Documentation for all MOOSE classes and functions

As visible in Python module

Table of Contents

- 1 MOOSE Classes
  - 1.1 Annotator
  - 1.2 Arith
  - 1.3 Boundary
  - 1.4 BufPool
  - 1.5 CaConc
  - 1.6 ChanBase
  - 1.7 ChemMesh
  - 1.8 Cinfo
  - 1.9 Clock
  - 1.10 Compartment
  - 1.11 CplxEnzBase
  - 1.12 CubeMesh
  - 1.13 CylMesh
  - 1.14 CylPanel
  - 1.15 DiagonalMsg
  - 1.16 DiffAmp
  - 1.17 DiskPanel
  - 1.18 Enz
  - 1.19 EnzBase
  - 1.20 Finfo
  - 1.21 FuncPool
  - 1.22 GHK
  - 1.23 Geometry
  - 1.24 Group
  - 1.25 GslIntegrator
  - 1.26 GssaStoich
  - 1.27 HDF5DataWriter
  - 1.28 HDF5WriterBase
  - 1.29 HHChannel
  - 1.30 HHChannel2D
  - 1.31 HHGate
  - 1.32 HHGate2D
  - 1.33 HSolve
  - 1.34 HemispherePanel
  - 1.35 IntFire
  - 1.36 Interpol2D
  - 1.37 IzhikevichNrn

- 1.38 LeakyIaF
- 1.39 MMenz
- 1.40 MarkovChannel
- 1.41 MarkovGslSolver
- 1.42 MarkovRateTable
- 1.43 MarkovSolver
- 1.44 MarkovSolverBase
- 1.45 MathFunc
- 1.46 Mdouble
- 1.47 MeshEntry
- 1.48 MgBlock
- 1.49 Msg
- 1.50 Mstring
- 1.51 NMDAChan
- 1.52 Nernst
- 1.53 NeuroMesh
- 1.54 Neuron
- 1.55 Neutral
- 1.56 OneToAllMsg
- 1.57 OneToOneMsg
- 1.58 PIDController
- 1.59 Panel
- 1.60 Pool
- 1.61 PoolBase
- 1.62 Port
- 1.63 PulseGen
- 1.64 RC
- 1.65 Reac
- 1.66 ReacBase
- 1.67 RectPanel
- 1.68 ReduceMsg
- 1.69 Shell
- 1.70 SimManager
- 1.71 SingleMsg
- 1.72 SparseMsg
- 1.73 Species
- 1.74 SpherePanel
- 1.75 SpikeGen
- 1.76 Stats
- 1.77 StimulusTable
- 1.78 Stoich
- 1.79 SumFunc
- 1.80 Surface
- 1.81 SymCompartment
- 1.82 SynBase
- 1.83 SynChan
- 1.84 SynChanBase
- 1.85 Synapse

- 1.86 Table
- 1.87 TableBase
- 1.88 TableEntry
- 1.89 Tick
- 1.90 TriPanel
- 1.91 VectorTable
- 1.92 ZombieBufPool
- 1.93 ZombieCaConc
- 1.94 ZombieCompartment
- 1.95 ZombieEnz
- 1.96 ZombieFuncPool
- 1.97 ZombieHHChannel
- 1.98 ZombieMMenz
- 1.99 ZombiePool
- 1.100 ZombieReac
- 1.101 ZombieSumFunc
- 1.102 testSched

- 2 MOOSE Functions
  - 2.1 ce
  - 2.2 connect
  - 2.3 copy
  - 2.4 delete
  - 2.5 element
  - 2.6 exists
  - 2.7 getCwe
  - 2.8 getField
  - 2.9 getFieldDict
  - 2.10 getFieldNames
  - 2.11 isRunning
  - 2.12 loadModel
  - 2.13 move
  - 2.14 quit
  - 2.15 reinit
  - 2.16 saveModel
  - 2.17 seed
  - 2.18 setClock
  - 2.19 setCwe
  - 2.20 start
  - 2.21 stop
  - 2.22 useClock
  - 2.23 wildcardFind
  - 2.24 writeSBML
  - 2.25 doc
  - 2.26 getfielddoc
  - 2.27 getmoosedoc
  - 2.28 le
  - 2.29 listmsg
  - 2.30 pwe

- 2.31 showfield
- 2.32 showfields
- 2.33 showmsg
- 2.34 syncDataHandler

1 MOOSE Classes

1.1 Annotator

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

x: ***double***

x field. Typically display coordinate x

y: ***double***

y field. Typically display coordinate y

z: ***double***

z field. Typically display coordinate z

notes: ***string***

A string to hold some text notes about parent object

color: ***string***

A string to hold a text string specifying display color.Can be a regular English color name, or an rgb code rrrgggbbb

textColor: ***string***

A string to hold a text string specifying color for text labelthat might be on the display for this object.Can be a regular English color name, or an rgb code rrrgggbbb

icon: ***string***

A string to specify icon to use for display

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.2 Arith

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

function: ***string***

Arithmetic function to perform on inputs.

outputValue: ***double***

Value of output as computed last timestep.

arg1Value: ***double***

Value of arg1 as computed last timestep.

- **Source message field**

childMsg: ***int***

Message to child Elements

output: ***double***

Sends out the computed value

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  arg1: ***double***

  Handles argument 1. This just assigns it

  arg2: ***double***

  Handles argument 2. This just assigns it

  arg3: ***double***

  Handles argument 3. This sums in each input, and clears each clock tick.

  arg1x2: ***double,double***

  Store the product of the two arguments in output_

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  anyValue: ***unsigned int,double***

  Value of any of the internal fields, output, arg1, arg2, arg3,as specified by the index argument from 0 to 3.

1.3 Boundary

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

reflectivity: ***double***

What happens to a molecule hitting it: bounces, absorbed, diffused?

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toAdjacent: ***void***

  Dummy message going to adjacent compartment.

  toInside: ***void***

  Dummy message going to surrounded compartment.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  adjacent: ***void***

  Dummy message coming from adjacent compartment to current oneImplies that compts are peers: do not surround each other

  outside: ***void***

  Dummy message coming from surrounding compartment to this one.Implies that the originating compartment surrounds this one

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.4 BufPool

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

increment: ***double***

Increments mol numbers by specified amount. Can be +ve or -ve

decrement: ***double***

Decrements mol numbers by specified amount. Can be +ve or -ve

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

- **Shared message field**

  reac: ***void***

  Connects to reaction

  proc: ***void***

  Shared message for process and reinit

  species: ***void***

  Shared message for connecting to species objects

  mesh: ***void***

  Shared message for dealing with mesh operations

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.5 CaConc

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Ca: ***double***

Calcium concentration.

CaBasal: ***double***

Basal Calcium concentration.

Ca_{base}: ***double***

Basal Calcium concentration, synonym for CaBasal

tau: ***double***

Settling time for Ca concentration

B: ***double***

Volume scaling factor

thick: ***double***

Thickness of Ca shell.

ceiling: ***double***

Ceiling value for Ca concentration. If Ca > ceiling, Ca = ceiling. If ceiling <= 0.0, there is no upper limit on Ca concentration value.

floor: ***double***

Floor value for Ca concentration. If Ca < floor, Ca = floor

- **Source message field**

childMsg: ***int***

Message to child Elements

concOut: ***double***

Concentration of Ca in pool

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

current: ***double***

Calcium Ion current, due to be converted to conc.

currentFraction: ***double,double***

Fraction of total Ion current, that is carried by Ca2+.

increase: ***double***

Any input current that increases the concentration.

decrease: ***double***

Any input current that decreases the concentration.

basal: ***double***

Synonym for assignment of basal conc.

- **Shared message field**

proc: ***void***

Shared message to receive Process message from scheduler

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.6 ChanBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  channelOut: ***double,double***

  Sends channel variables Gk and Ek to compartment

  permeability: ***double***

  Conductance term going out to GHK object

  IkOut: ***double***

  Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  Vm: ***double***

  Handles Vm message coming in from compartment

  Vm: ***double***

  Handles Vm message coming in from compartment

- **Shared message field**

  channel: ***void***

  This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

  ghk: ***void***

  Message to Goldman-Hodgkin-Katz object

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.7 ChemMesh

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

size: ***double***

Size of entire chemical domain.Assigning this assumes that the geometry is that of the default mesh, which may not be what you want. If so, usea more specific mesh assignment function.

numDimensions: ***unsigned int***

Number of spatial dimensions of this compartment. Usually 3 or 2

- **Source message field**

childMsg: ***int***

Message to child Elements

meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

Defines how meshEntries communicate between nodes.Args: oldVol, volListOfAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#]This message is meant to go to the SimManager and Stoich.

meshStats: ***unsigned int,vector<double>***

Basic statistics for mesh: Total # of entries, and a vector ofunique volumes of voxels

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

buildDefaultMesh: ***double,unsigned int***

Tells ChemMesh derived class to build a default mesh with thespecified size and number of meshEntries.

handleRequestMeshStats: ***void***

Handles request from SimManager for mesh stats

handleNodeInfo: ***unsigned int,unsigned int***

Tells ChemMesh how many nodes and threads per node it is allowed to use. Triggers a return meshSplit message.

- **Shared message field**

  nodeMeshing: ***void***

  Connects to SimManager to coordinate meshing with paralleldecomposition and with the Stoich

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.8 Cinfo

Author: Upi Bhalla

Description: Class information object.

Name: Cinfo

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

docs: ***string***

Documentation

baseClass: ***string***

Name of base class

- **Source message field**

childMsg: ***int***

Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.9 Clock

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest

ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

runTime: ***double***

Duration to run the simulation

currentTime: ***double***

Current simulation time

nsteps: ***unsigned int***

Number of steps to advance the simulation, in units of the smallest timestep on the clock ticks

numTicks: ***unsigned int***

Number of clock ticks

currentStep: ***unsigned int***

Current simulation step

dts: ***vector<double>***

Utility function returning the dt (timestep) of all ticks.

isRunning: ***bool***

Utility function to report if simulation is in progress.

- **Source message field**

childMsg: ***int***

Message to child Elements

childTick: ***void***

Parent of Tick element

finished: ***void***

Signal for completion of run

ack: ***unsigned int,unsigned int***

Acknowledgement signal for receipt/completion of function.Goes back to Shell on master node

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

start: ***double***

Sets off the simulation for the specified duration

step: ***unsigned int***

Sets off the simulation for the specified # of steps

stop: ***void***

Halts the simulation, with option to restart seamlessly

setupTick: ***unsigned int,double***

Sets up a specific clock tick: args tick#, dt

reinit: ***void***

Zeroes out all ticks, starts at t = 0

- **Shared message field**

clockControl: ***void***

Controls all scheduling aspects of Clock, usually from Shell

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.10 Compartment

Author: Upi Bhalla

Description: Compartment object, for branching neuron models.

Name: Compartment

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the

actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Vm: ***double***

membrane potential

Cm: ***double***

Membrane capacitance

Em: ***double***

Resting membrane potential

Im: ***double***

Current going through membrane

inject: ***double***

Current injection to deliver into compartment

initVm: ***double***

Initial value for membrane potential

Rm: ***double***

Membrane resistance

Ra: ***double***

Axial resistance of compartment

diameter: ***double***

Diameter of compartment

length: ***double***

Length of compartment

x0: ***double***

X coordinate of start of compartment

y0: ***double***

Y coordinate of start of compartment

z0: ***double***

Z coordinate of start of compartment

x: ***double***

x coordinate of end of compartment

y: ***double***

y coordinate of end of compartment

z: ***double***

z coordinate of end of compartment

- **Source message field**

childMsg: ***int***

Message to child Elements

VmOut: ***double***

Sends out Vm value of compartment on each timestep

axialOut: ***double***

Sends out Vm value of compartment to adjacent compartments,on each timestep

raxialOut: ***double,double***

Sends out Raxial information on each timestep, fields are Ra and Vm

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

randInject: ***double,double***

Sends a random injection current to the compartment. Must beupdated each timestep.Arguments to randInject are probability and current.

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

cable: ***void***

Message for organizing compartments into groups, calledcables. Doesn't do anything.

process: ***void***

Handles 'process' call

reinit: ***void***

Handles 'reinit' call

initProc: ***void***

Handles Process call for the 'init' phase of the Compartment calculations. These occur as a separate Tick cycle from the regular proc cycle, and should be called before the proc msg.

initReinit: ***void***

Handles Reinit call for the 'init' phase of the Compartment calculations.

handleChannel: ***double,double***

Handles conductance and Reversal potential arguments from Channel

handleRaxial: ***double,double***

Handles Raxial info: arguments are Ra and Vm.

handleAxial: ***double***

Handles Axial information. Argument is just Vm.

- **Shared message field**

proc: ***void***

This is a shared message to receive Process messages from the scheduler objects. The Process should be called second in each clock tick, after the Init message.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

init: ***void***

This is a shared message to receive Init messages from the scheduler objects. Its job is to separate the compartmental calculations from the message passing. It doesn't really need to be shared, as it does not use the reinit part, but the scheduler objects expect this form of message for all scheduled output. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a dummy MsgDest for the Reinit operation. It also uses ProcInfo.

channel: ***void***

This is a shared message from a compartment to channels. The first entry is a MsgDest for the info coming from the channel. It expects Gk and Ek from the channel as args. The second entry is a MsgSrc sending Vm

axial: ***void***

This is a shared message between asymmetric compartments. axial messages (this kind) connect up to raxial messages (defined below). The soma should use raxial messages to connect to the axial message of all the immediately adjacent dendritic compartments.This puts the (low) somatic resistance in series with these dendrites. Dendrites should then use raxial messages toconnect on to more distal dendrites. In other words, raxial messages should face outward from the soma. The first entry is a MsgSrc sending Vm to the axialFuncof the target compartment. The second entry is a MsgDest for the info coming from the other compt. It expects Ra and Vm from the other compt as args. Note that the message is named after the source type.

raxial: ***void***

This is a raxial shared message between asymmetric compartments. The first entry is a

MsgDest for the info coming from the other compt. It expects Vm from the other compt as an arg. The second is a MsgSrc sending Ra and Vm to the raxialFunc of the target compartment.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.11 CplxEnzBase

Author: Upi Bhalla

Description:: Base class for mass-action enzymes in which there is an explicit pool for the enzyme-substrate complex. It models the reaction: E + S <===> E.S -—> E + P

