Built-in operations

name	Description	Example
	Date	1
limestamp	The operation returns current system time in milliseconds as a string type value. It takes no parameters.	Result = "1392798008000"
FormatDate	The operation converts a date and time given in milliseconds to a human readable format.	Date = "1392798008000"
	 The operation takes two parameters: Date – a date and time in milliseconds that should be converted to a human readable format. It must be a string type value and can be the result of a Timestamp operation. Format – a date and time format for the conversion. It must be a string type value. For the date and time formats, refer to the SimpleDateFormat page. 	Format = "yyyy.MM.dd G 'at' HH:mm:ss z" Result = "2014.02.19 AD at 10:20:08 EET"
	FormatDate1 Value: Wyyy.MM.dd G 'at' HH:mm:ss	
	The result of this operation is a string type value.	
ParseDate	The operation converts a date and time in a human readable format to milliseconds. In other words, the operation reverses the result of the FormatDate operation (it returns the value that can be the Date parameter for a FormatDate operation). The operation takes two parameters:	Format = "yyyy.MM.dd G 'at' HH:mm:ss z" Formatted Date = "2014.02.19 AD at 10:20:08 EET"
	 Format – a current format of the date and time that should be converted to milliseconds. It must be a string type value. Formatted Date – a date and time that should be converted to milliseconds. It must be a string type value. 	Result = "1392798008000"
	ParseDate1 Walue: 2014.02.19 AD at 10:20:08 EET	
	The result of this operation is a string type value.	
	Collection	
TypeTest	 The operation tests, whether the type of an element matches a given type or stereotype. If the types matches, it returns <i>true</i>, and if they not – <i>false</i>. You can also use this operation to check, if an element is an instance of a given classifier. The operation takes three parameters: Element – a model element, whose type you need to test. 	Element = class Profile Type = Class Include Subtypes = false Result = true The element type matches the given type.
	 Type – a type, stereotype, or classifier for testing the element. Include Subtypes – true, if the inherited types, stereotypes, or classifiers of the selected Type parameter value should be included in the test; <i>false</i>, if not. 	Element = class Profile Type = Classifier Include Subtypes = true
	Image: Second state of the second s	Result = true Though the element type doe not match the given type, it is its subtype, and in this case
	The TypeTest operation can be used in the Filter operation as the condition. For this, click the Use as	subtypes are included in the test.

	Type Test () Use as Condition of a new Filter operation Remove Operation Name: ClassifierTypeTest	Element = class Profile Type = Classifier Include Subtypes = false Result = false Thought the element type is a
		subtype of the given type, in this case subtypes are not included in the test.
Filter	The operation analyses given collection of elements (one by one) and returns only those elements that meet the specified filter criteria. The operation takes two parameters: • Input Collection – an arbitrary collection of elements of arbitrary types. • Predicate – an operation determining filter criteria. Keep in mind that <i>this operation must have</i> exactly one input parameter and return a result of boolean type! Otherwise, the Filter operation fails and returns no value. Hence, if you need to use the script operation with many parameters as the filter criteria, you must wrap it inside the nested operation (the TypeTest operation or any other built-in operation is wrapped automatically). Filter 1 Filter 1 Filter 1 Filter 1 Filter 2 Filter 1 Filter 1 Filter 1 Filter 1 Filter 2 Filter 1 Filter 2 Filter 1 Filter 2 Filter 2 Filter 1 Filter 2 Filter 1 Filter 3 Filter 2 Filter 1 Filter 4 Filter 4 Filter 5 Filter 5 Filter 5 Filter 5 Filter 5 Filter 5 Filter 5 Filter 6 Filter 6 Filter 6 Filter 7 Filter 7 Filte	Input Collection = use case Log in use case Log out actor User use case Change password Predicate = TypeTest: Element = Input Collection Type = UseCase Include Subtypes = false Result Collection = use case Log in use case Log out use case Change password
InstanceNav igation	The operation finds the needed slot of the given instance and returns one or more values of that slot. The operation takes two parameters: Instance – an instance specification (instance), whose slot value you need to access. Navigate Property – a property, whose slot you need to find. Instance=WN345AB : Boat Navigate Property = -Year: String The result type of this operation matches the type of the property specified as the Navigate Property parameter.	Instance = WN345AB Navigate Property = Year Result = "1999" Boat - Boat - Make : String - Make : String - WN345AB : Boat - String - Make = "Hanter" - S Year = "1999"
First	 The operation finds the first element in a sequence from a specified scope. The operation consist of: Input - objects list. 	
