

SBML Level 3 Array Features Proposal

Andrew Finney¹, Victoria Gor²,
Ben Bornstein², Eric Mjolsness³

¹ University of Hertfordshire

² Jet Propulsion Laboratory

³ University of California, Irvine

Overview

- Motivation
- Integer and constant Math
- Arrays of species, compartments, reactions and parameters
 - _ Declarations
 - _ Dimension structures
- Object References
- Arrays in MathML
- Sparse Arrays
- Other features of Arrays proposal

Motivation

- Model phenomena with repeated structures:
 - _ logical connectivity in tissues and simple organisms where the detailed geometry is not significant.
 - the community effect in developmental gene regulation (Gurdon 1988)
 - _ development in populations of cells
 - Eric Mjolsness
 - _ model interaction of species with polymers/sequences
 - unpublished work in Bolouri laboratory @ ISB
 - _ Simple Stochastic models of RNA transcription

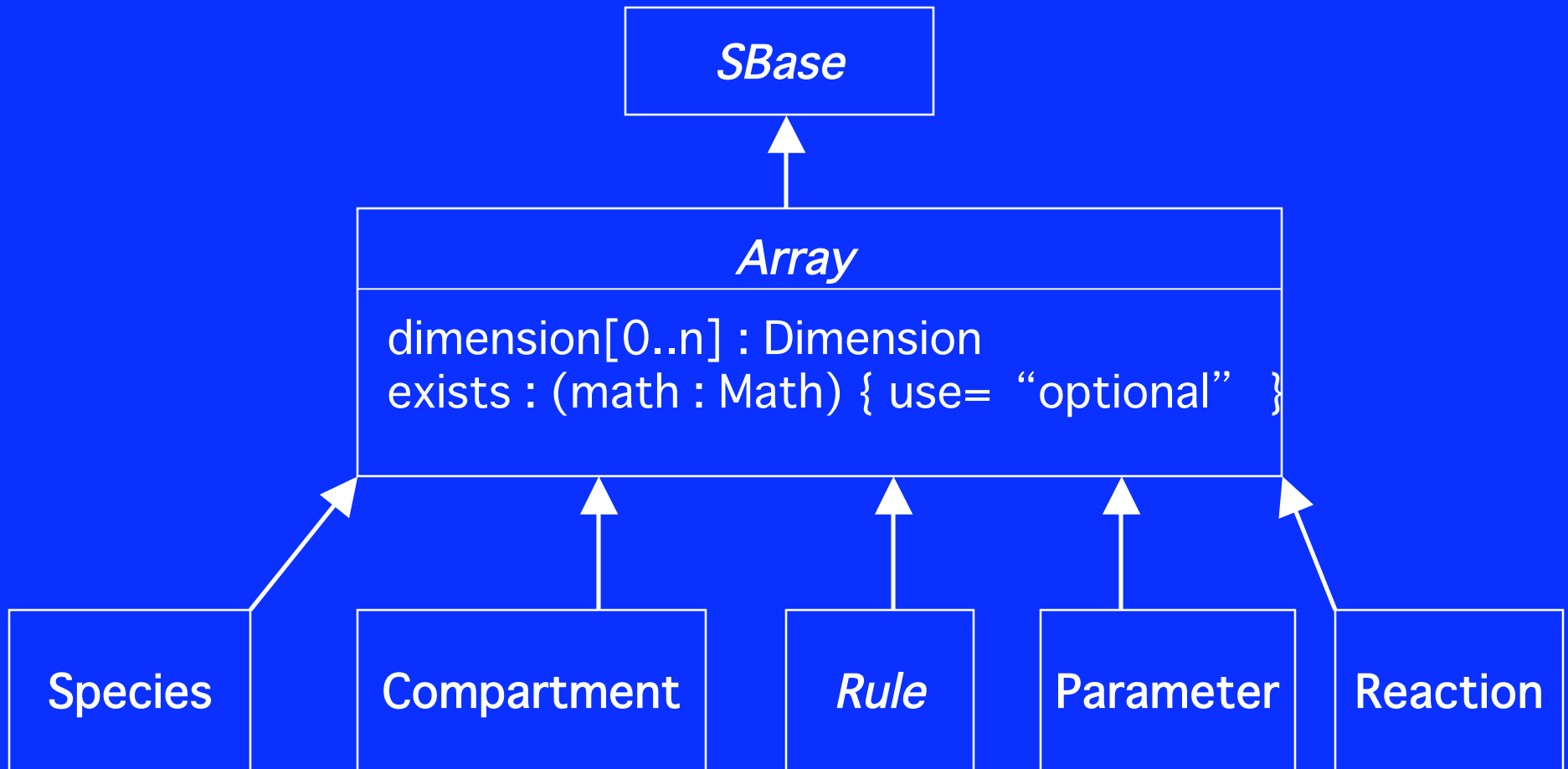
Integer Math

- **Arrays concept requires integer math concept**
 - _ Some expressions need to result in an integer value
 - e.g. indexing an array
 - _ need to check expression equality
- **Assume expression value is integer if and only if it is**
 - _ An Integer Parameter or Dimension Symbol (see later)
 - _ a number without decimal component
 - _ the result of `ceiling` or `floor` operators
 - _ the result of other operators that will return an integer given integer arguments
 - need to define this set (?)
- **Essentially assume simple type system**