Name: CplxEnzBase

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes

field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

k1: ***double***

Forward reaction from enz + sub to complex

k2: ***double***

Reverse reaction from complex to enz + sub

k3: ***double***

Forward rate constant from complex to product + enz

ratio: ***double***

Ratio of k2/k3

concK1: ***double***

K1 expressed in concentration (1/millimolar.sec) units

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

toEnz: ***double,double***

Sends out increment of molecules on product each timestep

toCplx: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

enzDest: ***double***

Handles # of molecules of Enzyme

subDest: ***double***

Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product. Dummy.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the MMEnz to recompute its numKm after remeshing

enzDest: ***double***

Handles # of molecules of Enzyme

cplxDest: ***double***

Handles # of molecules of enz-sub complex

- **Shared message field**

sub: ***void***

Connects to substrate molecule

prd: ***void***

Connects to product molecule

proc: ***void***

Shared message for process and reinit

enz: ***void***

Connects to enzyme pool

cplx: ***void***

Connects to enz-sub complex pool

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.12 CubeMesh

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

size: ***double***

Size of entire chemical domain.Assigning this assumes that the geometry is that of the default mesh, which may not be what you want. If so, usea more specific mesh assignment function.

numDimensions: ***unsigned int***

Number of spatial dimensions of this compartment. Usually 3 or 2

isToroid: ***bool***

Flag. True when the mesh should be toroidal, that is,when going beyond the right face brings us around to theleft-most mesh entry, and so on. If we have nx, ny, nzentries, this rule means that the coordinate (x, ny, z)will map onto (x, 0, z). Similarly,(-1, y, z) -> (nx-1, y, z)Default is false

preserveNumEntries: ***bool***

Flag. When it is true, the numbers nx, ny, nz remainunchanged when x0, x1, y0, y1, z0, z1 are altered. Thusdx, dy, dz would change instead. When it is false, thendx, dy, dz remain the same and nx, ny, nz are altered.Default is true

x0: ***double***

X coord of one end

y0: ***double***

Y coord of one end

z0: ***double***

Z coord of one end

x1: ***double***

X coord of other end

y1: ***double***

Y coord of other end

z1: ***double***

Z coord of other end

dx: ***double***

X size for mesh

dy: ***double***

Y size for mesh

dz: ***double***

Z size for mesh

nx: ***unsigned int***

Number of subdivisions in mesh in X

ny: ***unsigned int***

Number of subdivisions in mesh in Y

nz: ***unsigned int***

Number of subdivisions in mesh in Z

coords: ***vector<double>***

Set all the coords of the cuboid at once. Order is:x0 y0 z0 x1 y1 z1 dx dy dz

meshToSpace: ***vector<unsigned int>***

Array in which each mesh entry stores spatial (cubic) index

spaceToMesh: ***vector<unsigned int>***

Array in which each space index (obtained by linearizing the xyz coords) specifies which meshIndex is present.In many cases the index will store the EMPTY flag if there isno mesh entry at that spatial location

- **Source message field**

childMsg: ***int***

Message to child Elements

meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

Defines how meshEntries communicate between nodes.Args: oldVol, volListOfAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#]This

message is meant to go to the SimManager and Stoich.

meshStats: ***unsigned int,vector<double>***

Basic statistics for mesh: Total # of entries, and a vector ofunique volumes of voxels

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

buildDefaultMesh: ***double,unsigned int***

Tells ChemMesh derived class to build a default mesh with thespecified size and number of meshEntries.

handleRequestMeshStats: ***void***

Handles request from SimManager for mesh stats

handleNodeInfo: ***unsigned int,unsigned int***

Tells ChemMesh how many nodes and threads per node it is allowed to use. Triggers a return meshSplit message.

- **Shared message field**

nodeMeshing: ***void***

Connects to SimManager to coordinate meshing with paralleldecomposition and with the Stoich

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.13 CylMesh

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

size: ***double***

Size of entire chemical domain.Assigning this assumes that the geometry is that of the default mesh, which may not be what you want. If so, usea more specific mesh assignment function.

numDimensions: ***unsigned int***

Number of spatial dimensions of this compartment. Usually 3 or 2

x0: ***double***

x coord of one end

y0: ***double***

y coord of one end

z0: ***double***

z coord of one end

r0: ***double***

Radius of one end

x1: ***double***

x coord of other end

y1: ***double***

y coord of other end

z1: ***double***

z coord of other end

r1: ***double***

Radius of other end

lambda: ***double***

Length constant to use for subdivisionsThe system will attempt to subdivide using compartments oflength lambda on average. If the cylinder has different enddiameters r0 and r1, it will scale to smaller lengthsfor the smaller diameter end and vice versa.Once the value is set it will recompute lambda as totLength/numEntries

coords: ***vector<double>***

All the coords as a single vector: x0 y0 z0 x1 y1 z1 r0 r1 lambda

totLength: ***double***

Total length of cylinder

- **Source message field**

childMsg: ***int***

Message to child Elements

meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

Defines how meshEntries communicate between nodes.Args: oldVol, volListOfAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#]This message is meant to go to the SimManager and Stoich.

meshStats: ***unsigned int,vector<double>***

Basic statistics for mesh: Total # of entries, and a vector ofunique volumes of voxels

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

buildDefaultMesh: ***double,unsigned int***

Tells ChemMesh derived class to build a default mesh with thespecified size and number of meshEntries.

handleRequestMeshStats: ***void***

Handles request from SimManager for mesh stats

handleNodeInfo: ***unsigned int,unsigned int***

Tells ChemMesh how many nodes and threads per node it is allowed to use. Triggers a return meshSplit message.

- **Shared message field**

nodeMeshing: ***void***

Connects to SimManager to coordinate meshing with paralleldecomposition and with the Stoich

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.14 CylPanel

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toNeighbor: ***void***

  Identifies neighbors of the current panel

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

neighbor: ***void***

Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

x: ***unsigned int,double***

x coordinate identified by index

y: ***unsigned int,double***

y coordinate identified by index

z: ***unsigned int,double***

z coordinate identified by index

1.15 DiagonalMsg

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

stride: ***int***

The stride is the increment to the src DataId that gives thedest DataId. It can be positive or negative, but bounds checkingtakes place and it does not wrap around.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.16 DiffAmp

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

gain: ***double***

Gain of the amplifier. The output of the amplifier is the difference between the totals in plus and minus inputs multiplied by the gain. Defaults to 1

saturation: ***double***

Saturation is the bound on the output. If output goes beyond the +/-saturation range, it is truncated to the closer of +saturation and -saturation. Defaults to the maximum double precision floating point number representable on the system.

output: ***double***

Output of the amplifier, i.e. gain * (plus - minus).

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  outputOut: ***double***

  Current output level.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  gainIn: ***double***

  Destination message to control gain dynamically.

  plusIn: ***double***

  Positive input terminal of the amplifier. All the messages connected here are summed up to get total positive input.

  minusIn: ***double***

  Negative input terminal of the amplifier. All the messages connected here are summed up to get total positive input.

  process: ***void***

  Handles process call, updates internal time stamp.

  reinit: ***void***

  Handles reinit call.

- **Shared message field**

proc: ***void***

This is a shared message to receive Process messages from the scheduler objects.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.17 DiskPanel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

childMsg: ***int***

Message to child Elements

toNeighbor: ***void***

Identifies neighbors of the current panel

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  neighbor: ***void***

  Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  x: ***unsigned int,double***

  x coordinate identified by index

  y: ***unsigned int,double***

  y coordinate identified by index

  z: ***unsigned int,double***

  z coordinate identified by index

1.18 Enz

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

k1: ***double***

Forward reaction from enz + sub to complex

k2: ***double***

Reverse reaction from complex to enz + sub

k3: ***double***

Forward rate constant from complex to product + enz

ratio: ***double***

Ratio of k2/k3

concK1: ***double***

K1 expressed in concentration (1/millimolar.sec) units

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

toEnz: ***double,double***

Sends out increment of molecules on product each timestep

toCplx: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  enzDest: ***double***

  Handles # of molecules of Enzyme

  subDest: ***double***

  Handles # of molecules of substrate

  prdDest: ***double***

  Handles # of molecules of product. Dummy.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  remesh: ***void***

  Tells the MMEnz to recompute its numKm after remeshing

  enzDest: ***double***

  Handles # of molecules of Enzyme

  cplxDest: ***double***

  Handles # of molecules of enz-sub complex

- **Shared message field**

  sub: ***void***

  Connects to substrate molecule

  prd: ***void***

  Connects to product molecule

  proc: ***void***

  Shared message for process and reinit

  enz: ***void***

Connects to enzyme pool

cplx: ***void***

Connects to enz-sub complex pool

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.19 EnzBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  enzDest: ***double***

  Handles # of molecules of Enzyme

  subDest: ***double***

  Handles # of molecules of substrate

  prdDest: ***double***

  Handles # of molecules of product. Dummy.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  remesh: ***void***

  Tells the MMEnz to recompute its numKm after remeshing

- **Shared message field**

  sub: ***void***

  Connects to substrate molecule

  prd: ***void***

  Connects to product molecule

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.20 Finfo

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

name: ***string***

Name of Finfo

docs: ***string***

Documentation for Finfo

type: ***string***

RTTI type info for this Finfo

src: ***vector<string>***

Subsidiary SrcFinfos. Useful for SharedFinfos

dest: ***vector<string>***

Subsidiary DestFinfos. Useful for SharedFinfos

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.21 FuncPool

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

increment: ***double***

Increments mol numbers by specified amount. Can be +ve or -ve

decrement: ***double***

Decrements mol numbers by specified amount. Can be +ve or -ve

input: ***double***

Handles input to control value of n_

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.22 GHK

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Ik: ***double***

Membrane current

Gk: ***double***

Conductance

Ek: ***double***

Reversal Potential

T: ***double***

Temperature of system

p: ***double***

Permeability of channel

Vm: ***double***

Membrane potential

Cin: ***double***

Internal concentration

Cout: ***double***

External ion concentration

valency: ***double***

Valence of ion

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  channelOut: ***double,double***

  Sends channel variables Gk and Ek to compartment

  VmOut: ***double***

  Relay of membrane potential Vm.

  IkOut: ***double***

  MembraneCurrent.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  process: ***void***

  Handles process call

  handleVm: ***double***

  Handles Vm message coming in from compartment

  addPermeability: ***double***

  Handles permeability message coming in from channel

  CinDest: ***double***

Alias for set_{Cin}

CoutDest: ***double***

Alias for set_{Cout}

addPermeability: ***double***

Handles permeability message coming in from channel

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message from channel to current Goldman-Hodgkin-Katz objectThis shared message connects to an HHChannel. The first entry is a MsgSrc which relays the Vm received from a compartment. The second entry is a MsgDest which receives channel conductance, and interprets it as permeability.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.23 Geometry

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

epsilon: ***double***

epsilon is the max deviation of surface-point from surface.I think it refers to when the molecule is stuck to the surface. Need to check with Steven.

neighdist: ***double***

neighdist is capture distance from one panel to another.When a molecule diffuses off one panel and is within neighdist of the other, it is captured by the second.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  returnSize: ***double***

  Return size of compartment

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleSizeRequest: ***void***

  Handles a request for size. Part of SharedMsg to ChemCompt.

- **Shared message field**

  compt: ***void***

  Connects to compartment(s) to specify geometry.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.24 Group

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

- **Source message field**

childMsg: ***int***

Message to child Elements

group: ***void***

Handle for grouping Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.25 GslIntegrator

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

isInitialized: ***bool***

True if the Stoich message has come in to set parms

method: ***string***

Numerical method to use.

relativeAccuracy: ***double***

Accuracy criterion

absoluteAccuracy: ***double***

Another accuracy criterion

- **Source message field**

childMsg: ***int***

Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  stoich: ***Id***

  Handle data from Stoich

  remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

  Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.26 GssaStoich

Author: Upinder S. Bhalla, 2008, 2011, NCBS

Description: GssaStoich: Gillespie Stochastic Simulation Algorithm object.Closely based on the Stoich object and inherits its handling functions for constructing the matrix. Sets up stoichiometry matrix based calculations from a wildcard path for the reaction system.Knows how to compute derivatives for most common things, also knows how to handle special cases where the object will have to do its own computation.Generates a stoichiometry matrix, which is useful for lots of other operations as well.

Name: GssaStoich

- **Value field**

  this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

useOneWayReacs: ***bool***

Flag: use bidirectional or one-way reacs. One-way is neededfor Gillespie type stochastic calculations. Two-way islikely to be margninally more efficient in ODE calculations

nVarPools: ***unsigned int***

Number of variable molecule pools in the reac system

numMeshEntries: ***unsigned int***

Number of meshEntries in reac-diff system

estimatedDt: ***double***

Estimate of fastest (smallest) timescale in system.This is fallible because it depends on instantaneous concs,which of course change over the course of the simulation.

path: ***string***

Path of reaction system to take over

path: ***string***

Path of reaction system to take over and solve

method: ***string***

Numerical method to use for the GssaStoich. The defaultand currently the only method is Gillespie1.

- **Source message field**

childMsg: ***int***

Message to child Elements

plugin: ***Id***

Sends out Stoich Id so that plugins can directly access fields and functions

nodeDiffBoundary: ***unsigned int,vector<unsigned int>,vector<double>***

Sends mol #s across boundary between nodes, to calculate diffusionterms. arg1 is originating node, arg2 is list of meshIndices forwhich data is being transferred, and arg3 are

the 'n' values forall the pools on the specified meshIndices, to be plugged intothe appropriate place on the recipient node's S_ matrix

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

  Handles message from ChemMesh that defines how meshEntries are decomposed on this node, and how they communicate between nodes.Args: (oldVol, volumeVectorForAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#])

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinint call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.27 HDF5DataWriter

Author: Subhasis Ray

Description: HDF5 file writer for saving data tables. It saves the tables added to it via addObject function into an HDF5 file. At every process call it writes the contents of the tables to the file and clears the table vectors. You can explicitly save the data via the flush function.

Name: HDF5DataWriter

- **Value field**

  this: ***Neutral***

  Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

filename: ***string***

Name of the file associated with this HDF5 writer object.

isOpen: ***bool***

True if this object has an open file handle.

mode: ***unsigned int***

Depending on mode, if file already exists, if mode=1, data will be appended to existing file, if mode=2, file will be truncated, if mode=4, no writing will happen.

- **Source message field**

childMsg: ***int***

Message to child Elements

requestData: ***unsigned int***

Sends request for a field to target object

clear: ***void***

Send request to clear a Table vector.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

flush: ***void***

Write all buffer contents to file and clear the buffers.

recvData: ***bad***

Handles data sent back following request

process: ***void***

Handle process calls. Write data to file and clear all Table objects associated with this.

reinit: ***void***

Reinitialize the object

- **Shared message field**

  proc: ***void***

  Shared message to receive process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.28 HDF5WriterBase

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

filename: ***string***

Name of the file associated with this HDF5 writer object.

isOpen: ***bool***

True if this object has an open file handle.

mode: ***unsigned int***

Depending on mode, if file already exists, if mode=1, data will be appended to existing file, if mode=2, file will be truncated, if mode=4, no writing will happen.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

flush: ***void***

Write all buffer contents to file and clear the buffers.