Contains	 The operation verifies specified object in given collection and returns Boolean value <i>true</i> if sets contains object, otherwise operation returns value <i>false</i>. The operation consist of: Input - operation will verify this collections. Obj - sets will be verified according to this object. 	Input =

Size	The operation determine whether collection is empty, if it is empty, operation returns Boolean value <i>true</i> , otherwise operation returns value <i>false</i> . The operation consist of: • Input - specified object (for example array, set, list). The operation returns the number of elements in collection. The operation consist of: • Input - an object or collection (for example array, set, list, string).	
Size	Input - specified object (for example array, set, list). The operation returns the number of elements in collection. The operation consist of:	
Intersect	The operation returns the number of elements in collection. The operation consist of:	
Intersect	The operation consist of:	
Intersect		
	• Input - an object or collection (for example array, set, list, string).	
·	The operation returns the data common to both collections, with no repetitions and in sorted order.	C1 = [7 0 5], C2 = [7 1 5],
	The operation consist of:	returns C = [5 7];
	Collection1 and Collection2 - collections which will be intersected.	
	The operation joins two lists. The List1 and List2 are joined end-to-end. Keeps the order, allows duplicates.	
	The operation consist of:	
	List1 and List2 - lists which will be concatenated.	
Get	The operation returns the element at the specified position in collection.	
	The operation consist of:	
	 Input - specified collection. Index - index of the element to return (integer type). 	
	The operation maps objects from collection to collection of other objects and returns a collection of results in the same order.	Input = [1 2 3 4 5],
	The operation consist of:	applies operation map with function (Input + 1)
	 Input - collections with objects to map. MapOperation - given function, this function will be applied to each input element. 	returns = [2 3 4 5 6]
	The operation maps objects from collection to collection of other objects. Flattens mapping result if it is a collection.	Input = { {1,2}, {3,4}, {5,6} }
	The operation consist of:	applies operation MapFlat
	 Input - collections with objects to map. MapOperation - given function, this function will be applied to each input element. 	returns = {1,2,3,4,5,6}
Reduce	The operation reduce collection using given operation.	
	The operation consist of:	
	 Input - collection that will be reduced. ReduceOperation - given function, this function determines how collection will be reduced. 	
Zip	The operation zips two collections using given zip operation.	
	The operation consist of:	
	 Collection1 and Collection2 - to these collections, zip operation will be applied. ZipOperation - given function, according to this function, collections will be zipped. 	
Min	The operation returns minimum value from given collection.	
	The operation consist of:	
	• Input - collection which will be tested and minimum value determined.	

Max	The operation returns maximum value from given collection.	
	The operation consist of:	
	• Input - collection which will be tested and maximum value determined.	
AllMatch	The operation checks if all collection elements match given predicate and returns boolean value.	
	The operation consist of:	
	 Input - specified collection. Predicate - a predicate is applied to each element of collection. 	
AnyMatch	The operation checks if all collection elements match given predicate and returns boolean value.	
	The operation consist of:	
	 Input - specified collection. Predicate - a predicate is applied to each element of collection. 	
	Logical	
And	The logical conjunction operation returns Boolean value <i>true</i> if all parameters are <i>true</i> , in all other cases - operation returns <i>false</i> value.	
	The operation by default takes three parameters:	
	 A and B - two logical values. Result - a Boolean value which shows results of logical conjunction 	
Or	The logical disjunction operation returns Boolean value <i>true</i> if either or both parameters is <i>true</i> , otherwise operation returns <i>false</i> value.	
	The operation takes these parameters:	
	• A and B - two logical values.	
Xor	The logical exclusive disjunction operation returns Boolean value <i>true</i> if both parameters differ, otherwise operation returns <i>false</i> value.	
	The operation takes these parameters:	
	• A and B - two logical values.	
Not	The logical negation operation returns Boolean value <i>true</i> if parameter have value <i>false</i> , and <i>false</i> whe n parameter have value <i>true</i> .	
	The operation takes one parameter:	
	• A - logical value.	
	Comparison	
lfThenElse	The operation returns one or other object depending on condition.	
	The operation consist of:	
	 Condition - defines the <i>operation</i> that determines which value to assign. Then - <i>expression</i> defines the value to assign if <i>condition</i> is <i>true</i>. Else - <i>expression</i> defines the value to assign if <i>condition</i> is <i>false</i>. 	
LessThan	The operation returns <i>true</i> if the left parameter is less than the right parameter (A < B).	
