Ember+ Function Representation

Proposal for an extension to the Glow Schema used by the Ember+ protocol

Authors: [Kimon.Hoffmann@lawo.de](mailto:Kimon.Hoffmann@lawo.de), [Philip.Boger@l-s-b.de](mailto:Philip.Boger@l-s-b.de)  
Date: 2013-04-10  
Revision: 3

# Introduction

Although the main purpose of Ember+ is describing and mutating the state of interconnected systems by encoding the state itself, there are use cases which require remote procedure calls through the Ember+ interface. In some cases, the old way of triggering actions on the remote host through usage of simple “trigger” parameters is not sufficient because of the lack of trigger arguments.

Therefore this document describes a possible way of integrating function calls into the Glow schema.

Functions shall be described as leafs in the Glow tree (like parameters). The description shall include the number, names and types of function arguments and the type definition of the function’s return value. Both argument list and return value shall be described as n-tuples of primitive types.

Functions shall be invoked through a new Command (“invoke” command): the consumer may send a Function object containing an invoke command, which in turn contains the argument values and an invocation id. The provider shall respond with an object of the new type InvocationResult, which repeats the invocation id issued by the consumer and contains the return value.

The consumer may issue “Fire and forget” function calls by omitting the invocation id. This shall only be allowed for functions which do not yield a result.

# Type Definitions

## ASN.1 Notation

The new type Function stands alongside the types Node and Parameter (defined in Glow Schema 2.10):

|  |
| --- |
| Function ::=  [APPLICATION 19] IMPLICIT  SEQUENCE {  number [0] Integer32,  contents [1] FunctionContents OPTIONAL,  children [2] ElementCollection OPTIONAL  }  QualifiedFunction ::=  [APPLICATION 20] IMPLICIT  SEQUENCE {  path [0] RELATIVE-OID,  contents [1] FunctionContents OPTIONAL,  children [2] ElementCollection OPTIONAL  }  FunctionContents ::=  SET {  identifier [0] EmberString OPTIONAL,  description [1] EmberString OPTIONAL,  arguments [2] TupleDescription OPTIONAL,  result [3] TupleDescription OPTIONAL  }  TupleDescription ::=  SEQUENCE OF [0] TupleItemDescription  TupleItemDescription ::=  [APPLICATION 21] IMPLICIT  SEQUENCE {  type [0] ParameterType,  name [1] EmberString OPTIONAL  }  Invocation ::=  [APPLICATION 22] IMPLICIT  SEQUENCE {  invocationId [0] Integer32 OPTIONAL,  arguments [1] Tuple OPTIONAL  }  Tuple ::=  SEQUENCE OF [0] Value  InvocationResult ::=  [APPLICATION 23] IMPLICIT  SEQUENCE {  invocationId [0] Integer32,  success [1] BOOLEAN OPTIONAL,  result [2] Tuple OPTIONAL  } |

## Function

* number [0] Integer32,

The number of the Function object. Has same semantics as Parameter.number and Node.number.

* contents [1] FunctionContents OPTIONAL

Contents set of the function object. Analogical to Parameter.contents and Node.contents.

* children [2] ElementCollection OPTIONAL

Contains child elements of the function object. Conceptually, function objects are considered leafs in the Glow tree. Though, a matrix may contain a Glow Command in the same way as a parameter or matrix.

## QualifiedFunction

* path [0] RELATIVE-OID

The absolute path to the Function. Uses the same semantics as QualifiedParameter.path and QualifiedNode.path.

* contents [1] FunctionContents OPTIONAL

see Function.contents

* children [2] ElementCollection OPTIONAL

see Function.children

## FunctionContents

* identifier [0] EmberString OPTIONAL

The object identifier. Analogical to Parameter.identifier and Node.identifier.

* description [1] EmberString OPTIONAL

The display name of the matrix object. Analogical to Parameter.description and Node.description.

* arguments [2] TupleDescription OPTIONAL

Describes the arguments the function takes with name and type. If not present, the function does not take any arguments.

* result [3] TupleDescription OPTIONAL

Describes the return value of the function, which is a tuple of arbitrary length and type.

TupleItemDescription

* type [0] P

The type of the argument. Only primitive types are supported.

* name [1] EmberString OPTIONAL

The name of the argument. May be omitted if the function only takes one argument.

Invocation

* invocationId [0] Integer32 OPTIONAL

An integer number generated by the consumer and repeated by the provider in the InvocationResult. Used by the consumer to relate InvocationResults to Invocations.

* arguments [1] Tuple OPTIONAL

The values of the arguments as described in Function.arguments.

InvocationResult

* invocationId [0] Integer32

The invocation id originally issued by the consumer in the invoke command.

* success [1] BOOLEAN OPTIONAL

True or omitted if the provider did not encounter any errors while executing the function.

* result [2] Tuple OPTIONAL

The return value of the function call. The values in this tuple must match the description given in FunctionContents.result.

# Changes to Existing Types

The following changes only add to the existing types to preserve backwards compatibility.

|  |
| --- |
| Command ::=  [APPLICATION 2] IMPLICIT  SEQUENCE {  number [0] CommandType,  options CHOICE {  dirFieldMask [1] FieldFlags,  invocation [2] Invocation  } OPTIONAL  }  CommandType ::=  INTEGER {  subscribe (30),  unsubscribe (31),  getDirectory (32),  invoke (33)  }  Element ::=  CHOICE {  parameter Parameter,  node Node,  command Command,  matrix Matrix,  function Function  }  Root ::=  [APPLICATION 0]  CHOICE {  elements RootElementCollection,  streams StreamCollection,  invocationResult InvocationResult  }  RootElement ::=  CHOICE {  element Element,  qualifiedParameter QualifiedParameter,  qualifiedNode QualifiedNode,  qualifiedMatrix QualifiedMatrix,  qualifiedFunction QualifiedFunction  } |

# Use Cases

The following use cases are illustrated as XML samples (Glow is easily transposed to XML).

## Transmission of the Function Description

*Consumer -> Provider:*

|  |
| --- |
| <QualifiedNode path="1.2">  <children>  <Command number="32" />  </children>  </QualifiedNode> |

This message requests all children under the node with path “1.2”.

*Provider -> Consumer:*

|  |
| --- |
| <QualifiedNode path="1.2">  <children>  <Function number="1">  <contents>  <identifier>setObjectNameRecursive</identifier>  <description>Sample Function</description>  <arguments>  <TupleItemDescription type="INTEGER" name="objectId" />  <TupleItemDescription type="BOOLEAN" name="isRecursive" />  <TupleItemDescription type="UTF8" name="newName" />  </arguments>  <result>  <TupleItemDescription type="integer" name="changeCount" />  </result>  </contents>  </Function>  </children>  </QualifiedNode> |

This message reports one child under the requested node: a Function with three arguments (int objectId, bool isRecursive, string newName) and returning the number of changes committed by calling the function.

## Invocation of Functions

*Consumer -> Provider:*

|  |
| --- |
| <QualifiedFunction path="1.2.1">  <children>  <Command number="33">  <invocation invocationId="1">  <arguments>  <Value type="INTEGER">123</Value>  <Value type="BOOLEAN">True</Value>  <Value type="UTF8">Herbert</Value>  </arguments>  </invocation>  </Command>  </children>  </QualifiedFunction> |

This message issues a recursive rename of all objects beneath object 123.

*Provider -> Consumer:*

|  |
| --- |
| <InvocationResult invocationId="1">  <result>  <Value type="INTEGER">74</Value>  </result>  </InvocationResult> |

This message reports that 74 objects have been changed through invocation with id 1.