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.29 HHChannel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

Xpower: ***double***

Power for X gate

Ypower: ***double***

Power for Y gate

Zpower: ***double***

Power for Z gate

instant: ***int***

Bitmapped flag: bit 0 = Xgate, bit 1 = Ygate, bit 2 = ZgateWhen true, specifies that the lookup table value should beused directly as the state of the channel, rather than usedas a rate term for numerical integration for the state

X: ***double***

State variable for X gate

Y: ***double***

State variable for Y gate

Z: ***double***

State variable for Y gate

useConcentration: ***int***

Flag: when true, use concentration message rather than Vm tocontrol Z gate

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

concen: ***double***

Incoming message from Concen object to specific conc to usein the Z gate calculations

createGate: ***string***

Function to create specified gate.Argument: Gate type [X Y Z]

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.30 HHChannel2D

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

Xindex: ***string***

String for setting X index.

Yindex: ***string***

String for setting Y index.

Zindex: ***string***

String for setting Z index.

Xpower: ***double***

Power for X gate

Ypower: ***double***

Power for Y gate

Zpower: ***double***

Power for Z gate

instant: ***int***

Bitmapped flag: bit 0 = Xgate, bit 1 = Ygate, bit 2 = ZgateWhen true, specifies that the lookup table value should beused directly as the state of the channel, rather than usedas a rate term for numerical integration for the state

X: ***double***

State variable for X gate

Y: ***double***

State variable for Y gate

Z: ***double***

State variable for Y gate

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

concen: ***double***

Incoming message from Concen object to specific conc to useas the first concen variable

concen2: ***double***

Incoming message from Concen object to specific conc to useas the second concen variable

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.31 HHGate

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

alpha: ***vector<double>***

Parameters for voltage-dependent rates, alpha:Set up alpha term using 5 parameters, as follows:y(x) = (A + B * x) / (C + exp((x + D) / F))The original HH equations can readily be cast into this form

beta: ***vector<double>***

Parameters for voltage-dependent rates, beta:Set up beta term using 5 parameters, as

follows:y(x) = (A + B * x) / (C + exp((x + D) / F))The original HH equations can readily be cast into this form

tau: ***vector<double>***

Parameters for voltage-dependent rates, tau:Set up tau curve using 5 parameters, as follows:y(x) = (A + B * x) / (C + exp((x + D) / F))

mInfinity: ***vector<double>***

Parameters for voltage-dependent rates, mInfinity:Set up mInfinity curve using 5 parameters, as follows:y(x) = (A + B * x) / (C + exp((x + D) / F))The original HH equations can readily be cast into this form

min: ***double***

Minimum range for lookup

max: ***double***

Minimum range for lookup

divs: ***unsigned int***

Divisions for lookup. Zero means to use linear interpolation

tableA: ***vector<double>***

Table of A entries

tableB: ***vector<double>***

Table of alpha + beta entries

useInterpolation: ***bool***

Flag: use linear interpolation if true, else direct lookup

- **Source message field**

childMsg: ***int***

Message to child Elements

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

setupAlpha: ***vector<double>***

Set up both gates using 13 parameters, as follows:setupAlpha AA AB AC AD AF BA BB

BC BD BF xdivs xmin xmaxHere AA-AF are Coefficients A to F of the alpha (forward) termHere BA-BF are Coefficients A to F of the beta (reverse) termHere xdivs is the number of entries in the table,xmin and xmax define the range for lookup.Outside this range the returned value will be the low [high]entry of the table.The equation describing each table is:y(x) = (A + B * x) / (C + exp((x + D) / F))The original HH equations can readily be cast into this form

setupTau: ***vector<double>***

Identical to setupAlpha, except that the forms specified bythe 13 parameters are for the tau and m-infinity curves ratherthan the alpha and beta terms. So the parameters are:setupTau TA TB TC TD TF MA MB MC MD MF xdivs xmin xmaxAs before, the equation describing each curve is:y(x) = (A + B * x) / (C + exp((x + D) / F))

tweakAlpha: ***void***

Dummy function for backward compatibility. It used to convertthe tables from alpha, beta values to alpha, alpha+betabecause the internal calculations used these forms. Notneeded now, deprecated.

tweakTau: ***void***

Dummy function for backward compatibility. It used to convertthe tables from tau, minf values to alpha, alpha+betabecause the internal calculations used these forms. Notneeded now, deprecated.

setupGate: ***vector<double>***

Sets up one gate at a time using the alpha/beta form.Has 9 parameters, as follows:setupGate A B C D F xdivs xmin xmax is$_{\text{betaThis}}$ sets up the gate using the equation:y(x) = (A + B * x) / (C + exp((x + D) / F))Deprecated.

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

A: ***double,double***

lookupA: Look up the A gate value from a double. Usually doesso by direct scaling and offset to an integer lookup, usinga fine enough table granularity that there is little error.Alternatively uses linear interpolation.The range of the double is predefined based on knowledge ofvoltage or conc ranges, and the granularity is specified bythe xmin, xmax, and dV fields.

B: ***double,double***

lookupB: Look up the B gate value from a double.Note that this looks up the raw tables, which are transformedfrom the reference parameters.

1.32 HHGate2D

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  A: ***vector<double>,double***

  lookupA: Look up the A gate value from two doubles, passedin as a vector. Uses linear interpolation in the 2D tableThe range of the lookup doubles is predefined based on knowledge of voltage or conc ranges, and the granularity is specified by the xmin, xmax, and dx field, and their y-axis counterparts.

  B: ***vector<double>,double***

  lookupB: Look up B gate value from two doubles in a vector.

1.33 HSolve

- **Value field**

  this: ***Neutral***

  Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

seed: ***Id***

Use this field to specify path to a 'seed' compartment, that is, any compartment within a neuron. The HSolve object uses this seed as a handle to discover the rest of the neuronal model, which means all the remaining compartments, channels, synapses, etc.

target: ***string***

Specifies the path to a compartmental model to be taken over. This can be the path to any container object that has the model under it (found by performing a deep search). Alternatively, this can also be the path to any compartment within the neuron. This compartment will be used as a handle to discover the rest of the model, which means all the remaining compartments, channels, synapses, etc.

dt: ***double***

The time-step for this solver.

caAdvance: ***int***

This flag determines how current flowing into a calcium pool is computed. A value of 0 means that the membrane potential at the beginning of the time-step is used for the calculation. This is how GENESIS does its computations. A value of 1 means the membrane potential at the middle of the time-step is used. This is the correct way of integration, and is the default way.

vDiv: ***int***

Specifies number of divisions for lookup tables of voltage-sensitive channels.

vMin: ***double***

Specifies the lower bound for lookup tables of voltage-sensitive channels. Default is to automatically decide based on the tables of the channels that the solver reads in.

vMax: ***double***

Specifies the upper bound for lookup tables of voltage-sensitive channels. Default is to automatically decide based on the tables of the channels that the solver reads in.

caDiv: ***int***

Specifies number of divisions for lookup tables of calcium-sensitive channels.

caMin: ***double***

Specifies the lower bound for lookup tables of calcium-sensitive channels. Default is to automatically decide based on the tables of the channels that the solver reads in.

caMax: ***double***

Specifies the upper bound for lookup tables of calcium-sensitive channels. Default is to automatically decide based on the tables of the channels that the solver reads in.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  process: ***void***

  Handles 'process' call: Solver advances by one time-step.

  reinit: ***void***

  Handles 'reinit' call: Solver reads in model.

- **Shared message field**

  proc: ***void***

  Handles 'reinit' and 'process' calls from a clock.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.34 HemispherePanel

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toNeighbor: ***void***

  Identifies neighbors of the current panel

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  neighbor: ***void***

  Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  x: ***unsigned int,double***

  x coordinate identified by index

y: ***unsigned int,double***

y coordinate identified by index

z: ***unsigned int,double***

z coordinate identified by index

1.35 IntFire

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

numSynapses: ***unsigned int***

Number of synapses on SynBase

Vm: ***double***

Membrane potential

tau: ***double***

charging time-course

thresh: ***double***

firing threshold

refractoryPeriod: ***double***

Minimum time between successive spikes

- **Source message field**

childMsg: ***int***

Message to child Elements

spike: ***double***

Sends out spike events

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.36 Interpol2D

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

xmin: ***double***

Minimum value for x axis of lookup table

xmax: ***double***

Maximum value for x axis of lookup table

xdivs: ***unsigned int***

# of divisions on x axis of lookup table

dx: ***double***

Increment on x axis of lookup table

ymin: ***double***

Minimum value for y axis of lookup table

ymax: ***double***

Maximum value for y axis of lookup table

ydivs: ***unsigned int***

# of divisions on y axis of lookup table

dy: ***double***

Increment on y axis of lookup table

tableVector2D: ***vector< vector<double> >***

Get the entire table.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  trig: ***double***

  respond to a request for a value lookup

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  lookup: ***double,double***

  Looks up table value based on indices v1 and v2, and sendsvalue back using the 'trig' message

- **Shared message field**

  lookupReturn2D: ***void***

  This is a shared message for doing lookups on the table. Receives 2 doubles: x, y. Sends back a double with the looked-up z value.

- **Lookup field**

  neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

table: ***vector<unsigned int>,double***

Lookup an entry on the table

z: ***vector<double>,double***

Interpolated value for specified x and y. This is provided for debugging. Normally other objects will retrieve interpolated values via lookup message.

1.37 IzhikevichNrn

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Vmax: ***double***

Maximum membrane potential. Membrane potential is reset to c whenever it reaches Vmax. NOTE: Izhikevich model specifies the PEAK voltage, rather than THRSHOLD voltage. The threshold depends on the previous history.

c: ***double***

Reset potential. Membrane potential is reset to c whenever it reaches Vmax.

d: ***double***

Parameter d in Izhikevich model. Unit is V/s.

a: ***double***

Parameter a in Izhikevich model. Unit is s^{-1}

b: ***double***

Parameter b in Izhikevich model. Unit is s^{-1}

u: ***double***

Parameter u in Izhikevich equation. Unit is V/s^{-1}

Vm: ***double***

Membrane potential, equivalent to v in Izhikevich equation.

Im: ***double***

Total current going through the membrane. Unit is A.

Rm: ***double***

Hidden cefficient of input current term (I) in Izhikevich model. Defaults to 1e6 Ohm.

initVm: ***double***

Initial membrane potential. Unit is V.

initU: ***double***

Initial value of u.

alpha: ***double***

Coefficient of v^2 in Izhikevich equation. Defaults to 0.04 in physiological unit. In SI it should be 40000.0. Unit is V^{-1} s^{-1}

beta: ***double***

Coefficient of v in Izhikevich model. Defaults to 5 in physiological unit, 5000.0 for SI units. Unit is s^{-1}

gamma: ***double***

Constant term in Izhikevich model. Defaults to 140 in both physiological and SI units. unit is V/s.

- **Source message field**

childMsg: ***int***

Message to child Elements

VmOut: ***double***

Sends out Vm

spike: ***double***

Sends out spike events

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

injectDest: ***double***

Injection current into the neuron.

cDest: ***double***

Destination message to modify parameter c at runtime.

dDest: ***double***

Destination message to modify parameter d at runtime.

bDest: ***double***

Destination message to modify parameter b at runtime

aDest: ***double***

Destination message modify parameter a at runtime.

- **Shared message field**

  proc: ***void***

  Shared message to receive Process message from scheduler

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.38 LeakyIaF

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Cm: ***double***

Membrane capacitance.

Rm: ***double***

Membrane resistance, inverse of leak-conductance.

Em: ***double***

Leak reversal potential

Vm: ***double***

Membrane potential

initVm: ***double***

Inital value of membrane potential

Vreset: ***double***

Reset potnetial after firing.

Vthreshold: ***double***

firing threshold

refractoryPeriod: ***double***

Minimum time between successive spikes

inject: ***double***

Injection current.

tSpike: ***double***

Time of the last spike

- **Source message field**

childMsg: ***int***

Message to child Elements

spike: ***double***

Sends out spike events

VmOut: ***double***

Sends out Vm

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  injectDest: ***double***

  Destination for current input.

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.39 MMenz

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

enzDest: ***double***

Handles # of molecules of Enzyme

subDest: ***double***

Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product. Dummy.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the MMEnz to recompute its numKm after remeshing

- **Shared message field**

sub: ***void***

Connects to substrate molecule

prd: ***void***

Connects to product molecule

proc: ***void***

Shared message for process and reinit

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.40 MarkovChannel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes

field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

ligandconc: ***double***

Ligand concentration.

vm: ***double***

Membrane voltage.

numstates: ***unsigned int***

The number of states that the channel can occupy.

numopenstates: ***unsigned int***

The number of states which are open/conducting.

state: ***vector<double>***

This is a row vector that contains the probabilities of finding the channel in each state.

initialstate: ***vector<double>***

This is a row vector that contains the probabilities of finding the channel in each state at t = 0. The state of the channel is reset to this value during a call to reinit()

labels: ***vector<string>***

Labels for each state.

gbar: ***vector<double>***

A row vector containing the conductance associated with each of the open/conducting states.

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleligandconc: ***double***

Deals with incoming messages containing information of ligand concentration

handlestate: ***vector<double>***

Deals with incoming message from MarkovSolver object containing state information of the channel.

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.41 MarkovGslSolver

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

isInitialized: ***bool***

True if the message has come in to set solver parameters.

method: ***string***

Numerical method to use.

relativeAccuracy: ***double***

Accuracy criterion

absoluteAccuracy: ***double***

Another accuracy criterion

internalDt: ***double***

internal timestep to use.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  stateOut: ***vector<double>***

  Sends updated state to the MarkovChannel class.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  init: ***vector<double>***

  Initialize solver parameters.

handleQ: ***vector< vector<double> >***

Handles information regarding the instantaneous rate matrix from the MarkovRateTable class.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

- **Shared message field**

proc: ***void***

Shared message for process and reinit

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.42 MarkovRateTable

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

vm: ***double***

Membrane voltage.

ligandconc: ***double***

Ligand concentration.

Q: ***vector< vector<double> >***

Instantaneous rate matrix.

size: ***unsigned int***

Dimension of the families of lookup tables. Is always equal to the number of states in the model.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  instratesOut: ***vector< vector<double> >***

  Sends out instantaneous rate information of varying transition rates at each time step.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleVm: ***double***

  Handles incoming message containing voltage information.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  init: ***unsigned int***

  Initialization of the class. Allocates memory for all the tables.

  handleLigandConc: ***double***

  Handles incoming message containing ligand concentration.

  set1d: ***unsigned int,unsigned int,Id,unsigned int***

  Setting up of 1D lookup table for the (i,j)'th rate.

  set2d: ***unsigned int,unsigned int,Id***

  Setting up of 2D lookup table for the (i,j)'th rate.

  setconst: ***unsigned int,unsigned int,double***

  Setting a constant value for the (i,j)'th rate. Internally, this is stored as a 1-D rate with a lookup table containing 1 entry.

- **Shared message field**

channel: ***void***

This message couples the rate table to the compartment. The rate table needs updates on voltage in order to compute the rate table.