Constant Math

- **Arrays need constant math**
 - _ arrays are meant to define structures that remain normally unchanged during simulation
- **An expression is constant if and only if it is**
 - _ a symbol declared constant
 - includes Integer Parameters and Dimension Symbols (see later)
 - _ a number
 - _ the result of an operator given constant arguments

Arrays



Arrays: Dimension structures

Dimension
<pre>id : SId name : String { use = "optional" } lowerLimit : (math : Math) upperLimit : (math : Math)</pre>

- limit expressions are constant integer expressions
- limits are inclusive
- declares `id` as integer symbol

Arrays: Example

Fragment declaring an array of 10 compartments:

```
<compartment id="cell" volume="1">
  <listOfDimensions>
    <dimension id="x">
      <lowerLimit>
        <m:math>
          <m:cn>0</m:cn>
        </m:math>
      </lowerLimit>
      <upperLimit>
        <m:math>
          <m:cn>9</m:cn>
        </m:math>
      </upperLimit>
    </dimension>
  </listOfDimensions>
</compartment>
```

Object References

ObjectRef

```
object : SId  
index : (math : Math)[0..n]  
...
```

Species

```
compartment : SId { use= "optional" }  
compartmentLink : ObjectRef { use = "optional" }  
...
```

SimpleSpeciesReference

```
species : SId { use= "optional" }  
speciesLink : ObjectRef { use = "optional" }
```

index is list of constant integer expressions

Object References: Example

placing a species in a specific cell:

```
<species id="s" initialAmount="1">  
  <compartmentLink object="cell">  
    <listOfIndices>  
      <m:math>  
        <m:cn>1</m:cn>  
      </m:math>  
    </listOfIndices>  
  </compartmentLink>  
</species>
```

Arrays in MathML

- **Map Arrays to Vectors and Matrices in MathML**
 - _ matrices are 2 dimensional in MathML
 - _ assume semantics extend to n dimensions (?)
 - _ introduce new operators where required
- **Use the `selector` operator to access elements of an array/matrix**
 - _ first argument is array/matrix
 - _ remaining arguments are integer
 - and constant (?)
 - _ need SBML built-in operator for 3+ dimensions
 - use this exclusively?

Example of selector operator

The following XML is equivalent to

A[i]

```
<m:math>  
  <m:apply>  
    <m:selector/>  
    <m:ci>A</m:ci>  
    <m:ci>i</m:ci>  
  </m:apply>  
</m:math>
```

Integer Parameters

Model

```
IntegerParameter[0..n] : IntegerParameter  
...
```

IntegerParameter

```
id : SId  
name : string {use = optional}  
value : integer
```

- declares `id` as integer symbol
- constant
 - no rules can be applied to symbol

Conditional Structures

- **exists** field
 - _ On Species, Rules, Compartments, Reactions and Parameters
 - _ Boolean field
 - _ defines whether the object is actually part of the model
- **Model is dynamic if exists is not constant**
 - _ some models need to be dynamic
- **enables definition of sparse arrays**
- **sparse arrays useful way to represent connections between elements of another array**

Other features of Array Proposal

- Rules for Initial Conditions
 - _ enables expressions to be used for initial conditions
 - _ well defined semantics
- Syntactic Sugar simplifies models
 - _ verbosity reduction
 - strictly redundant
- Result of Rule can be an array/matrix
- Extending the set of MathML operators
 - _ *constructors* `matrix`, `matrixrow`, `vector`
 - _ *element reference operator* `selector`
 - _ *linear algebra operators* `vectorproduct`, `scalarproduct`, `outerproduct`, `transpose`
 - _ *sum product operators* `sum`, `product`
 - _ *quantifier operators* `forall`, `exists`
 - _ *qualifier components* `bvar`, `lowlimit`, `uplimit`, `interval`, `condition`

Summary

- Proposal allows construction of n-dimensional arrays of components
- Can refer to array elements to link components
- Maps to MathML
- Object Reference mechanism shared with model composition
- Features independent of arrays but complimentary
 - `exists`
 - initial condition rules