	The operation consist of:	

Less ThanO The operation returns true if the left side parameter is less than or equal to the right parameter (A <= 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,			
Image: A and B - parameters which will be compared. Image: A and B - parameters which will be compared GreaterThan The operation returns true if the left parameter is greater than the right parameter (A > B). The operation consist of: Image: A and B - parameters which will be compared GreaterThan The operation returns true if the left parameter is greater than or equal to the right parameter (A > B). The operation consist of: Image: A and B - parameters which will be compared Equals The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters which will be compared Equals The operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters which will be compared StringComet The operation consist of: • A and B - parameters which will be compared A = Helio B = world StringComet The operation returns Boolean value true, if specified string values are joined end-to-end. • A and B - two string values. A = 0 B = world StringComet The operation returns Boolean value true, if specified string values is find in specified scope, otherwise. • A - according to this string, operation will verify B A = 0 B = world StringComet The operation returns Boolean value true if project uses specified diagram, otherwise returns value. • B = world <th></th> <th></th> <th></th>			
GreaterThan The operation returns true if the left parameter is greater than the right parameter (A > B). The operation consist of: • A and B - parameters which will be compared Image: Compare the compared of the compared of the compared of the compared of the compares them. If parameter (A >= B). The operation consist of: • A and B - parameters which will be compared Image: Compare the compares them. If parameter (A >= B). The operation consist of: • A and B - parameters which will be compared Image: Compare the compares them. If parameter (A >= B). The operation consist of: • A and B - parameters which will be compared Equals The operation consist of: • A and B - parameters which will be compared Image: Compare them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: • A and B - parameters which will be compared Image: Compare them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: • A and B - parameters which will be compared Image: Compare them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: • A and B - parameters which will be compared Image: Compare them. If parameters are equal, operation returns true if they are not the same type, then compares them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: • A and B - parameters which will be compared Image: Compare them. If parameters are equal, operation returns false. StringConcat The operation returns Boolean value		The operation consist of:	
The operation consist of: A and B - parameters which will be compared GreaterThan The operation returns true if the left parameter is greater than or equal to the right parameter (A >= B). The operation consist of: A and B - parameters which will be compared Equals The operation consist of: The operation consist of: A and B - parameters if they are not the same type, then compares them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: A and B - parameters which will be compared NotEquals The operation consist of: A and B - parameters which will be compared A and B - parameters which will be compared NotEquals The operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation returns Boolean value true, otherwise operation returns false. The operation consist of: A and B - parameters which will be compared StringConcet The operation consist of: B = worldl A and B - parameters which will be compared A = Hello B = worldl Return = Hello world! A = A and B - parameters with string values. A and B string values are joined end-to-end. A = 0 B = world! Return = Hello world! Return = True Inservertion consist of:		• A and B - parameters which will be compared.	
Image: A and B - parameters which will be compared Image: A and B - parameters which will be compared Greater The operation neturns true if the left parameter is greater than or equal to the right parameter (A >= B). Image: A and B - parameters which will be compared Equals The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters if they are not the same type, then compares them. If parameters The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters if they are not the same type, then compares them. If parameters The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters if they are not the same type, then compares them. If parameters The operation consist of: • A and B - parameters which will be compared Image: A and B - parameters if they are not the same type, then compares them. If parameters are not equal, operation returns Boolean value true, otherwise operation returns faise. The operation consist of: • A and B - parameters with string values. A and B string values are joined end-to-end. A = no end image: A and B - two string values. A = Hello B = world! Return = Hello world! Return = Hello world! Return = True StringContal Insequent to returns Boolean value true, if specified string value is find in specified scope, otherwise, b = string value. A = 0 B = world! Return = True Diagram Type = true degram to this string, operation will verify B b = string value. Diagram = Use Case Diagr	GreaterThan	The operation returns <i>true</i> if the left parameter is greater than the right parameter $(A > B)$.	