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.43 MarkovSolver

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Q: ***vector< vector<double> >***

Instantaneous rate matrix.

state: ***vector<double>***

Current state of the channel.

initialstate: ***vector<double>***

Initial state of the channel.

xmin: ***double***

Minimum value for x axis of lookup table

xmax: ***double***

Maximum value for x axis of lookup table

xdivs: ***unsigned int***

# of divisions on x axis of lookup table

invdx: ***double***

Reciprocal of increment on x axis of lookup table

ymin: ***double***

Minimum value for y axis of lookup table

ymax: ***double***

Maximum value for y axis of lookup table

ydivs: ***unsigned int***

# of divisions on y axis of lookup table

invdy: ***double***

Reciprocal of increment on y axis of lookup table

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  stateOut: ***vector<double>***

  Sends updated state to the MarkovChannel class.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleVm: ***double***

  Handles incoming message containing voltage information.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  ligandconc: ***double***

Handles incoming message containing ligand concentration.

init: ***Id,double***

Setups the table of matrix exponentials associated with the solver object.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

- **Shared message field**

channel: ***void***

This message couples the MarkovSolverBase to the Compartment. The compartment needs Vm in order to look up the correct matrix exponential for computing the state.

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.44 MarkovSolverBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Q: ***vector< vector<double> >***

Instantaneous rate matrix.

state: ***vector<double>***

Current state of the channel.

initialstate: ***vector<double>***

Initial state of the channel.

xmin: ***double***

Minimum value for x axis of lookup table

xmax: ***double***

Maximum value for x axis of lookup table

xdivs: ***unsigned int***

# of divisions on x axis of lookup table

invdx: ***double***

Reciprocal of increment on x axis of lookup table

ymin: ***double***

Minimum value for y axis of lookup table

ymax: ***double***

Maximum value for y axis of lookup table

ydivs: ***unsigned int***

# of divisions on y axis of lookup table

invdy: ***double***

Reciprocal of increment on y axis of lookup table

- **Source message field**

childMsg: ***int***

Message to child Elements

stateOut: ***vector<double>***

Sends updated state to the MarkovChannel class.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleVm: ***double***

  Handles incoming message containing voltage information.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  ligandconc: ***double***

  Handles incoming message containing ligand concentration.

  init: ***Id,double***

  Setups the table of matrix exponentials associated with the solver object.

- **Shared message field**

  channel: ***void***

  This message couples the MarkovSolverBase to the Compartment. The compartment needs Vm in order to look up the correct matrix exponential for computing the state.

  proc: ***void***

  This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.45 MathFunc

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

mathML: ***string***

MathML version of expression to compute

function: ***string***

function is for functions of form f(x, y) = x + y

result: ***double***

result value

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  output: ***double***

  Sends out result of computation

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  arg1: ***double***

  Handle arg1

  arg2: ***double***

  Handle arg2

  arg3: ***double***

  Handle arg3

  arg4: ***double***

  Handle arg4

process: ***void***

Handle process call

reinit: ***void***

Handle reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.46 Mdouble

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

this: ***double***

Access function for entire Mdouble object.

value: ***double***

Access function for value field of Mdouble object,which happens also to be the entire contents of the object.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.47 MeshEntry

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

size: ***double***

Volume of this MeshEntry

dimensions: ***unsigned int***

number of dimensions of this MeshEntry

meshType: ***unsigned int***

The MeshType defines the shape of the mesh entry. 0: Not assigned 1: cuboid 2: cylinder 3. cylindrical shell 4: cylindrical shell segment 5: sphere 6: spherical shell 7: spherical shell segment 8: Tetrahedral

Coordinates: ***vector<double>***

Coordinates that define current MeshEntry. Depend on MeshType.

neighbors: ***vector<unsigned int>***

Indices of other MeshEntries that this one connects to

DiffusionArea: ***vector<double>***

Diffusion area for geometry of interface

DiffusionScaling: ***vector<double>***

Diffusion scaling for geometry of interface

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

  Tells the target pool or other entity that the compartment subdivision(meshing) has changed, and that it has to redo its volume and memory allocation accordingly.Arguments are: oldvol, numTotalEntries, startEntry, localIndices, volsThe vols specifies volumes of each local mesh entry. It also specifieshow many meshEntries are present on the local node.The localIndices vector is used for general load balancing only.It has a list of the all meshEntries on current node.If it is empty, we assume block load balancing. In this secondcase the contents of the current node go from startEntry to startEntry + vols.size().

  remeshReacs: ***void***

  Tells connected enz or reac that the compartment subdivision(meshing) has changed, and that it has to redo its volume-dependent rate terms like numKf_ accordingly.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  group: ***void***

  Handle for grouping. Doesn't do anything.

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

  mesh: ***void***

  Shared message for updating mesh volumes and subdivisions,typically controls pool sizes

- **Lookup field**

  neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.48 MgBlock

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

KMg_A: ***double***

1/eta

KMg_B: ***double***

1/gamma

CMg: ***double***

[Mg] in mM

Ik: ***double***

Current through MgBlock

Zk: ***double***

Charge on ion

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

origChannel: ***double,double***

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.49 Msg

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.50 Mstring

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

this: ***string***

Access function for entire Mstring object.

value: ***string***

Access function for value field of Mstring object,which happens also to be the entire contents of the object.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.51 NMDAChan

Author: Subhasis Ray, 2010, NCBS

Description: NMDAChan: Extracellular [Mg2+] dependent NMDA channel.This channel has four states as described by Jahr and Stevens (J. Neurosci. 1990, 10(9)) This implementation is based on equation 4(a) in that article. The channel conductance is defined as : k * g(V, [Mg2+]o) * S(t) where k is a scaling constant. S(t) is the legand gated component of the conductance. It rises linearly for t = tau2. Then decays exponentially with time constant t = tau1. g is a function of voltage and the extracellular [Mg2+] defined as: 1 / { 1 + (a1 + a2) * (a1 * B1 + a2 * B2)/ [A * a1 * (b1 + B1) + A * a2 * (b2 + B2)]} a1 = 1e3 * exp(- c0 * V - c1) s^{-1}, c0 = 16.0 / V, c1 = 2.91 a2 = 1e-3 * [Mg2+] * exp(-c2 * V - c3) mM^{-1} s, c2 = 45.0 / V, c3 = 6.97 b1 = 1e3 * exp(c4 * V + c5) s^{-1}, c4 = 9.0 / V, c5 = 1.22 b2 = 1e3 * exp(c6 * V + c7) s^{-1}, c6 = 17.0 / V, c7 = 0.96 A = 1e3 * exp(-c8) s^{-1}, c8 = 2.847 B1 = 1e3 * exp(-c9) s^{-1}, c9 = 0.693 s^{-1} B2 = 1e3 * exp(-c10) s^{-1}, c10 = 3.101. The behaviour of S(t) is as follows: If a spike arrives, then the slope of the linear rise of S(t) is incremented by weight / tau2. After tau2 time, this component is removed from the slope (reduced by weight/tau) and added over to the rate of decay of S(t).

Name: NMDAChan

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes

field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

numSynapses: ***unsigned int***

Number of synapses on SynBase

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

tau1: ***double***

Decay time constant for the synaptic conductance, tau1 >= tau2.

tau2: ***double***

Rise time constant for the synaptic conductance, tau1 >= tau2.

normalizeWeights: ***bool***

Flag. If true, the overall conductance is normalized by the number of individual synapses in this SynChan object.

unblocked: ***double***

Fraction of channels recovered from Mg2+ block. This is an intermediate variable which corresponds to g(V, [Mg2+]o) in the equation for conductance: k * g(V, [Mg2+]o) * S(t) where k is a constant.

MgConc: ***double***

External Mg2+ concentration

unblocked: ***double***

Fraction of channels recovered from Mg2+ block. This is an intermediate variable which corresponds to g(V, [Mg2+]o) in the equation for conductance: k * g(V, [Mg2+]o) * S(t) where k is a constant.

saturation: ***double***

Upper limit on the NMDA conductance.

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  Vm: ***double***

  Handles Vm message coming in from compartment

  Vm: ***double***

  Handles Vm message coming in from compartment

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  activation: ***double***

  Sometimes we want to continuously activate the channel

  modulator: ***double***

  Modulate channel response

  MgConcDest: ***double***

  Update [Mg2+] from other sources at every time step.

- **Shared message field**

  channel: ***void***

  This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

  ghk: ***void***

  Message to Goldman-Hodgkin-Katz object

  proc: ***void***

  Shared message to receive Process message from scheduler

- **Lookup field**

  neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

c: ***unsigned int,double***

Transition parameters c0 to c10 in the Mg2+ dependentstate transitions.

1.52 Nernst

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

E: ***double***

Computed reversal potential

Temperature: ***double***

Temperature of cell

valence: ***int***

Valence of ion in Nernst calculation

Cin: ***double***

Internal conc of ion

Cout: ***double***

External conc of ion

scale: ***double***

Voltage scale factor

- **Source message field**

childMsg: ***int***

Message to child Elements

Eout: ***double***

Computed reversal potential

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  ci: ***double***

  Set internal conc of ion, and immediately send out the updated E

  co: ***double***

  Set external conc of ion, and immediately send out the updated E

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.53 NeuroMesh

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

size: ***double***

Size of entire chemical domain.Assigning this assumes that the geometry is that of the default mesh, which may not be what you want. If so, usea more specific mesh assignment function.

numDimensions: ***unsigned int***

Number of spatial dimensions of this compartment. Usually 3 or 2

cell: ***Id***

Id for base element of cell model. Uses this to traverse theentire tree of the cell to build the mesh.

subTree: ***vector<Id>***

Set of compartments to model. If they happen to be contiguousthen also set up diffusion between the compartments. Can alsohandle cases where the same cell is divided into multiplenon-diffusively-coupled compartments

skipSpines: ***bool***

Flag: when skipSpines is true, the traversal does not includeany compartment with the string 'spine' or 'neck' in its name,and also then skips compartments below this skipped one.Allows to set up separate mesh for spines, based on the same cell model.

numSegments: ***unsigned int***

Number of cylindrical/spherical segments in model

numDiffCompts: ***unsigned int***

Number of diffusive compartments in model

diffLength: ***double***

Diffusive length constant to use for subdivisions. The system willattempt to subdivide cell using diffusive compartments ofthe specified diffusion lengths as a maximum.In order to get integral numbersof compartments in each segment, it may subdivide more finely.Uses default of 0.5 microns, that is, half typical lambda.For default, consider a tau of about 1 second for mostreactions, and a diffusion const of about 1e-12 um^2/sec.This gives lambda of 1 micron

geometryPolicy: ***string***

Policy for how to interpret electrical model geometry (which is a branching 1-dimensional tree) in terms of 3-D constructslike spheres, cylinders, and cones.There are three options, default, trousers, and cylinder:default mode: - Use frustrums of cones. Distal diameter is always from compt dia. - For linear dendrites (no branching), proximal diameter is diameter of the parent compartment - For branching dendrites and dendrites emerging from soma, proximal diameter is from compt dia. Don't worry about overlap. - Place somatic dendrites on surface of spherical soma, or at ends of cylindrical soma - Place dendritic spines on surface of cylindrical dendrites, not emerging from their middle.trousers mode: - Use frustrums of cones. Distal diameter is always from compt dia. - For linear dendrites (no branching), proximal diameter is diameter of the parent compartment - For branching dendrites, use a trouser function. Avoid overlap. - For soma, use some variant of trousers. Here we must avoid overlap - For spines, use a way to smoothly merge into parent dend. Radius of curvature should be similar to that of the spine neck. - Place somatic dendrites on surface of spherical soma, or at ends of cylindrical soma - Place dendritic spines on surface of cylindrical dendrites, not emerging from their middle.cylinder mode: - Use cylinders. Diameter is just compartment dia. - Place somatic dendrites on surface of spherical soma, or at ends of cylindrical soma - Place dendritic spines on surface of cylindrical dendrites, not emerging from their middle. - Ignore spatial overlap.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

  Defines how meshEntries communicate between nodes.Args: oldVol, volListOfAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#]This message is meant to go to the SimManager and Stoich.

  meshStats: ***unsigned int,vector<double>***

  Basic statistics for mesh: Total # of entries, and a vector ofunique volumes of voxels

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  buildDefaultMesh: ***double,unsigned int***

  Tells ChemMesh derived class to build a default mesh with thespecified size and number of meshEntries.

  handleRequestMeshStats: ***void***

  Handles request from SimManager for mesh stats

  handleNodeInfo: ***unsigned int,unsigned int***

  Tells ChemMesh how many nodes and threads per node it is allowed to use. Triggers a return meshSplit message.

- **Shared message field**

  nodeMeshing: ***void***

  Connects to SimManager to coordinate meshing with paralleldecomposition and with the Stoich

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.54 Neuron

Author: C H Chaitanya

Description: Neuron - A compartment container

Name: Neuron

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing

thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.55 Neutral

Author: Upinder S. Bhalla, 2007, NCBS

Description: Neutral: Base class for all MOOSE classes. Providesaccess functions for housekeeping fields and operations, messagetraversal, and so on.

Name: Neutral

- **Value field**

  this: ***Neutral***

  Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.56 OneToAllMsg

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

i1: ***DataId***

DataId of source Element.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.57 OneToOneMsg

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.58 PIDController

- **Value field**

  this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

gain: ***double***

This is the proportional gain (Kp). This tuning parameter scales the proportional term. Larger gain usually results in faster response, but too much will lead to instability and oscillation.

saturation: ***double***

Bound on the permissible range of output. Defaults to maximum double value.

command: ***double***

The command (desired) value of the sensed parameter. In control theory this is commonly known as setpoint(SP).

sensed: ***double***

Sensed (measured) value. This is commonly known as process variable(PV) in control theory.

tauI: ***double***

The integration time constant, typically = dt. This is actually proportional gain divided by integral gain (Kp/Ki)). Larger Ki (smaller tauI) usually leads to fast elimination of steady state errors at the cost of larger overshoot.

tauD: ***double***

The differentiation time constant, typically = dt / 4. This is derivative gain (Kd) times proportional gain (Kp). Larger Kd (tauD) decreases overshoot at the cost of slowing down transient response and may lead to instability.

output: ***double***

Output of the PIDController. This is given by: gain * (error + INTEGRAL[error dt] / tau_i + tau_d * d(error)/dt) Where gain = proportional gain (Kp), tau_i = integral gain (Kp/Ki) and tau_d = derivative gain (Kd/Kp). In control theory this is also known as the manipulated variable (MV)

error: ***double***

The error term, which is the difference between command and sensed value.

integral: ***double***

The integral term. It is calculated as INTEGRAL(error dt) = $previous_{integral}$ + dt * (error + $e_{previous}$)/2.

derivative: ***double***

The derivative term. This is (error - $e_{previous}$)/dt.

