT   |
| Function Description | Returns the quantity of the regions within a region group that are selected |
| Return Type          | Integer   |
| Return Description   | The number of selected regions within a region group                        |

| Attributes  | Type       | Description  |
|-------------|------------|--|
| regionGroup | Identifier | Identifier bound to a string representing a region group |

### GETSLOPE

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | GETSLOPE   |
| Function Description | Calculates the slope of a line   |
| Return Type          | Float  |
| Return Description   | A value representing the slope of the line. Returns NaN if the object is not a line. |

| Attributes | Type       | Description                                      |
|------------|------------|--|
| vector     | Identifier | Identifier bound to a string representing a line |

### GETVECTOR

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | GETVECTOR  |
| Function Description | Returns the index-th vector of a multi-vector object. Vectors are sorted starting with the vector containing the top-most, left-most point, and moving clockwise. Where more than two line segments meet at the same point, the top-most, left-most comes first. |
| Return Type          | String   |
| Return Description   | A string representation of the vector  |

| Attributes | Type       | Description  |
|------------|------------|--|
| Order      | Integer    | Index of desired vector                                    |
| Object     | Identifier | Identifier bound to a string representing a graphic object |

### ISGRAPHICTYPE

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | ISGRAPHICTYPE  |
| Function Description | Determines if the object is a graphic object of the identified type    |
| Return Type          | Boolean  |
| Return Description   | Returns true if the graphic object is an object of the identified type |

| Attributes  | Type       | Description  |
|-------------|------------|--|
| Object      | identifier | identifier bound to a string representing a graphic object               |
| graphicType | string     | must be one of the following: "PALETTEIMAGE", "VECTOR", "ARROW", "POINT" |

### GETLENGTH

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETLENGTH   |
| Function Description | Returns the length of the referenced vector   |
| Return Type          | Float   |
| Return Description   | A value indicating the length of the vector or NaN if anything but a vector is provided |

| Attributes | Type       | Description   |
|------------|------------|---|
| vector     | identifier | Identifier bound to a string representing a vector. |

### GETNAME

| Custom Operator | Description |
|-----------------|-------------|
| Function Name   | GETNAME     |

| Custom Operator      | Description  |
|----------------------|--|
| Function Description | Returns the name of a named graphic object   |
| Return Type          | String   |
| Return Description   | The name of the graphic object or an empty string if the graphic object is unnamed |

| Attributes | Type       | Description  |
|------------|------------|--|
| Object     | Identifier | Identifier bound to a string representing a graphic object |

### INTERSECTSPPOINT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | INTERSECTSPPOINT  |
| Function Description | Determines if the object intersects the designated point                    |
| Return Type          | Boolean   |
| Return Description   | Returns true if the object intersects the designated point, false otherwise |

| Attributes | Type       | Description  |
|------------|------------|--|
| Object     | Identifier | Identifier bound to a string representing a graphic object   |
| X          | Integer    | x coordinate of the point to check the intersection against  |
| Y          | Integer    | y coordinate of the point to check the intersection against  |
| Tolerance  | Float      | Allowable distance from the point still considered intersecting in the same units used in the coordinate plane of the item |

### HASVERTEX

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | HASVERTEX  |
| Function Description | Determines if a point is among the vertices of an object                             |
| Return Type          | Boolean  |
| Return Description   | Returns true if the point (x,y) is among the vertices of the object, false otherwise |

| Attributes | Type | Description |
|------------|------|-------------|
|------------|------|-------------|



| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier bound to a string representing a graphic object   |
| tolerance  | float      | allowable distance from the point still considered intersecting in the same units used in the coordinate plane of the item |

### TABLE Item Custom Operators

This section describes the custom operators used in response processing of SmarterApp Table Interaction (TI) item type rubrics. All custom operators in this section are of type = TABLE.

#### GETCOLUMN

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETCOLUMN   |
| Function Description | Obtains the requested column  |
| Return Type          | String  |
| Return Description   | A string representing the requested column. The string may be empty |

| Attributes | Type       | Description                                       |
|------------|------------|---|
| table      | Identifier | identifier bound to a string representing a table |
| columnName | String     | the name of the desired column                    |

#### GETHEADERROW

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETHEADERROW  |
| Function Description | Obtains the header row of the table                                     |
| Return Type          | String  |
| Return Description   | A string representing the requested header row. The string may be empty |

| Attributes | Type       | Description                                       |
|------------|------------|---|
| table      | identifier | identifier bound to a string representing a table |

#### GETVALUENUMERIC

| Custom Operator      | Description                                     |
|----------------------|---|
| Function Name        | GETVALUENUMERIC                                 |
| Function Description | Obtains the numeric value of the indicated cell |

| Custom Operator    | Description   |
|--------------------|---|
| Return Type        | Float   |
| Return Description | The numeric value of the indicated cell or NaN if the value cannot be successfully parsed |

| Attributes  | Type       | Description                                       |
|-------------|------------|---|
| tableVector | Identifier | identifier bound to a string representing a table |
| index       | Integer    | the index of the desired cell                     |

### EQUATION Item Custom Operators

This section describes the custom operators used in response processing of SmarterApp Equation (EQ) item type rubrics. All custom operators in this section are of type = EQ.

