Ember+ Matrix Representation

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

Author: [pbo@l-s-b.de](mailto:pbo@l-s-b.de)  
Date: 2012-08-23

Contents

[Introduction 3](#_Toc326662661)

[Examples 3](#_Toc326662662)

[Operators 4](#_Toc326662663)

[Built-in Constants 6](#_Toc326662664)

[Built-in Functions 7](#_Toc326662665)

# Introduction

This document describes a possible way of integrating matrices into the Glow schema. A matrix in the sense of this document is a two-dimensional array of Boolean values.

Possible applications of matrices in the context of gadget control:

* Signal routing
* GP-I/O signaling
* Key assignment for Intercom systems
* Group and conference management for Intercom systems

The entities represented by the rows and columns of a matrix are called signals. The signals represented by the matrix columns are called targets. The signals represented by the matrix rows are called sources.   
The values contained in the matrix cells may be interpreted as connections from sources to targets: If matrix[S, T] is true, source S is connected to target T.

Matrices inherently have three different connection semantics:

* 1:N  
  one source may be connected to n targets, but each target must not be connected to more than one source
* 1:1  
  one source may be connected to only one target, and each target must not be connected to more than one source
* N:N  
  a source may be connected to n targets, and a target may have n sources connected to it.

Ember+ must support all three types of matrix types.

# Type Definitions

The new type Matrix stands alongside the types Node and Parameter (defined in Glow Schema 2.5):

|  |
| --- |
| Matrix ::=  [APPLICATION 13] IMPLICIT  SEQUENCE {  number [0] Integer32,  contents [1] MatrixContents OPTIONAL,  children [2] ElementCollection OPTIONAL,  targets [3] TargetCollection OPTIONAL,  sources [4] SourceCollection OPTIONAL,  connections [5] ConnectionCollection OPTIONAL,  }  MatrixContents ::=  SET {  identifier [0] EmberString,  description [1] EmberString OPTIONAL,  type [2] MatrixType OPTIONAL,  addressingMode [3] MatrixAddressingMode OPTIONAL,  targetCount [4] Integer32,  sourceCount [5] Integer32,  maximumTotalConnects [6] Integer32 OPTIONAL,  maximumConnectsPerTarget [7] Integer32 OPTIONAL,  parametersLocation [8] ParametersLocation OPTIONAL,  labels [9] LabelCollection OPTIONAL  }  MatrixAddressingMode ::=  INTEGER {  linear (0),  nonLinear (1)  }  MatrixType ::=  INTEGER {  oneToN (0),  oneToOne (1),  nToN (2)  }  MatrixParametersLocation ::=  CHOICE {  basePath RELATIVE-OID,  inline Integer32  }  LabelCollection ::=  SEQUENCE OF [0] Label  Label ::=  [APPLICATION 18] IMPLICIT  SEQUENCE {  basePath [0] RELATIVE-OID,  description [1] EmberString  }  TargetCollection ::=  SEQUENCE OF [0] Target  Target ::=  [APPLICATION 14] IMPLICIT  Signal  Signal ::=  SEQUENCE {  number [0] Integer32,  }  SourceCollection ::=  SEQUENCE OF [0] Source  Source ::=  [APPLICATION 15] IMPLICIT  Signal  ConnectionCollection ::=  SEQUENCE OF [0] Connection  Connection ::=  [APPLICATION 16] IMPLICIT  SEQUENCE {  target [0] Integer32,  sources [1] PackedNumbers,  operation [2] ConnectionOperation OPTIONAL,  errorLevel [3] ConnectionErrorLevel OPTIONAL,  }  PackedNumbers ::=  RELATIVE-OID  ConnectionOperation ::=  INTEGER {  absolute (0),  connect (1),  disconnect (2)  }  ConnectionErrorLevel ::=  INTEGER {  tally (0),  taken (1),  pending (2),  locked (3)  -- more tbd.  }  QualifiedMatrix ::=  [APPLICATION 17] IMPLICIT  SEQUENCE {  path [0] RELATIVE-OID,  contents [1] MatrixContents OPTIONAL,  children [2] ElementCollection OPTIONAL,  targets [3] TargetCollection OPTIONAL,  sources [4] SourceCollection OPTIONAL,  connections [5] ConnectionCollection OPTIONAL  } |