$e_{previous}$: ***double***

The error term for previous step.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  outputOut: ***double***

  Sends the output of the PIDController. This is known as manipulated variable (MV) in control theory. This should be fed into the process which we are trying to control.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  commandIn: ***double***

  Command (desired value) input. This is known as setpoint (SP) in control theory.

  sensedIn: ***double***

  Sensed parameter - this is the one to be tuned. This is known as process variable (PV) in control theory. This comes from the process we are trying to control.

  gainDest: ***double***

  Destination message to control the PIDController gain dynamically.

  process: ***void***

  Handle process calls.

  reinit: ***void***

  Reinitialize the object.

- **Shared message field**

proc: ***void***

This is a shared message to receive Process messages from the scheduler objects.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.59 Panel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

childMsg: ***int***

Message to child Elements

toNeighbor: ***void***

Identifies neighbors of the current panel

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

neighbor: ***void***

Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

x: ***unsigned int,double***

x coordinate identified by index

y: ***unsigned int,double***

y coordinate identified by index

z: ***unsigned int,double***

z coordinate identified by index

1.60 Pool

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

increment: ***double***

Increments mol numbers by specified amount. Can be +ve or -ve

decrement: ***double***

Decrements mol numbers by specified amount. Can be +ve or -ve

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.61 PoolBase

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.62 Port

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field

dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

scaleOutRate: ***double***

Scaling factor for outgoing rates. Applies to the RateTermscontrolled by this port. Represents a diffusion related term,or the permeability of the port

inStart: ***unsigned int***

Start index to S_ vector into which incoming molecules should add.

inEnd: ***unsigned int***

End index to S_ vector into which incoming molecules should add.

outStart: ***unsigned int***

Start index to S_ vector from where outgoing molecules come.

outEnd: ***unsigned int***

End index to S_ vector from where outgoing molecules come.

- **Source message field**

childMsg: ***int***

Message to child Elements

availableMolsAtPort: ***vector<Id>***

Sends out the full set of molecule Ids that are available for data transfer

efflux: ***vector<double>***

Molecule #s going out

matchedMolsAtPort: ***vector<Id>***

Sends out the set of molecule Ids that match between both ports

efflux: ***vector<double>***

Molecule #s going out

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleMatchedMolsAtPort: ***vector<unsigned int>***

  Handles list of matched molecules worked out by the other port

  influx: ***vector<double>***

  Molecule #s coming back in

  handleAvailableMolsAtPort: ***vector<unsigned int>***

  Handles list of all species that the other port cares about

  influx: ***vector<double>***

  Molecule #s coming back in

- **Shared message field**

  port1: ***void***

  Shared message for port. This one initiates the request forsetting up the communications between the portsThe shared message also handles the runtime data transfer

  port2: ***void***

  Shared message for port. This one responds to the request forsetting up the communications between the portsThe shared message also handles the runtime data transfer

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.63 PulseGen

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

output: ***double***

Output amplitude

baseLevel: ***double***

Basal level of the stimulus

firstLevel: ***double***

Amplitude of the first pulse in a sequence

firstWidth: ***double***

Width of the first pulse in a sequence

firstDelay: ***double***

Delay to start of the first pulse in a sequence

secondLevel: ***double***

Amplitude of the second pulse in a sequence

secondWidth: ***double***

Width of the second pulse in a sequence

secondDelay: ***double***

Delay to start of of the second pulse in a sequence

count: ***unsigned int***

Number of pulses in a sequence

trigMode: ***unsigned int***

Trigger mode for pulses in the sequence. 0 : free-running mode where it keeps looping its

output 1 : external trigger, where it is triggered by an external input (and stops after creating the first train of pulses) 2 : external gate mode, where it keeps generating the pulses in a loop as long as the input is high.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  outputOut: ***double***

  Current output level.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  input: ***double***

  Handle incoming input that determines gating/triggering onset.

  levelIn: ***unsigned int,double***

  Handle level value coming from other objects

  widthIn: ***unsigned int,double***

  Handle width value coming from other objects

  delayIn: ***unsigned int,double***

  Handle delay value coming from other objects

  process: ***void***

  Handles process call, updates internal time stamp.

  reinit: ***void***

  Handles reinit call.

- **Shared message field**

  proc: ***void***

  This is a shared message to receive Process messages from the scheduler objects.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  level: ***unsigned int,double***

  Level of the pulse at specified index

  width: ***unsigned int,double***

  Width of the pulse at specified index

  delay: ***unsigned int,double***

  Delay of the pulse at specified index

1.64 RC

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes

field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

V0: ***double***

Initial value of 'state'

R: ***double***

Series resistance of the RC circuit.

C: ***double***

Parallel capacitance of the RC circuit.

state: ***double***

Output value of the RC circuit. This is the voltage across the capacitor.

inject: ***double***

Input value to the RC circuit.This is handled as an input current to the circuit.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  outputOut: ***double***

  Current output level.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  injectIn: ***double***

  Receives input to the RC circuit. All incoming messages are summed up to give the total input current.

  process: ***void***

  Handles process call.

  reinit: ***void***

  Handle reinitialization

- **Shared message field**

  proc: ***void***

  This is a shared message to receive Process messages from the scheduler objects.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.65 Reac

- **Value field**

  this: ***Neutral***

  Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

kf: ***double***

Forward rate constant, in # units

kb: ***double***

Reverse rate constant, in # units

Kf: ***double***

Forward rate constant, in concentration units

Kb: ***double***

Reverse rate constant, in concentration units

numSubstrates: ***unsigned int***

Number of substrates of reaction

numProducts: ***unsigned int***

Number of products of reaction

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toSub: ***double,double***

  Sends out increment of molecules on product each timestep

  toPrd: ***double,double***

  Sends out increment of molecules on product each timestep

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  subDest: ***double***

  Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the reac to recompute its numRates, as remeshing has happened

- **Shared message field**

sub: ***void***

Connects to substrate pool

prd: ***void***

Connects to substrate pool

proc: ***void***

Shared message for process and reinit

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.66 ReacBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

kf: ***double***

Forward rate constant, in # units

kb: ***double***

Reverse rate constant, in # units

Kf: ***double***

Forward rate constant, in concentration units

Kb: ***double***

Reverse rate constant, in concentration units

numSubstrates: ***unsigned int***

Number of substrates of reaction

numProducts: ***unsigned int***

Number of products of reaction

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

subDest: ***double***

Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the reac to recompute its numRates, as remeshing has happened

- **Shared message field**

sub: ***void***

Connects to substrate pool

prd: ***void***

Connects to substrate pool

proc: ***void***

Shared message for process and reinit

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.67 RectPanel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toNeighbor: ***void***

  Identifies neighbors of the current panel

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  neighbor: ***void***

  Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  x: ***unsigned int,double***

  x coordinate identified by index

  y: ***unsigned int,double***

  y coordinate identified by index

  z: ***unsigned int,double***

  z coordinate identified by index

1.68 ReduceMsg

- **Value field**

  this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

i1: ***DataId***

DataId of source Element.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.69 Shell

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

- **Source message field**

childMsg: ***int***

Message to child Elements

reduceArraySize: ***unsigned int***

Look up maximum value of an index, here ragged array size,across many nodes, and assign uniformly to all nodes. Normallyfollowed by an operation to assign the size to the object thatwas resized.

requestCreate: ***string,Id,Id,string,vector<int>***

requestCreate(class, parent, newElm, name, dimensions): creates a new Element on all nodes with the specified Id. Initiates a callback to indicate completion of operation. Goes to all nodes including self.

requestDelete: ***Id***

requestDelete(doomedElement):Deletes specified Element on all nodes.Initiates a callback to indicate completion of operation.Goes to all nodes including self.

requestAddMsg: ***string,unsigned int,ObjId,string,ObjId,string***

requestAddMsg(type, src, srcField, dest, destField);Creates specified Msg between specified Element on all nodes.Initiates a callback to indicate completion of operation.Goes to all nodes including self.

requestQuit: ***void***

requestQuit():Emerges from the inner loop, and wraps up. No return value.

move: ***Id,Id***

move(origId, newParent);Moves origId to become a child of newParent

copy: ***vector<Id>,string,unsigned int,bool,bool***

copy(origId, newParent, numRepeats, toGlobal, copyExtMsg);Copies origId to become a child of newParent

useClock: ***string,string,unsigned int***

useClock(path, field, tick#);Specifies which clock tick to use for all elements in Path.The 'field' is typically process, but some cases need to sendupdates to the 'init' field.Tick # specifies which tick to be attached to the objects.

sync: ***Id,unsigned int***

sync(ElementId, FuncId);Synchronizes Element data indexing across all nodes.Used when distributed ops like message setup might set updifferent #s of data entries on Elements on different nodes.The ElementId is the element being synchronized.The FuncId is the 'get' function for the synchronized field.

requestReMesh: ***Id***

requestReMesh(meshId);Chops up specified mesh.

requestSetParserIdleFlag: ***bool***

SetParserIdleFlag(bool isParserIdle);When True, the main ProcessLoop waits a little each cycleso as to avoid pounding on the CPU.

ack: ***unsigned int,unsigned int***

ack(unsigned int node#, unsigned int status):Acknowledges receipt and completion of a command on a worker node.Goes back only to master node.

requestStart: ***double***

requestStart(runtime):Starts a simulation. Goes to all nodes including self.Initiates a callback to indicate completion of run.

requestStep: ***unsigned int***

requestStep():Advances a simulation for the specified # of steps.Goes to all nodes including self.

requestStop: ***void***

requestStop():Gently stops a simulation after completing current ops.After this op it is save to do 'start' again, and it willresume where it left offGoes to all nodes including self.

requestSetupTick: ***unsigned int,double***

requestSetupTick():Asks the Clock to coordinate the assignment of a specificclock tick. Args: Tick#, dt.Goes to all nodes including self.

requestReinit: ***void***

requestReinit():Reinits a simulation: sets to time 0.If simulation is running it stops it first.Goes to all nodes including self.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

receiveGet: ***bad***

receiveGet(Uint node#, Uint status, PrepackedBuffer data)Function on master shell that handles the value relayed from worker.

setclock: ***unsigned int,double,bool***

Assigns clock ticks. Args: tick#, dt

handleAck: ***unsigned int,unsigned int***

Keeps track of # of acks to a blocking shell command. Arg: Source node num.

create: ***string,Id,Id,string,vector<int>***

create(class, parent, newElm, name, dimensions)

delete: ***Id***

Destroys Element, all its messages, and all its children. Args: Id

handleAddMsg: ***string,unsigned int,ObjId,string,ObjId,string***

Makes a msg

handleQuit: ***void***

Stops simulation running and quits the simulator

move: ***Id,Id***

handleMove(Id orig, Id newParent): moves an Element to a new parent

handleCopy: ***vector<Id>,string,unsigned int,bool,bool***

handleCopy(vector< Id > args, string newName, unsigned int nCopies, bool toGlobal, bool copyExtMsgs): The vector< Id > has Id orig, Id newParent, Id newElm. This function

copies an Element and all its children to a new parent. May also expand out the original into nCopies copies. Normally all messages within the copy tree are also copied. If the flag copyExtMsgs is true, then all msgs going out are also copied.

handleUseClock: ***string,string,unsigned int***

Deals with assignment of path to a given clock.

handleSync: ***Id,unsigned int***

handleSync(Id Element): Synchronizes DataHandler indexing across nodesThe ElementId is the element being synchronized.The FuncId is the 'get' function for the synchronized field.

handleReMesh: ***Id***

handleReMesh(Id BaseMesh): Deals with outcome of resizing the meshing in a cellularcompartment (the ChemMesh class). The mesh change has topropagate down to the molecules and reactions managed by this.Mesh. The ElementId is the mesh being synchronized.

handleSetParserIdleFlag: ***bool***

handleSetParserIdleFlag(bool isParserIdle): When True, tells the ProcessLoop to wait as the Parser is idle.

handleAck: ***unsigned int,unsigned int***

Keeps track of # of acks to a blocking shell command. Arg: Source node num.

- **Shared message field**

master: ***void***

Issues commands from master shell to worker shells located on different nodes. Also handles acknowledgements from them.

worker: ***void***

Handles commands arriving from master shell on node 0.Sends out acknowledgements from them.

clockControl: ***void***

Controls the system Clock

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.70 SimManager

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

syncTime: ***double***

SyncTime is the interval between synchronizing solvers5 msec is a typical value

autoPlot: ***bool***

When the autoPlot flag is true, the simManager guesses whichplots are of interest, and builds them.

plotDt: ***double***

plotDt is the timestep for plotting variables. As most will bechemical, a default of 1 sec is reasonable

runTime: ***double***

runTime is the requested duration of the simulation that is stored in some kinds of model definition files.

method: ***string***

method is the numerical method used for the calculations.This will set up or even replace the solver with one ableto use the specified method. Currently works only with two solvers: GSL and GSSA.The GSL solver has a variety of ODE methods, by defaultRunge-Kutta-Fehlberg.The GSSA solver currently uses the Gillespie StochasticSystems Algorithm, somewhat optimized over the originalmethod.

version: ***unsigned int***

Numerical version number. Used by kkit

modelFamily: ***string***

Family classification of model: *kinetic, and *neuron are the options so far. In due course expect to see thingslike detailedNetwork, intFireNetwork, sigNeur and so on.

- **Source message field**

childMsg: ***int***

Message to child Elements

requestMeshStats: ***void***

Asks for basic stats for mesh:Total # of entries, and a vector of unique volumes of voxels

nodeInfo: ***unsigned int,unsigned int***

Sends out # of nodes to use for meshing, and # of threads to use on each node, to the ChemMesh. These numbers sometimesdiffer from the total # of nodes and threads, because the SimManager may have other portions of the model to allocate.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

build: ***string***

Sets up model, with the specified method. The method may beempty if the intention is that methods be set up through hints in the ChemMesh compartments.

makeStandardElements: ***string***

Sets up the usual infrastructure for a model, with theChemMesh, Stoich, solver and suitable messaging.The argument is the MeshClass to use.

meshSplit: ***double,vector<unsigned int>,vector<unsigned int>,vector<unsigned int>,vector<unsigned int>***

Handles message from ChemMesh that defines howmeshEntries communicate between nodes.First arg is oldvol, next is list of other nodes, third arg is list number ofmeshEntries to be transferred for each of these nodes, fourth arg is catenated list of meshEntries indices onmy node going to each of the other connected nodes, andlast arg is matching list of meshEntries on other nodes

meshStats: ***unsigned int,vector<double>***

Basic statistics for mesh: Total # of entries, and a vectorof unique volumes of voxels

- **Shared message field**

nodeMeshing: ***void***

Connects to ChemMesh to coordinate meshing with paralleldecomposition and with the Stoich

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.71 SingleMsg

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing

thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

i1: ***DataId***

Index of source object.

i2: ***DataId***

Index of dest object.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.72 SparseMsg

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

e1: ***Id***

Id of source Element.

e2: ***Id***

Id of source Element.

srcFieldsOnE1: ***vector<string>***

Names of SrcFinfos for messages going from e1 to e2. There arematching entries in the destFieldsOnE2 vector

destFieldsOnE2: ***vector<string>***

Names of DestFinfos for messages going from e1 to e2. There arematching entries in the srcFieldsOnE1 vector

srcFieldsOnE2: ***vector<string>***

Names of SrcFinfos for messages going from e2 to e1. There arematching entries in the destFieldsOnE1 vector

destFieldsOnE1: ***vector<string>***

Names of destFinfos for messages going from e2 to e1. There arematching entries in the srcFieldsOnE2 vector

numRows: ***unsigned int***

Number of rows in matrix.

numColumns: ***unsigned int***

Number of columns in matrix.

numEntries: ***unsigned int***

Number of Entries in matrix.

probability: ***double***

connection probability for random connectivity.

seed: ***long***

Random number seed for generating probabilistic connectivity.