#### PREPROCESSRESPONSE

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | PREPROCESSRESPONSE   |
| Function Description | Translates equation responses to a collection of strings representing the objects in the set |
| Return Type          | Basetype   |
| Return Description   | A basetype “string” with cardinality “ordered”   |

| Attributes | Type       | Description  |
|------------|------------|--|
| Response   | identifier | names the original XML response to be preprocessed |

#### ISEQUIVALENT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | ISEQUIVALENT  |
| Function Description | Compares the equation expression to an exemplar and determines if they are equivalent |
| Return Type          | Boolean   |
| Return Description   | Returns true if the equation referenced by object is equivalent to the exemplar       |

| Attributes | Type       | Description  |
|------------|------------|--|
| Object     | Identifier | identifier referencing a string representing a mathematical expression |
| exemplar   | String     | string representing a mathematical expression of an exemplar           |
| simplify   | Boolean    | Flag indicating that simplifications can be performed (e.g. $1+1=2$ )  |

#### ISEQUIVALENTLOG

| Custom Operator | Description     |
|-----------------|-----------------|
| Function Name   | ISEQUIVALENTLOG |

| Custom Operator      | Description  |
|----------------------|--|
| Function Description | Compares the equation expression to an exemplar and determines if they are equivalent using properties of Logs |
| Return Type          | Boolean  |
| Return Description   | Returns true if math expression object is equivalent to exemplar expression using properties of Logs           |

| Attributes  | Type       | Description  |
|-------------|------------|--|
| Object      | identifier | identifier referencing a string representing a mathematical expression |
| exemplar    | string     | string representing a mathematical expression of an exemplar           |
| assumptions | Boolean    | Flag indicating that bases are positive and exponents are real         |

### ISEQUIVALENTTRIG

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | ISEQUIVALENTTRIG  |
| Function Description | Compares the equation expression to an exemplar and determines if they are equivalent using Trig identities |
| Return Type          | Boolean   |
| Return Description   | Returns true if math expression object is equivalent to exemplar expression using Trig identities           |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | Identifier | identifier referencing a string representing a mathematical expression |
| exemplar   | String     | string representing a mathematical expression of an exemplar           |

### ISEMPTY

| Custom Operator      | Description                                     |
|----------------------|---|
| Function Name        | ISEMPTY   |
| Function Description | Determines if a math expression is empty        |
| Return Type          | Boolean   |
| Return Description   | Returns true if math expression object is empty |

| Attributes | Type | Description |
|------------|------|-------------|
|------------|------|-------------|



| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier referencing a string representing a mathematical expression |

### NUMBERFROMEXPRESSION

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | NUMBERFROMEXPRESSION  |
| Function Description | Converts the math expression object to a number   |
| Return Type          | double  |
| Return Description   | Returns the numerical value represented by the math expression or Double.NaN if the expression cannot be parsed |

| Attributes | Type       | Description   |
|------------|------------|---|
| object     | identifier | identifier referencing a string representing a mathematical expression whose value is of interest |

### GETINEQUALITIESCOUNT

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | GETINEQUALITIESCOUNT   |
| Function Description | Determines the number of inequalities in a math expression     |
| Return Type          | Integer  |
| Return Description   | Returns the number of inequalities in a math expression object |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier referencing a string representing a mathematical expression |

### ISMATCH

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | ISMATCH  |
| Function Description | Determines if an expression can be matched to a parameterized pattern        |
| Return Type          | Boolean  |
| Return Description   | Returns true if expression can be matched by the given parameterized pattern |

| Attributes  | Type       | Description  |
|-------------|------------|--|
| object      | identifier | identifier referencing a string representing an expression to be matched to a given pattern                      |
| pattern     | string     | A string representing a mathematical expression  |
| parameters  | string     | Comma-separated parameters in the pattern (e.g. a,b,c)   |
| constraints | string     | Comma-separated constraints on parameter values (e.g. $a \geq 1$ and $a < 4$ , $b = 1$ or $b = 2$ , $c \neq 0$ ) |
| variables   | string     | Comma-separated variables in the pattern (e.g. x,t)  |
| simplify    | Boolean    | Flag indicating that simplifications can be performed (e.g. $1+1=2$ )  |