Greaterinam The operation returns true if the left parameter is greater than or equal to the right parameter (A >= B). The operation consist of: • A and B - parameters which will be compared Equals The operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared NotEquals The operation consist of: • A and B - parameters which will be compared A = Hello Bar and the sparameters which will be compared A = Hello StringConcat The operation consist of: A and B - parameters which will be compared StringConcat The operation consist of: A and B - two string values. A and B string values are joined end-to-end. A = Hello StringConcat The operation consist of: B - world! B - world! • A and B - two string val		The operation consist of:	
Or Equais The operation consist of: A and B - parameters which will be compared Equais The operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. Ferritor operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation converts parameters which will be compared NotEquais The operation converts parameters which will be compared A and B - parameters which will be compared A and B - parameters which will be compared A and B - parameters which will be compared A = Hello B = world! Return = Hello world! A = no eparation consist of: A and B - two string values. A = o B = world! Return = Hello world! A = o concling to this string, operation will verify B B = world! Return = True B = world! A - a cocording to this string, operation will verify B B - string value. Diagram - all diagrams in the specified scope, "Type - write diagram name exacity how it is named in modeling tool. Diag		• A and B - parameters which will be compared	
The operation consist of: • A and B - parameters which will be compared Image: Comparison of the operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation returns Boolean value true, otherwise operation returns false. Image: Comparison of the operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation returns Boolean value true, otherwise operation returns false. Image: Comparison terums Boolean value true, otherwise operation returns false. NotEquals The operation converts parameters if they are not the same type, then compares them. If parameters are not equal, operation returns Boolean value true, otherwise operation returns false. Image: Comparison terums Boolean value true, otherwise operation returns false. NotEquals The operation consist of: • A and B - parameters which will be compared A = Hello StringConcat The operation consist of: • A and B - two string values. A = Hello StringContating The operation returns Boolean value true, if specified string value is find in specified scope, otherwise are interested. A = o StringContating The operation returns Boolean value true if project uses specified diagram, otherwise returns value are interested. A = o Bigram = interparation consist of: • A = according to this string, operation will verify B B = world! Diag		The operation returns <i>true</i> if the left parameter is greater than or equal to the right parameter ($A \ge B$).	
Equals The operation converts parameters if they are not the same type, then compares them. If parameters are equal, operation returns Boolean value <i>true</i> , otherwise operation returns <i>false</i> . NotEquals The operation converts parameters which will be compared Image: Compare the same type, then compares them. If parameters are not equal, operation returns Boolean value <i>true</i> , otherwise operation returns <i>false</i> . Image: Compare the same type, then compares them. If parameters are not equal, operation converts parameters which will be compared NotEquals The operation converts parameters which will be compared String StringConcat The operation converts parameters which string values. A and B string values are joined end-to-end. A = Hello B = world! Return = Hello world! Return = Hello world! Return = Hello world! StringContat The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. A = o B's string value. Diagram is the specified scope. Other Diagram = Use Case Diagram Free Form Diagram	•	The operation consist of:	
are equal, operation returns Boolean value <i>true</i> , otherwise operation returns <i>talse</i> . Image: Comparison consist of: Image: Comparison cons		• A and B - parameters which will be compared	
Image: A and B - parameters which will be compared Image: A and B - parameters are not equal, operation returns Boolean value true, otherwise operation returns false. Image: A and B - parameters are not equal, operation returns Boolean value true, otherwise operation returns false. Image: A and B - parameters are not equal, operation returns Boolean value true, otherwise operation returns false. Image: A and B - parameters which will be compared StringConcat The operation consist of: A and B - parameters which will be compared A = Hello StringConcat The operation joins two parameters with string values. A and B string values are joined end-to-end. A = Hello B = world! Return = Hello world! Return = Hello world! StringContat The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. A = o B = world! Return = True B = world! Return = True DiagramTyp The operation returns Boolean value true if project uses specified diagram, otherwise returns value Diagram = Use Case Diagram	Equals		
NotEquals Image: Constraint of the pertation returns Boolean value true, otherwise operation returns false. Image: Constraint of the pertation returns false. The operation consist of: A and B - parameters which will be compared String StringConcation The operation consist of: B = world! The operation consist of: B = world! B = world! The operation consist of: B = world! B = world! The operation consist of: B = world! B = world! Note operation returns Boolean value true, if specified string values is find in specified scope, otherwise. A = 0 StringContation The operation consist of: B = world! The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. A = 0 B = world! A - a correding to this string, operation will verify B B = world! DiagramType of Tables. The operation consist of: Diagram = Use Case Diagram Class Diagram Free Form Diagram Pree For		The operation consist of:	