- **Source message field**

childMsg: ***int***

Message to child Elements

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

setRandomConnectivity: ***double,long***

Assigns connectivity with specified probability and seed

setEntry: ***unsigned int,unsigned int,unsigned int***

Assigns single row,column value

unsetEntry: ***unsigned int,unsigned int***

Clears single row,column entry

clear: ***void***

Clears out the entire matrix

transpose: ***void***

Transposes the sparse matrix

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.73 Species

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest

ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

molWt: ***double***

Molecular weight of species

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  sendMolWt: ***double***

  returns molWt.

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  handleMolWtRequest: ***void***

  Handle requests for molWt.

- **Shared message field**

  pool: ***void***

  Connects to pools of this Species type

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.74 SpherePanel

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

childMsg: ***int***

Message to child Elements

toNeighbor: ***void***

Identifies neighbors of the current panel

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

neighbor: ***void***

Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

x: ***unsigned int,double***

x coordinate identified by index

y: ***unsigned int,double***

y coordinate identified by index

z: ***unsigned int,double***

z coordinate identified by index

1.75 SpikeGen

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

threshold: ***double***

Spiking threshold, must cross it going up

refractT: ***double***

Refractory Time.

$abs_{refract}$: *double*

Absolute refractory time. Synonym for refractT.

hasFired: *bool*

True if SpikeGen has just fired

edgeTriggered: *bool*

When edgeTriggered = 0, the SpikeGen will fire an event in each timestep while incoming Vm is > threshold and at least $abs_{refracttime}$ has passed since last event. This may be problematic if the incoming Vm remains above threshold for longer than $abs_{refract}$. Setting edgeTriggered to 1 resolves this as the SpikeGen generatesan event only on the rising edge of the incoming Vm and will remain idle unless the incoming Vm goes below threshold.

- **Source message field**

  childMsg: *int*

  Message to child Elements

  event: *double*

  Sends out a trigger for an event.

- **Destination message field**

  parentMsg: *int*

  Message from Parent Element(s)

  process: *void*

  Handles process call

  reinit: *void*

  Handles reinit call

  Vm: *double*

  Handles Vm message coming in from compartment

- **Shared message field**

  proc: *void*

  Shared message to receive Process message from scheduler

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.76 Stats

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

mean: ***double***

Mean of all sampled values.

sdev: ***double***

Standard Deviation of all sampled values.

sum: ***double***

Sum of all sampled values.

num: ***unsigned int***

Number of all sampled values.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  reduce: ***unsigned int***

  Execute statistics reduction operation on all targets andplace results in this object

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

trig: ***void***

Triggers Reduction operation.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.77 StimulusTable

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

vec: ***vector<double>***

vector with all table entries

outputValue: ***double***

Output value holding current table entry or output of a calculation

size: ***unsigned int***

size of table. Note that this is the number of x divisions +1since it must represent the largest value as well as thesmallest

startTime: ***double***

Start time used when table is emitting values. For lookupvalues below this, the table just sends out its zero entry.Corresponds to zeroth entry of table.

stopTime: ***double***

Time to stop emitting values.If time exceeds this, then the table sends out its last entry.The stopTime corresponds to the last entry of table.

loopTime: ***double***

If looping, this is the time between successive cycle starts.Defaults to the difference between stopTime and startTime, so that the output waveform cycles with precisely the same duration as the table contents.If larger than stopTime - startTime, then it pauses at the last table value till it is time to go around again.If smaller than stopTime - startTime, then it begins the next cycle even before the first one has reached the end of the table.

stepSize: ***double***

Increment in lookup (x) value on every timestep. If it isless than or equal to zero, the StimulusTable uses the current timeas the lookup value.

stepPosition: ***double***

Current value of lookup (x) value.If stepSize is less than or equal to zero, this is set tothe current time to use as the lookup value.

doLoop: ***bool***

Flag: Should it loop around to startTime once it has reachedstopTime. Default (zero) is to do a single pass.

- **Source message field**

childMsg: ***int***

Message to child Elements

output: ***double***

Sends out tabulated data according to lookup parameters.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

linearTransform: ***double,double***

Linearly scales and offsets data. Scale first, then offset.

xplot: ***string,string***

Dumps table contents to xplot-format file. Argument 1 is filename, argument 2 is plotname

plainPlot: ***string***

Dumps table contents to single-column ascii file. Uses scientific notation. Argument 1 is filename

loadCSV: ***string,int,int,char***

Reads a single column from a CSV file. Arguments: filename, column#, starting row#, separator

loadXplot: ***string,string***

Reads a single plot from an xplot file. Arguments: filename, plotnameWhen the file has 2 columns, the 2nd column is loaded.

loadXplotRange: ***string,string,unsigned int,unsigned int***

Reads a single plot from an xplot file, and selects a subset of points from it. Arguments: filename, plotname, startindex, endindexUses C convention: startindex included, endindex not included.When the file has 2 columns, the 2nd column is loaded.

compareXplot: ***string,string,string***

Reads a plot from an xplot file and compares with contents of TableBase.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: filename, plotname, $\text{comparison}_{\text{operationOperations}}$: rmsd (for RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

compareVec: ***vector<double>,string***

Compares contents of TableBase with a vector of doubles.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: Other vector, $\text{comparison}_{\text{operationOperations}}$: rmsd (for RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

clearVec: ***void***

Handles request to clear the data vector

process: ***void***

Handles process call, updates internal time stamp.

reinit: ***void***

Handles reinit call.

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  y: ***unsigned int,double***

  Value of table at specified index

1.78 Stoich

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

useOneWayReacs: ***bool***

Flag: use bidirectional or one-way reacs. One-way is neededfor Gillespie type stochastic calculations. Two-way islikely to be margninally more efficient in ODE calculations

nVarPools: ***unsigned int***

Number of variable molecule pools in the reac system

numMeshEntries: ***unsigned int***

Number of meshEntries in reac-diff system

estimatedDt: ***double***

Estimate of fastest (smallest) timescale in system.This is fallible because it depends on instantaneous concs,which of course change over the course of the simulation.

path: ***string***

Path of reaction system to take over

- **Source message field**

childMsg: ***int***

Message to child Elements

plugin: ***Id***

Sends out Stoich Id so that plugins can directly access fields and functions

nodeDiffBoundary: ***unsigned int,vector<unsigned int>,vector<double>***

Sends mol #s across boundary between nodes, to calculate diffusionterms. arg1 is originating node, arg2 is list of meshIndices forwhich data is being transferred, and arg3 are the 'n' values forall the pools on the specified meshIndices, to be plugged intothe appropriate place on the recipient node's S_ matrix

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

meshSplit: ***double,vector<double>,vector<unsigned int>,vector< vector<unsigned int> >,vector< vector<unsigned int> >***

Handles message from ChemMesh that defines how meshEntries are decomposed on this node, and how they communicate between nodes.Args: (oldVol, volumeVectorForAllEntries, localEntryList, outgoingDiffusion[node#][entry#], incomingDiffusion[node#][entry#])

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.79 SumFunc

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

result: ***double***

outcome of summation

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  output: ***double***

  Sends out sum on each timestep

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  input: ***double***

  Handles input values

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.80 Surface

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

volume: ***double***

This is something I'll need to write a function to compute.Perhaps have an update routine as it may be hard to compute but is needed often by the molecules.

- **Source message field**

childMsg: ***int***

Message to child Elements

absorb: ***void***

these help the system define non-standard operations for what a molecule does when it hits a surface.The default is reflect.As a molecule may interact with multiple surfaces, it isn't enough to confer a property on the molecule itself. We have to use messages. Perhaps we don't need these, but instead put entities on the surface which the molecule interacts with if it doesn't do the basic reflect operation.

transmit: ***void***

Surface lets molecules through

jump: ***void***

dunno

mixture: ***void***

dunno

surface: ***double,double,double***

Connects up to a compartment, either as interior or exterior Args are volume, area, perimeter

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.81 SymCompartment

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Vm: ***double***

membrane potential

Cm: ***double***

Membrane capacitance

Em: ***double***

Resting membrane potential

Im: ***double***

Current going through membrane

inject: ***double***

Current injection to deliver into compartment

initVm: ***double***

Initial value for membrane potential

Rm: ***double***

Membrane resistance

Ra: ***double***

Axial resistance of compartment

diameter: ***double***

Diameter of compartment

length: ***double***

Length of compartment

x0: ***double***

X coordinate of start of compartment

y0: ***double***

Y coordinate of start of compartment

z0: ***double***

Z coordinate of start of compartment

x: ***double***

x coordinate of end of compartment

y: ***double***

y coordinate of end of compartment

z: ***double***

z coordinate of end of compartment

- **Source message field**

childMsg: ***int***

Message to child Elements

VmOut: ***double***

Sends out Vm value of compartment on each timestep

axialOut: ***double***

Sends out Vm value of compartment to adjacent compartments,on each timestep

raxialOut: ***double,double***

Sends out Raxial information on each timestep, fields are Ra and Vm

raxialOut: ***double,double***

Sends out Ra and Vm on each timestep

sumRaxialOut: ***double***

Sends out Ra

requestSumAxial: ***void***

Sends out request for Ra.

raxialOut: ***double,double***

Sends out Ra and Vm on each timestep

sumRaxialOut: ***double***

Sends out Ra

requestSumAxial: ***void***

Sends out request for Ra.

Raxial2Out: ***double,double***

Sends out Ra and Vm

sumRaxial2Out: ***double***

Sends out Ra

requestSumAxial2: ***void***

Sends out request for Ra.

Raxial2Out: ***double,double***

Sends out Ra and Vm

sumRaxial2Out: ***double***

Sends out Ra

requestSumAxial2: ***void***

Sends out request for Ra.

Raxial2Out: ***double,double***

Sends out Ra and Vm

sumRaxial2Out: ***double***

Sends out Ra

requestSumAxial2: ***void***

Sends out request for Ra.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

randInject: ***double,double***

Sends a random injection current to the compartment. Must beupdated each timestep.Arguments to randInject are probability and current.

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

cable: ***void***

Message for organizing compartments into groups, calledcables. Doesn't do anything.

process: ***void***

Handles 'process' call

reinit: ***void***

Handles 'reinit' call

initProc: ***void***

Handles Process call for the 'init' phase of the Compartment calculations. These occur as a separate Tick cycle from the regular proc cycle, and should be called before the proc msg.

initReinit: ***void***

Handles Reinit call for the 'init' phase of the Compartment calculations.

handleChannel: ***double,double***

Handles conductance and Reversal potential arguments from Channel

handleRaxial: ***double,double***

Handles Raxial info: arguments are Ra and Vm.

handleAxial: ***double***

Handles Axial information. Argument is just Vm.

raxialSym: ***double,double***

Expects Ra and Vm from other compartment.

sumRaxial: ***double***

Expects Ra from other compartment.

handleSumRaxialRequest: ***void***

Handle request to send back Ra to originating compartment.

parentMsg: ***int***

Message from Parent Element(s)

- **Shared message field**

proc: ***void***

This is a shared message to receive Process messages from the scheduler objects. The Process should be called second in each clock tick, after the Init message.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

init: ***void***

This is a shared message to receive Init messages from the scheduler objects. Its job is to separate the compartmental calculations from the message passing. It doesn't really need to be shared, as it does not use the reinit part, but the scheduler objects expect this form of message for all scheduled output. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a dummy MsgDest for the Reinit operation. It also uses ProcInfo.

channel: ***void***

This is a shared message from a compartment to channels. The first entry is a MsgDest for the info coming from the channel. It expects Gk and Ek from the channel as args. The second entry is a MsgSrc sending Vm

axial: ***void***

This is a shared message between asymmetric compartments. axial messages (this kind) connect up to raxial messages (defined below). The soma should use raxial messages to connect to the axial message of all the immediately adjacent dendritic compartments.This puts the (low) somatic resistance in series with these dendrites. Dendrites should then use raxial messages toconnect on to more distal dendrites. In other words, raxial messages should face outward from the soma. The first entry is a MsgSrc sending Vm to the axialFuncof the target compartment. The second entry is a MsgDest for the info coming from the other compt. It expects Ra and Vm from the other compt as args. Note that the message is named after the source type.

raxial: ***void***

This is a raxial shared message between asymmetric compartments. The first entry is a MsgDest for the info coming from the other compt. It expects Vm from the other compt as an arg. The second is a MsgSrc sending Ra and Vm to the raxialFunc of the target compartment.

raxial1: ***void***

This is a raxial shared message between symmetric compartments.It goes from the tail of the current compartment to one closer to the soma.

CONNECTTAIL: ***void***

This is a raxial shared message between symmetric compartments.It is an alias for raxial1.

raxial2: ***void***

This is a raxial2 shared message between symmetric compartments.It goes from the head of the current compartment to a compartment further away from the soma

CONNECTHEAD: ***void***

This is a raxial2 shared message between symmetric compartments.It is an alias for raxial2.It goes from the current compartment to one further from the soma

CONNECTCROSS: ***void***

This is a raxial2 shared message between symmetric compartments.It is an alias for raxial2.Conceptually, this goes from the tail of the current compartment to the tail of a sibling compartment. However,this works out to the same as CONNECTHEAD in terms of equivalentcircuit.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.82 SynBase

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

numSynapses: ***unsigned int***

Number of synapses on SynBase

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.83 SynChan

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

numSynapses: ***unsigned int***

Number of synapses on SynBase

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

tau1: ***double***

Decay time constant for the synaptic conductance, tau1 >= tau2.

tau2: ***double***

Rise time constant for the synaptic conductance, tau1 >= tau2.

normalizeWeights: ***bool***

Flag. If true, the overall conductance is normalized by the number of individual synapses in this SynChan object.