## LINECONTAINS

| Custom Operator      | Description  |
|----------------------|--|
| Function Name        | LINECONTAINS   |
| Function Description | Returns true if line contains an expression equivalent to exemplar expression. For ex, <code>LineContains(8+8+8=24-2=2, 24-2=22)</code> would return true. |
| Return Type          | Boolean  |
| Return Description   | True/False   |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier referencing a string representing a mathematical expression |
| Exemplar   | String     | string representing a mathematical expression of an exemplar           |
| Simplify   | Boolean    | Flag indicating that simplifications can be performed (e.g. $1+1=2$ )  |

## EXPRESSIONCONTAINS

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | EXPRESSIONCONTAINS  |
| Function Description | Determines if a given math expression object contains a given substring |
| Return Type          | Boolean   |
| Return Description   | Returns true if the substring is found in the given math expression     |

| Attributes | Type | Description |
|------------|------|-------------|
|------------|------|-------------|

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | Identifier referencing a string representing a mathematical expression |
| string     | String     | Substring that we are looking for                                      |

## EVALUATE

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | EVALUATE  |
| Function Description | Converts expression to floating-point approximation (decimal number)                  |
| Return Type          | Float   |
| Return Description   | A decimal number that represents the expression or Double.NaN if it cannot be parsed. |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier referencing a string representing a mathematical expression |

## GETEQUATIONSCOUNT

| Custom Operator      | Description   |
|----------------------|---|
| Function Name        | GETEQUATIONSCOUNT   |
| Function Description | Determines the number of equations in a math expression object        |
| Return Type          | Integer   |
| Return Description   | An integer count of the equations in the input math expression object |

| Attributes | Type       | Description  |
|------------|------------|--|
| object     | identifier | identifier referencing a string representing a mathematical expression |

### Example Grid Rubric (Type = GRAPHIC)

This section provides an example of a SmarterApp Grid item rubric that contains CTRL and GRAPHIC custom operators.

```
<?xml version="1.0" encoding="utf-8"?>
<AssessmentItem xmlns="http://www.imsglobal.org/xsd/imsqti_v2p1">
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="string" cardinality="ordered" identifier="PP_RESPONSE" />
  <outcomeDeclaration identifier="a2" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a2FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e2" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="b3" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="b3FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="f3" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="c2" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="c2FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="g2" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="d5" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="d5FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="h5" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="cbool" baseType="integer" cardinality="single" />
  <outcomeDeclaration identifier="a0" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a0FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e0" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a1" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a1FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e1" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a3" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a3FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e3" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a4" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a4FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e4" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a5" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a5FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e5" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a6" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a6FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e6" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a7" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a7FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e7" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a8" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a8FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e8" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="a9" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="a9FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="e9" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="b0" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="b0FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="f0" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="b1" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="b1FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="f1" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="b2" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="b2FirstObject" baseType="string" cardinality="single" />
  <outcomeDeclaration identifier="f2" baseType="boolean" cardinality="single" />
  <outcomeDeclaration identifier="b4" baseType="string" cardinality="ordered" />
</AssessmentItem>
```



```

<outcomeDeclaration identifier="h8" baseType="boolean" cardinality="single" />
<outcomeDeclaration identifier="d9" baseType="string" cardinality="ordered" />
<outcomeDeclaration identifier="d9FirstObject" baseType="string" cardinality="single" />
<outcomeDeclaration identifier="h9" baseType="boolean" cardinality="single" />
<outcomeDeclaration identifier="wbool" baseType="integer" cardinality="single" />
<responseProcessing>
  <setOutcomeValue identifier="PP_RESPONSE">
    <customOperator type="GRAPHIC" functionName="PREPROCESSRESPONSE" response="RESPONSE" />
  </setOutcomeValue>
  <setOutcomeValue identifier="a2">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">a2</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="a2FirstObject">
    <index n="1">
      <variable identifier="a2" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="e2">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a2FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="b3">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">b3</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="b3FirstObject">
    <index n="1">
      <variable identifier="b3" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="f3">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b3FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="c2">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">c2</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="c2FirstObject">
    <index n="1">
      <variable identifier="c2" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="g2">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c2FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="d5">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">d5</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="d5FirstObject">
    <index n="1">