are not equal, operation returns Boolean value true, otherwise operation returns false. Interpretation consist of: Interpretation consist		• A and B - parameters which will be compared	
• A and B - parameters which will be compared String StringConcat The operation joins two parameters with string values. A and B string values are joined end-to-end. A = Hello B = world! B = world! • A and B - two string values. A = 0 B = two paration returns Boolean value true, if specified string value is find in specified scope, otherwise. A = 0 StringContatins The operation consist of: • A - according to this string, operation will verify B • A - according to this string, operation will verify B B = world! Return = True DiagramType The operation consist of: • Diagram = Use Case Diagram Free Form	NotEquals		
StringConcat The operation joins two parameters with string values. A and B string values are joined end-to-end. A = Hello The operation consist of: • A and B - two string values. B = world! • A and B - two string values. Return = Hello world! StringContatins The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. A = o String Contatins The operation consist of: • A - according to this string, operation will verify B A = o String value. DiagramType The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. Diagram = Use Case Diagram Free Form Diagram Type = Class Diagram Class Diagram Free Form Diagram Type = Class Diagram Free Form Diagram Fr		The operation consist of:	
StringConcat The operation joins two parameters with string values. A and B string values are joined end-to-end. The operation consist of: • A and B - two string values. A = Hello B = world! Return = Hello world! StringConta ins The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. • B - string value. A = o B = world! Return = True DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. Diagram = Use Case Diagram Free Form Diagra Free Form Diagram Free Form Diagram Free Form Diagram Free Form Diagram Return = True Diagram = Use Case Diagram Free Form Diagram 		• A and B - parameters which will be compared	
The operation consist of: • A and B - two string values. B = world! StringConta ins The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. A = o The operation consist of: • A - according to this string, operation will verify B B = world! • B - string value. Return = True B = world! DiagramTyp eTest The operation consist of: • Diagram - all diagrams in the specified scope. • Type - write diagram name exactly how it is named in modeling tool. Diagram - True		String	1
A and B - two string values.Return = Hello world!StringConta insThe operation returns Boolean value true, if specified string value is find in specified scope, otherwise. The operation consist of: • B - string value.A = o B = world! Return = TrueDiagramTyp eTestThe operation returns Boolean value true if project uses specified diagram, otherwise returns value • Diagram - all diagrams in the specified scope. • Type - write diagram name exactly how it is named in modeling tool.Diagram - Use Case Diagram Free Form Diagra	StringConcat	The operation joins two parameters with string values. A and B string values are joined end-to-end.	A = Hello
StringConta ins The operation returns Boolean value true, if specified string value is find in specified scope, otherwise. The operation consist of: • A - according to this string, operation will verify B • B - string value. A = o B = world! Return = True DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. The operation consist of: • Diagram - all diagrams in the specified scope. • Type - write diagram name exactly how it is named in modeling tool. Diagram = Use Case Diagram Type = Class Diagram Return = True		The operation consist of:	B = world!
ins The operation consist of: A - according to this string, operation will verify B B - string value. Return = True DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. Diagram = DiagramTyp eTest The operation consist of: Use Case Diagram Class Diagram Free Form Diagram Free Form Diagram Return = True		• A and B - two string values.	Return = Hello world!
The operation consist of: B = world! • A - according to this string, operation will verify B Return = True • B - string value. Other DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. Diagram = Use Case Diagram Class Diagram Free Form Diagram Free Form Diagram Return = True		The operation returns Boolean value true, if specified string value is find in specified scope, otherwise.	A = 0
 B - string value. DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. The operation consist of: Diagram - all diagrams in the specified scope. Type - write diagram name exactly how it is named in modeling tool. 	1115	The operation consist of:	B = world!
DiagramTyp eTest The operation returns Boolean value true if project uses specified diagram, otherwise returns value false. Diagram = The operation consist of: Use Case Diagram Free Form Diagram Free Form Diagram Free Form Diagram Return = True Image: Case Diagram Class Diagram Pree Form Diagram Pree Point Dia			Return = <i>True</i>
eTest false. Use Case Diagram The operation consist of: • Diagram - all diagrams in the specified scope. Use Case Diagram • Type - write diagram name exactly how it is named in modeling tool. Type = Class Diagram Return = True True		Other	1
The operation consist of: Class Diagram • Diagram - all diagrams in the specified scope. Free Form Diagra • Type - write diagram name exactly how it is named in modeling tool. Type = Class Diagram Return = True True	• • • •		, , , , , , , , , , , , , , , , , , ,
 Diagram - all diagrams in the specified scope. Type - write diagram name exactly how it is named in modeling tool. Type = Class Diagram Return = <i>True</i> 		The operation consist of:	Class Diagram
Return = <i>True</i>			
		• Type - write diagram name exactly how it is named in modeling tool.	
grams builds generic tables, dependency matrices, relationship maps, etc. It should be best avoided and used only if you understand the possible consequence of building all diagrams in the project.	UsageInDia grams		
Searches for usages only in all diagrams.		Searches for usages only in all diagrams.	
The operation consists of:		The operation consists of:	
element - the element to search for		element - the element to search for	