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

activation: ***double***

Sometimes we want to continuously activate the channel

modulator: ***double***

Modulate channel response

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

proc: ***void***

Shared message to receive Process message from scheduler

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.84 SynChanBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

numSynapses: ***unsigned int***

Number of synapses on SynBase

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.85 Synapse

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

weight: ***double***

Synaptic weight

delay: ***double***

Axonal propagation delay to this synapse

- **Source message field**

childMsg: ***int***

Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  addSpike: ***double***

  Handles arriving spike messages, by redirecting up to parent SynBase object

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.86 Table

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

vec: ***vector<double>***

vector with all table entries

outputValue: ***double***

Output value holding current table entry or output of a calculation

size: ***unsigned int***

size of table. Note that this is the number of x divisions +1since it must represent the largest value as well as thesmallest

threshold: ***double***

threshold used when Table acts as a buffer for spikes

- **Source message field**

childMsg: ***int***

Message to child Elements

requestData: ***unsigned int***

Sends request for a field to target object

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

linearTransform: ***double,double***

Linearly scales and offsets data. Scale first, then offset.

xplot: ***string,string***

Dumps table contents to xplot-format file. Argument 1 is filename, argument 2 is plotname

plainPlot: ***string***

Dumps table contents to single-column ascii file. Uses scientific notation. Argument 1 is filename

loadCSV: ***string,int,int,char***

Reads a single column from a CSV file. Arguments: filename, column#, starting row#, separator

loadXplot: ***string,string***

Reads a single plot from an xplot file. Arguments: filename, plotnameWhen the file has 2 columns, the 2nd column is loaded.

loadXplotRange: ***string,string,unsigned int,unsigned int***

Reads a single plot from an xplot file, and selects a subset of points from it. Arguments: filename, plotname, startindex, endindexUses C convention: startindex included, endindex not included.When the file has 2 columns, the 2nd column is loaded.

compareXplot: ***string,string,string***

Reads a plot from an xplot file and compares with contents of TableBase.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: filename, plotname, comparison$_{\text{operationOperations}}$: rmsd (for

RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

compareVec: ***vector<double>,string***

Compares contents of TableBase with a vector of doubles.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: Other vector, $\text{comparison}_{\text{operationOperations}}$: rmsd (for RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

clearVec: ***void***

Handles request to clear the data vector

input: ***double***

Fills data into the Table.

spike: ***double***

Fills spike timings into the Table. Signal has to exceed thresh

recvData: ***bad***

Handles data sent back following request

process: ***void***

Handles process call, updates internal time stamp.

reinit: ***void***

Handles reinit call.

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  y: ***unsigned int,double***

  Value of table at specified index

1.87 TableBase

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

vec: ***vector<double>***

vector with all table entries

outputValue: ***double***

Output value holding current table entry or output of a calculation

size: ***unsigned int***

size of table. Note that this is the number of x divisions +1since it must represent the largest value as well as thesmallest

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  group: ***void***

  Handle for grouping. Doesn't do anything.

  linearTransform: ***double,double***

  Linearly scales and offsets data. Scale first, then offset.

  xplot: ***string,string***

  Dumps table contents to xplot-format file. Argument 1 is filename, argument 2 is plotname

  plainPlot: ***string***

  Dumps table contents to single-column ascii file. Uses scientific notation. Argument 1 is filename

  loadCSV: ***string,int,int,char***

Reads a single column from a CSV file. Arguments: filename, column#, starting row#, separator

loadXplot: ***string,string***

Reads a single plot from an xplot file. Arguments: filename, plotnameWhen the file has 2 columns, the 2nd column is loaded.

loadXplotRange: ***string,string,unsigned int,unsigned int***

Reads a single plot from an xplot file, and selects a subset of points from it. Arguments: filename, plotname, startindex, endindexUses C convention: startindex included, endindex not included.When the file has 2 columns, the 2nd column is loaded.

compareXplot: ***string,string,string***

Reads a plot from an xplot file and compares with contents of TableBase.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: filename, plotname, $\text{comparison}_{\text{operationOperations}}$: rmsd (for RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

compareVec: ***vector<double>,string***

Compares contents of TableBase with a vector of doubles.Result is put in 'output' field of table.If the comparison fails (e.g., due to zero entries), the return value is -1.Arguments: Other vector, $\text{comparison}_{\text{operationOperations}}$: rmsd (for RMSDifference), rmsr (RMSratio), dotp (Dot product, not yet implemented).

clearVec: ***void***

Handles request to clear the data vector

- **Shared message field**
- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

y: ***unsigned int,double***

Value of table at specified index

1.88 TableEntry

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

value: ***double***

Data value in this entry

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.89 Tick

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

dt: ***double***

Timestep for this tick

localdt: ***double***

Timestep for this tick

- **Source message field**

childMsg: ***int***

Message to child Elements

process0: ***PK8ProcInfo***

Process for Tick 0

reinit0: ***PK8ProcInfo***

Reinit for Tick 0

process1: ***PK8ProcInfo***

Process for Tick 1

reinit1: ***PK8ProcInfo***

Reinit for Tick 1

process2: ***PK8ProcInfo***

Process for Tick 2

reinit2: ***PK8ProcInfo***

Reinit for Tick 2

process3: ***PK8ProcInfo***

Process for Tick 3

reinit3: ***PK8ProcInfo***

Reinit for Tick 3

process4: ***PK8ProcInfo***

Process for Tick 4

reinit4: ***PK8ProcInfo***

Reinit for Tick 4

process5: ***PK8ProcInfo***

Process for Tick 5

reinit5: ***PK8ProcInfo***

Reinit for Tick 5

process6: ***PK8ProcInfo***

Process for Tick 6

reinit6: ***PK8ProcInfo***

Reinit for Tick 6

process7: ***PK8ProcInfo***

Process for Tick 7

reinit7: ***PK8ProcInfo***

Reinit for Tick 7

process8: ***PK8ProcInfo***

Process for Tick 8

reinit8: ***PK8ProcInfo***

Reinit for Tick 8

process9: ***PK8ProcInfo***

Process for Tick 9

reinit9: ***PK8ProcInfo***

Reinit for Tick 9

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**

  proc0: ***void***

  Shared proc/reinit message

  proc1: ***void***

  Shared proc/reinit message

  proc2: ***void***

  Shared proc/reinit message

  proc3: ***void***

  Shared proc/reinit message

proc4: ***void***

Shared proc/reinit message

proc5: ***void***

Shared proc/reinit message

proc6: ***void***

Shared proc/reinit message

proc7: ***void***

Shared proc/reinit message

proc8: ***void***

Shared proc/reinit message

proc9: ***void***

Shared proc/reinit message

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.90 TriPanel

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

nPts: ***unsigned int***

Number of points used by panel to specify geometry

nDims: ***unsigned int***

Number of Dimensions used by panel to specify geometry

numNeighbors: ***unsigned int***

Number of Neighbors of panel

shapeId: ***unsigned int***

Identifier for shape type, as used by Smoldyn

coords: ***vector<double>***

All the coordinates for the panel. X vector, then Y, then ZZ can be left out for 2-D panels.Z and Y can be left out for 1-D panels.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toNeighbor: ***void***

  Identifies neighbors of the current panel

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  neighbor: ***void***

  Handles incoming message from neighbor

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  x: ***unsigned int,double***

  x coordinate identified by index

  y: ***unsigned int,double***

  y coordinate identified by index

  z: ***unsigned int,double***

  z coordinate identified by index

1.91 VectorTable

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

  localNumField: ***unsigned int***

  For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

xdivs: ***unsigned int***

Number of divisions.

xmin: ***double***

Minimum value in table.

xmax: ***double***

Maximum value in table.

invdx: ***double***

Maximum value in table.

table: ***vector<double>***

The lookup table.

- **Source message field**

  childMsg: ***int***

  Message to child Elements

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

- **Shared message field**
- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

  lookupvalue: ***double,double***

Lookup function that performs interpolation to return a value.

lookupindex: ***unsigned int,double***

Lookup function that returns value by index.

1.92 ZombieBufPool

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.93 ZombieCaConc

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes

field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Ca: ***double***

Calcium concentration.

CaBasal: ***double***

Basal Calcium concentration.

Ca_{base}: ***double***

Basal Calcium concentration, synonym for CaBasal

tau: ***double***

Settling time for Ca concentration

B: ***double***

Volume scaling factor

thick: ***double***

Thickness of Ca shell.

ceiling: ***double***

Ceiling value for Ca concentration. If Ca > ceiling, Ca = ceiling. If ceiling <= 0.0, there is no upper limit on Ca concentration value.

floor: ***double***

Floor value for Ca concentration. If Ca < floor, Ca = floor

- **Source message field**

childMsg: ***int***

Message to child Elements

concOut: ***double***

Concentration of Ca in pool

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

current: ***double***

Calcium Ion current, due to be converted to conc.

currentFraction: ***double,double***

Fraction of total Ion current, that is carried by Ca2+.

increase: ***double***

Any input current that increases the concentration.

decrease: ***double***

Any input current that decreases the concentration.

basal: ***double***

Synonym for assignment of basal conc.

- **Shared message field**

proc: ***void***

Shared message to receive Process message from scheduler

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.94 ZombieCompartment

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the

actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Vm: ***double***

membrane potential

Cm: ***double***

Membrane capacitance

Em: ***double***

Resting membrane potential

Im: ***double***

Current going through membrane

inject: ***double***

Current injection to deliver into compartment

initVm: ***double***

Initial value for membrane potential

Rm: ***double***

Membrane resistance

Ra: ***double***

Axial resistance of compartment

diameter: ***double***

Diameter of compartment

length: ***double***

Length of compartment

x0: ***double***

X coordinate of start of compartment

y0: ***double***

Y coordinate of start of compartment

z0: ***double***

Z coordinate of start of compartment

x: ***double***

x coordinate of end of compartment

y: ***double***

y coordinate of end of compartment

z: ***double***

z coordinate of end of compartment

- **Source message field**

childMsg: ***int***

Message to child Elements

VmOut: ***double***

Sends out Vm value of compartment on each timestep

axialOut: ***double***

Sends out Vm value of compartment to adjacent compartments,on each timestep

raxialOut: ***double,double***

Sends out Raxial information on each timestep, fields are Ra and Vm

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

randInject: ***double,double***

Sends a random injection current to the compartment. Must beupdated each timestep.Arguments to randInject are probability and current.

injectMsg: ***double***

The injectMsg corresponds to the INJECT message in the GENESIS compartment. Unlike the 'inject' field, any value assigned by handleInject applies only for a single timestep.So it needs to be updated every dt for a steady (or varying)injection current

cable: ***void***

Message for organizing compartments into groups, calledcables. Doesn't do anything.

process: ***void***

Handles 'process' call

reinit: ***void***

Handles 'reinit' call

initProc: ***void***

Handles Process call for the 'init' phase of the Compartment calculations. These occur as a separate Tick cycle from the regular proc cycle, and should be called before the proc msg.

initReinit: ***void***

Handles Reinit call for the 'init' phase of the Compartment calculations.

handleChannel: ***double,double***

Handles conductance and Reversal potential arguments from Channel

handleRaxial: ***double,double***

Handles Raxial info: arguments are Ra and Vm.

handleAxial: ***double***

Handles Axial information. Argument is just Vm.

- **Shared message field**

proc: ***void***

This is a shared message to receive Process messages from the scheduler objects. The Process should be called second in each clock tick, after the Init message.The first entry in the shared msg is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

init: ***void***

This is a shared message to receive Init messages from the scheduler objects. Its job is to separate the compartmental calculations from the message passing. It doesn't really need to be shared, as it does not use the reinit part, but the scheduler objects expect this form of message for all scheduled output. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt and so on. The second entry is a dummy MsgDest for the Reinit operation. It also uses ProcInfo.

channel: ***void***

This is a shared message from a compartment to channels. The first entry is a MsgDest for the info coming from the channel. It expects Gk and Ek from the channel as args. The second entry is a MsgSrc sending Vm

axial: ***void***

This is a shared message between asymmetric compartments. axial messages (this kind) connect up to raxial messages (defined below). The soma should use raxial messages to connect to the axial message of all the immediately adjacent dendritic compartments.This puts the (low) somatic resistance in series with these dendrites. Dendrites should then use raxial messages toconnect on to more distal dendrites. In other words, raxial messages should face outward from the soma. The first entry is a MsgSrc sending Vm to the axialFuncof the target compartment. The second entry is a MsgDest for the info coming from the other compt. It expects Ra and Vm from the other compt as args. Note that the message is named after the source type.

raxial: ***void***

This is a raxial shared message between asymmetric compartments. The first entry is a

MsgDest for the info coming from the other compt. It expects Vm from the other compt as an arg. The second is a MsgSrc sending Ra and Vm to the raxialFunc of the target compartment.

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.95 ZombieEnz

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field

dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

k1: ***double***

Forward reaction from enz + sub to complex

k2: ***double***

Reverse reaction from complex to enz + sub

k3: ***double***

Forward rate constant from complex to product + enz

ratio: ***double***

Ratio of k2/k3

concK1: ***double***

K1 expressed in concentration (1/millimolar.sec) units

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

toEnz: ***double,double***

Sends out increment of molecules on product each timestep

toCplx: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

enzDest: ***double***

Handles # of molecules of Enzyme

subDest: ***double***

Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product. Dummy.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the MMEnz to recompute its numKm after remeshing

enzDest: ***double***

Handles # of molecules of Enzyme

cplxDest: ***double***

Handles # of molecules of enz-sub complex

- **Shared message field**

sub: ***void***

Connects to substrate molecule

prd: ***void***

Connects to product molecule

proc: ***void***

Shared message for process and reinit

enz: ***void***

Connects to enzyme pool

cplx: ***void***

Connects to enz-sub complex pool

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.96 ZombieFuncPool

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

input: ***double***

Handles input to control value of n_

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.97 ZombieHHChannel

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

  # of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

  objectDimensions: ***vector<unsigned int>***

  Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

  lastDimension: ***unsigned int***

  Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Gbar: ***double***

Maximal channel conductance

Ek: ***double***

Reversal potential of channel

Gk: ***double***

Channel conductance variable

Ik: ***double***

Channel current variable

Xpower: ***double***

Power for X gate

Ypower: ***double***

Power for Y gate

Zpower: ***double***

Power for Z gate

instant: ***int***

Bitmapped flag: bit 0 = Xgate, bit 1 = Ygate, bit 2 = ZgateWhen true, specifies that the lookup table value should beused directly as the state of the channel, rather than usedas a rate term for numerical integration for the state

X: ***double***

State variable for X gate

Y: ***double***

State variable for Y gate

Z: ***double***

State variable for Y gate

useConcentration: ***int***

Flag: when true, use concentration message rather than Vm tocontrol Z gate

- **Source message field**

childMsg: ***int***

Message to child Elements

channelOut: ***double,double***

Sends channel variables Gk and Ek to compartment

permeability: ***double***

Conductance term going out to GHK object

IkOut: ***double***

Channel current. This message typically goes to concenobjects that keep track of ion concentration.