```

```

    <variable identifier="d5" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h5">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d5FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="cbool">
  <customOperator type="CTRL" functionName="COUNTBOOL" list="$e2,$f3,$g2,$h5"
booleanValue="True" />
</setOutcomeValue>
<setOutcomeValue identifier="a0">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a0</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a0FirstObject">
  <index n="1">
    <variable identifier="a0" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e0">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a0FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a1">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a1</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a1FirstObject">
  <index n="1">
    <variable identifier="a1" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e1">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a1FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a3">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a3</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a3FirstObject">
  <index n="1">
    <variable identifier="a3" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e3">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a3FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a4">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a4</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a4FirstObject">

```



```

<index n="1">
  <variable identifier="a4" />
</index>
</setOutcomeValue>
<setOutcomeValue identifier="e4">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a4FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a5">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a5</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a5FirstObject">
  <index n="1">
    <variable identifier="a5" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e5">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a5FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a6">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a6</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a6FirstObject">
  <index n="1">
    <variable identifier="a6" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e6">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a6FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a7">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a7</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a7FirstObject">
  <index n="1">
    <variable identifier="a7" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e7">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a7FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a8">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a8</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a8FirstObject">
  <index n="1">
    <variable identifier="a8" />
  </index>

```



```

</setOutcomeValue>
<setOutcomeValue identifier="e8">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a8FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="a9">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">a9</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="a9FirstObject">
  <index n="1">
    <variable identifier="a9" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="e9">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="a9FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b0">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b0</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b0FirstObject">
  <index n="1">
    <variable identifier="b0" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f0">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b0FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b1">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b1</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b1FirstObject">
  <index n="1">
    <variable identifier="b1" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f1">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b1FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b2">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b2</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b2FirstObject">
  <index n="1">
    <variable identifier="b2" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f2">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b2FirstObject" />

```

```

</setOutcomeValue>
<setOutcomeValue identifier="b4">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b4</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b4FirstObject">
  <index n="1">
    <variable identifier="b4" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f4">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b4FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b5">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b5</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b5FirstObject">
  <index n="1">
    <variable identifier="b5" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f5">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b5FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b6">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b6</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b6FirstObject">
  <index n="1">
    <variable identifier="b6" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f6">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b6FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b7">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b7</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b7FirstObject">
  <index n="1">
    <variable identifier="b7" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f7">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b7FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b8">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">

```

```

    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b8</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b8FirstObject">
  <index n="1">
    <variable identifier="b8" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f8">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b8FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="b9">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">b9</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="b9FirstObject">
  <index n="1">
    <variable identifier="b9" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="f9">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="b9FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c0">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c0</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c0FirstObject">
  <index n="1">
    <variable identifier="c0" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g0">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c0FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c1">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c1</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c1FirstObject">
  <index n="1">
    <variable identifier="c1" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g1">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c1FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c3">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c3</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>

```

```

    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c3FirstObject">
  <index n="1">
    <variable identifier="c3" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g3">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c3FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c4">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c4</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c4FirstObject">
  <index n="1">
    <variable identifier="c4" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g4">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c4FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c5">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c5</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c5FirstObject">
  <index n="1">
    <variable identifier="c5" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g5">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c5FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c6">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c6</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c6FirstObject">
  <index n="1">
    <variable identifier="c6" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g6">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c6FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c7">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c7</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@ " />
    </stringMatch>
  </customOperator>
</setOutcomeValue>

```

```

<setOutcomeValue identifier="c7FirstObject">
  <index n="1">
    <variable identifier="c7" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g7">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c7FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c8">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c8</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c8FirstObject">
  <index n="1">
    <variable identifier="c8" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g8">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c8FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="c9">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">c9</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="c9FirstObject">
  <index n="1">
    <variable identifier="c9" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="g9">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="c9FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d0">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d0</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d0FirstObject">
  <index n="1">
    <variable identifier="d0" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h0">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d0FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d1">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d1</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d1FirstObject">
  <index n="1">
    <variable identifier="d1" />
  </index>

```

```

</index>
</setOutcomeValue>
<setOutcomeValue identifier="h1">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d1FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d2">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d2</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d2FirstObject">
  <index n="1">
    <variable identifier="d2" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h2">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d2FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d3">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d3</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d3FirstObject">
  <index n="1">
    <variable identifier="d3" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h3">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d3FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d4">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d4</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d4FirstObject">
  <index n="1">
    <variable identifier="d4" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h4">
  <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d4FirstObject" />
</setOutcomeValue>
<setOutcomeValue identifier="d6">
  <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
    <stringMatch caseSensitive="True">
      <baseValue baseType="string">d6</baseValue>
      <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
    </stringMatch>
  </customOperator>
</setOutcomeValue>
<setOutcomeValue identifier="d6FirstObject">
  <index n="1">
    <variable identifier="d6" />
  </index>
</setOutcomeValue>
<setOutcomeValue identifier="h6">