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

Vm: ***double***

Handles Vm message coming in from compartment

Vm: ***double***

Handles Vm message coming in from compartment

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

concen: ***double***

Incoming message from Concen object to specific conc to usein the Z gate calculations

createGate: ***string***

Function to create specified gate.Argument: Gate type [X Y Z]

- **Shared message field**

channel: ***void***

This is a shared message to couple channel to compartment. The first entry is a MsgSrc to send Gk and Ek to the compartment The second entry is a MsgDest for Vm from the compartment.

ghk: ***void***

Message to Goldman-Hodgkin-Katz object

proc: ***void***

This is a shared message to receive Process message from thescheduler. The first entry is a MsgDest for the Process operation. It has a single argument, ProcInfo, which holds lots of information about current time, thread, dt andso on. The second entry is a MsgDest for the Reinit operation. It also uses ProcInfo.

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.98 ZombieMMenz

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

Km: ***double***

Michaelis-Menten constant in SI conc units (milliMolar)

numKm: ***double***

Michaelis-Menten constant in number units, volume dependent

kcat: ***double***

Forward rate constant for enzyme, units 1/sec

numSubstrates: ***unsigned int***

Number of substrates in this MM reaction. Usually 1.Does not include the enzyme itself

- **Source message field**

childMsg: ***int***

Message to child Elements

toSub: ***double,double***

Sends out increment of molecules on product each timestep

toPrd: ***double,double***

Sends out increment of molecules on product each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

enzDest: ***double***

Handles # of molecules of Enzyme

subDest: ***double***

Handles # of molecules of substrate

prdDest: ***double***

Handles # of molecules of product. Dummy.

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

remesh: ***void***

Tells the MMEnz to recompute its numKm after remeshing

- **Shared message field**

  sub: ***void***

  Connects to substrate molecule

  prd: ***void***

  Connects to product molecule

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.99 ZombiePool

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

  linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

n: ***double***

Number of molecules in pool

nInit: ***double***

Initial value of number of molecules in pool

diffConst: ***double***

Diffusion constant of molecule

conc: ***double***

Concentration of molecules in this pool

concInit: ***double***

Initial value of molecular concentration in pool

size: ***double***

Size of compartment. Units are SI. Utility field, the actual size info is stored on a volume mesh entry in the parent compartment.This is hooked up by a message. If the message isn'tavailable size is just taken as 1

speciesId: ***unsigned int***

Species identifier for this mol pool. Eventually link to ontology.

- **Source message field**

childMsg: ***int***

Message to child Elements

nOut: ***double***

Sends out # of molecules in pool on each timestep

requestMolWt: ***void***

Requests Species object for mol wt

requestSize: ***double***

Requests Size of pool from matching mesh entry

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

group: ***void***

Handle for grouping. Doesn't do anything.

reacDest: ***double,double***

Handles reaction input

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

handleMolWt: ***double***

Separate finfo to assign molWt, and consequently diffusion const.Should only be used in

SharedMsg with species.

remesh: ***double,unsigned int,unsigned int,vector<unsigned int>,vector<double>***

Handle commands to remesh the pool. This may involve changing the number of pool entries, as well as changing their volumes

- **Shared message field**

reac: ***void***

Connects to reaction

proc: ***void***

Shared message for process and reinit

species: ***void***

Shared message for connecting to species objects

mesh: ***void***

Shared message for dealing with mesh operations

- **Lookup field**

neighbours: ***string,vector<Id>***

Ids of Elements connected this Element on specified field.

1.100 ZombieReac

- **Value field**

this: ***Neutral***

Access function for entire object

name: ***string***

Name of object

me: ***ObjId***

ObjId for current object

parent: ***ObjId***

Parent ObjId for current object

children: ***vector<Id>***

vector of ObjIds listing all children of current object

path: ***string***

text path for object

class: ***string***

Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

kf: ***double***

Forward rate constant, in # units

kb: ***double***

Reverse rate constant, in # units

Kf: ***double***

Forward rate constant, in concentration units

Kb: ***double***

Reverse rate constant, in concentration units

numSubstrates: ***unsigned int***

Number of substrates of reaction

numProducts: ***unsigned int***

Number of products of reaction

- **Source message field**

  childMsg: ***int***

  Message to child Elements

  toSub: ***double,double***

  Sends out increment of molecules on product each timestep

  toPrd: ***double,double***

  Sends out increment of molecules on product each timestep

- **Destination message field**

  parentMsg: ***int***

  Message from Parent Element(s)

  subDest: ***double***

  Handles # of molecules of substrate

  prdDest: ***double***

  Handles # of molecules of product

  process: ***void***

  Handles process call

  reinit: ***void***

  Handles reinit call

  remesh: ***void***

Tells the reac to recompute its numRates, as remeshing has happened

- **Shared message field**

  sub: ***void***

  Connects to substrate pool

  prd: ***void***

  Connects to substrate pool

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.101 ZombieSumFunc

- **Value field**

  this: ***Neutral***

  Access function for entire object

  name: ***string***

  Name of object

  me: ***ObjId***

  ObjId for current object

  parent: ***ObjId***

  Parent ObjId for current object

  children: ***vector<Id>***

  vector of ObjIds listing all children of current object

  path: ***string***

  text path for object

  class: ***string***

  Class Name of object

linearSize: ***unsigned int***

# of entries on Element: product of all dimensions.Note that on a FieldElement this includes field entries.If field entries form a ragged array, then the linearSize may begreater than the actual number of allocated entries, since thelastDimension is at least as big as the largest ragged array.

objectDimensions: ***vector<unsigned int>***

Array Dimensions of object on the Element.This includes the lastDimension (field dimension) if present.

lastDimension: ***unsigned int***

Max size of the last dimension of the object.In the case of regular objects, resizing this value resizesthe last dimensionIn the case of ragged arrays (such as synapses), resizing thisvalue resizes the upper limit of the last dimension,but cannot make it smaller than the biggest ragged array size.Normally is only assigned from Shell::doSyncDataHandler.

localNumField: ***unsigned int***

For a FieldElement: number of entries of self on current nodeFor a regular Element: zero.

pathIndices: ***vector< vector<unsigned int> >***

Indices of the entire path hierarchy leading up to this Object.

msgOut: ***vector<ObjId>***

Messages going out from this Element

msgIn: ***vector<ObjId>***

Messages coming in to this Element

result: ***double***

outcome of summation

- **Source message field**

childMsg: ***int***

Message to child Elements

output: ***double***

Sends out sum on each timestep

- **Destination message field**

parentMsg: ***int***

Message from Parent Element(s)

input: ***double***

Handles input values

process: ***void***

Handles process call

reinit: ***void***

Handles reinit call

- **Shared message field**

  proc: ***void***

  Shared message for process and reinit

- **Lookup field**

  neighbours: ***string,vector<Id>***

  Ids of Elements connected this Element on specified field.

1.102 testSched

- **Value field**
- **Source message field**
- **Destination message field**

  process: ***void***

  handles process call

- **Shared message field**
- **Lookup field**

2 MOOSE Functions

2.1 ce

Set the current working element. 'ce' is an alias of this function

2.2 connect

connect(src, src_{field}, dest, $dest_{field}$, $message_{type}$) -> bool

Create a message between `$src_{field}$` on `src` object to `$dest_{field}$` on `dest` object.

Parameters

---

src : element the source object src_{field} : str the source field name. Fields listed under `srcFinfo` and `sharedFinfo` qualify for this. dest : element the destination object. $dest_{field}$: str the destination field name. Fields listed under `destFinfo` and `sharedFinfo` qualify for this. $message_{type}$: str (optional) Type of the message. Can be `Single`, `OneToOne`, `OneToAll`. If not specified, it defaults to `Single`.

Returns

---

element of the message-manager for the newly created message.

Example

Connect the output of a pulse generator to the input of a spike generator:

>>> pulsegen = moose.PulseGen('pulsegen') >>> spikegen = moose.SpikeGen('spikegen') >>> moose.connect(pulsegen, 'outputOut', spikegen, 'Vm') 1

2.3 copy

copy(src, dest, name, n, toGlobal, copyExtMsg) -> bool Make copies of a moose object. Parameters

---

src : ematrix, element or str source object. dest : ematrix, element or str Destination object to copy into. name : str Name of the new object. If omitted, name of the original will be used. n : int Number of copies to make. toGlobal: int Relevant for parallel environments only. If false, the copies will reside on local node, otherwise all nodes get the copies. copyExtMsg: int If true, messages to/from external objects are also copied.

Returns

---

ematrix of the copied object

2.4 delete

moose.delete(id)

Delete the underlying moose object. This does not delete any of the Python objects referring to this ematrix but does invalidate them. Any attempt to access them will raise a ValueError.

Parameters

---

id : ematrix ematrix of the object to be deleted.

2.5 element

moose.element(arg) -> moose object

Convert a path or an object to the appropriate builtin moose class instance Parameters

---

arg: str or ematrix or moose object path of the moose element to be converted or another element (possibly available as a superclass instance).

Returns An element of the moose builtin class the specified object belongs to.

2.6 exists

True if there is an object with specified path.

2.7 getCwe

Get the current working element. 'pwe' is an alias of this function.

2.8 getField

getField(element, field, fieldtype) – Get specified field of specified type from object ematrix.

2.9 getFieldDict

getFieldDict(className, finfoType) -> dict

Get dictionary of field names and types for specified class. Parameters

---

className : str MOOSE class to find the fields of. finfoType : str (optional) Finfo type of the fields to find. If empty or not specified, all fields will be retrieved. note: This behaviour is different from `getFieldNames` where only `valueFinfo`s are returned when `finfoType` remains unspecified.

Example

List all the source fields on class Neutral: >>> moose.getFieldDict('Neutral', 'srcFinfo') {'childMsg': 'int'}

2.10 getFieldNames

getFieldNames(className, finfoType='valueFinfo') -> tuple

Get a tuple containing the name of all the fields of `finfoType` kind.

Parameters

---

className : string Name of the class to look up. finfoType : string The kind of field (`valueFinfo`, `srcFinfo`, `destFinfo`, `lookupFinfo`, `fieldElementFinfo`.).

2.11 isRunning

True if the simulation is currently running.

2.12 loadModel

loadModel(filename, modelpath, solverclass) -> moose.ematrix

Load model from a file to a specified path.

Parameters

---

filename : str model description file. modelpath : str moose path for the top level element of the model to be created. solverclass : str (optional) solver type to be used for simulating the model.

Returns

---

ematrix instance refering to the loaded model container.

2.13 move

Move a ematrix object to a destination.

2.14 quit

Finalize MOOSE threads and quit MOOSE. This is made available for debugging purpose only. It will automatically get called when moose module is unloaded. End user should not use this function.

2.15 reinit

reinit() -> None

Reinitialize simulation.

This function (re)initializes moose simulation. It must be called before you start the simulation (see moose.start). If you want to continue simulation after you have called moose.reinit() and moose.start(), you must NOT call moose.reinit() again. Calling moose.reinit() again will take the system back to initial setting (like clear out all data recording tables, set state variables to their initial values, etc.

2.16 saveModel

saveModel(source, fileame)

Save model rooted at `source` to file `filename`.

Parameters

---

source: ematrix or element or str root of the model tree

filename: str destination file to save the model in.

Returns

None

2.17 seed

moose.seed(seedvalue) -> None

Reseed MOOSE random number generator.

Parameters

---

seed: int Optional value to use for seeding. If 0, a random seed is automatically created using the current system time and other information. If not specified, it defaults to 0.

2.18 setClock

Set the dt of a clock.

2.19 setCwe

Set the current working element. 'ce' is an alias of this function

2.20 start

start(t) -> None

Run simulation for `t` time. Advances the simulator clock by `t` time.

After setting up a simulation, YOU MUST CALL MOOSE.REINIT() before CALLING MOOSE.START() TO EXECUTE THE SIMULATION. Otherwise, the simulator behaviour will be undefined. Once moose.reinit() has been called, you can call moose.start(t) as many time as you like. This will continue the simulation from the last state for `t` time.

Parameters

---

t : float duration of simulation.

Returns

None

See also

---

moose.reinit : (Re)initialize simulation

2.21 stop

Stop simulation

2.22 useClock

Schedule objects on a specified clock

2.23 wildcardFind

moose.wildcardFind(expression) -> tuple of ematrices.

Find an object by wildcard.

Parameters

---

expression: str MOOSE allows wildcard expressions of the form {PATH}/{WILDCARD}[{CONDITION}] where {PATH} is valid path in the element tree. {WILDCARD} can be `#` or `##`. `#` causes the search to be restricted to the children of the element specified by {PATH}. `##` makes the search to recursively go through all the descendants of the {PATH} element. {CONDITION} can be TYPE={CLASSNAME} : an element satisfies this condition if it is of class {CLASSNAME}. ISA={CLASSNAME} : alias for TYPE={CLASSNAME} CLASS={CLASSNAME} : alias for TYPE={CLASSNAME} FIELD({FIELDNAME}){OPERATOR}{VALUE} : compare field {FIELDNAME} with {VALUE} by {OPERATOR} where {OPERATOR} is a comparison operator (=, !=, >, <, >=, <=). For example, /mymodel/##[FIELD(Vm)>=-65] will return a list of all the objects under /mymodel whose Vm field is >= -65.

2.24 writeSBML

Export biochemical model to an SBML file.

2.25 doc

Display the documentation for class or field in a class.

Parameters

---

arg: str or moose class or instance of melement or instance of ematrix

argument can be a string specifying a moose class name and a field name separated by a dot. e.g., 'Neutral.name'. Prepending `moose.` is allowed. Thus moose.doc('moose.Neutral.name') is equivalent to the above.

argument can also be string specifying just a moose class name or a moose class or a moose object

(instance of melement or ematrix or there subclasses). In that case, the builtin documentation for the corresponding moose class is displayed.

paged: bool

Whether to display the docs via builtin pager or print and exit. If not specified, it defaults to False and moose.doc(xyz) will print help on xyz and return control to command line.

2.26 getfielddoc

Get the documentation for field specified by tokens.

tokens should be a two element list/tuple where first element is a MOOSE class name and second is the field name.

2.27 getmoosedoc

Retrieve MOOSE builtin documentation for tokens.

tokens is a list or tuple containing: (classname, [fieldname])

2.28 le

List elements.

Parameters

---

el: str/melement/ematrix/None The element or the path under which to look. If `None`, children of current working element are displayed.

2.29 listmsg

Return a list containing the incoming and outgoing messages of the given object.

2.30 pwe

Print present working element. Convenience function for GENESIS users.

2.31 showfield

Show the fields of the element, their data types and values in human readable format. Convenience function for GENESIS users.

Parameters:

elem: str/melement instance Element or path of an existing element.

field: str Field to be displayed. If '*', all fields are displayed.

showtype: bool If True show the data type of each field.

2.32 showfields

Convenience function. Should be deprecated if nobody uses it.

2.33 showmsg

Prints the incoming and outgoing messages of the given object.

2.34 syncDataHandler

Synchronize data handlers for target.

Parameter: target – target element or path or ematrix.

Date: 2012-10-04 20:12:08 IST

Author: Automatically extracted on 2012-10-04T20:11:55.291611