```

```

    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d6FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="d7">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">d7</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="d7FirstObject">
    <index n="1">
      <variable identifier="d7" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="h7">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d7FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="d8">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">d8</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="d8FirstObject">
    <index n="1">
      <variable identifier="d8" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="h8">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d8FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="d9">
    <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
      <stringMatch caseSensitive="True">
        <baseValue baseType="string">d9</baseValue>
        <customOperator type="GRAPHIC" functionName="GETNAME" object="@" />
      </stringMatch>
    </customOperator>
  </setOutcomeValue>
  <setOutcomeValue identifier="d9FirstObject">
    <index n="1">
      <variable identifier="d9" />
    </index>
  </setOutcomeValue>
  <setOutcomeValue identifier="h9">
    <customOperator type="GRAPHIC" functionName="ISREGIONSELECTED" region="d9FirstObject" />
  </setOutcomeValue>
  <setOutcomeValue identifier="wbool">
    <customOperator type="CTRL" functionName="COUNTBOOL"
list="$e0,$e1,$e3,$e4,$e5,$e6,$e7,$e8,$e9,$f0,$f1,$f2,$f4,$f5,$f6,$f7,$f8,$f9,$g0,$g1,$g3,$g4,$g5
,$g6,$g7,$g8,$g9,$h0,$h1,$h2,$h3,$h4,$h6,$h7,$h8,$h9" booleanValue="True" />
  </setOutcomeValue>
  <responseCondition>
    <responseIf>
      <and>
        <equal>
          <variable identifier="cbool" />
          <baseValue baseType="integer">4</baseValue>
        </equal>
        <equal>
          <variable identifier="wbool" />
          <baseValue baseType="integer">0</baseValue>
        </equal>
      </and>
    </responseIf>
  </responseCondition>

```



```
<setOutcomeValue identifier="SCORE">  
  <baseValue baseType="integer">1</baseValue>  
</setOutcomeValue>  
</responseIf>  
</responseCondition>  
</responseProcessing>  
</AssessmentItem>
```



### Example Equation Rubric (Type = EQ)

This section provides an example of a SmarterApp Grid item rubric that contains CTRL and EQ custom operators.

```
<AssessmentItem xmlns="http://www.imsglobal.org/xsd/imsqti_v2p1">
  <outcomeDeclaration baseType="integer" cardinality="single" identifier="SCORE">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <responseDeclaration baseType="string" cardinality="single" identifier="RESPONSE" />
  <outcomeDeclaration baseType="string" cardinality="ordered" identifier="PP_RESPONSE" />
  <outcomeDeclaration identifier="obj1" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="obj1Count" baseType="integer" cardinality="single" />
  <outcomeDeclaration identifier="obj" baseType="string" cardinality="ordered" />
  <outcomeDeclaration identifier="objCount" baseType="integer" cardinality="single" />
  <responseProcessing>
    <setOutcomeValue identifier="PP_RESPONSE">
      <customOperator type="EQ" functionName="PREPROCESSRESPONSE" response="RESPONSE" />
    </setOutcomeValue>
    <setOutcomeValue identifier="obj1">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <customOperator type="EQ" functionName="ISEQUIVALENT" object="@" exemplar="4960/32"
simplify="False" />
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="obj1Count">
      <containerSize>
        <variable identifier="obj1" />
      </containerSize>
    </setOutcomeValue>
    <setOutcomeValue identifier="obj">
      <customOperator type="CTRL" functionName="mapExpression" container="PP_RESPONSE">
        <customOperator type="EQ" functionName="ISEQUIVALENT" object="@" exemplar="155"
simplify="True" />
      </customOperator>
    </setOutcomeValue>
    <setOutcomeValue identifier="objCount">
      <containerSize>
        <variable identifier="obj" />
      </containerSize>
    </setOutcomeValue>
    <responseCondition>
      <responseIf>
        <and>
          <equal>
            <variable identifier="obj1Count" />
            <baseValue baseType="integer">0</baseValue>
          </equal>
          <equal>
            <variable identifier="objCount" />
            <baseValue baseType="integer">1</baseValue>
          </equal>
        </and>
        <setOutcomeValue identifier="SCORE">
          <baseValue baseType="integer">1</baseValue>
        </setOutcomeValue>
      </responseIf>
    </responseCondition>
  </responseProcessing>
</AssessmentItem>
```