The Distributions Package for SBML Level 3

Authors

Stuart L Moodie
moodie@ebi.ac.uk
EMBL-EBI
Hinxton, UK

Darren Wilkinson
darren.wilkinson@ncl.ac.uk
University of Newcastle
Newcastle, UK

Nicolas Le Novère
lenov@ebi.ac.uk
EMBL-EBI
Hinxton, UK

Contributors

Colin Gillespie
c.gillespie@ncl.ac.uk
University of Newcastle
Newcastle, UK

19 July, 2012

Version 0.6 (Draft)

Disclaimer: This is a working draft of the SBML Level 3 "distib" package proposal. It is not a normative document. Please send comments and other feedback to the mailing list: sbml-distrib@lists.sourceforge.net.

Contents

Revision History

Version	Date	Author	Comments
0.1 (Draft)	15 Oct 2011	Stuart Moodie	First draft
0.2 (Draft)	16 Oct 2011	Stuart Moodie	Added introductory text and background info. Other minor changes etc.
0.3 (Draft)	16 Oct 2011	Stuart Moodie	Filled empty invocation semantics section.
0.4 (Draft)	4 Jan 2012	Stuart Moodie	Incorporated comments from NlN, MS and SK. Some minor revisions and corrections.
0.5 (Draft)	6 Jan 2012	Stuart Moodie	Incorporated addition comments on aim of package from NlN.
0.6 (Draft)	19 Jul 2012	Stuart Moodie	Incorporated revisions discussed and agreed at HARMONY 2012.

1 Introduction and motivation

1.1 What is it?

The Distributions package (also affectionately known as distrib for short) provides an extension to SBML Level 3 that enables it to encode models that sample vales from statistical distributions. Applications of the package include for instance descriptions population based models: an important subset of which are pharmacokinetic/pharmacodynamic (PK/PD) models[1], which are used to model the action of drugs.

Note that originally the package was called Distributions and Ranges, but Ranges are no longer in the scope of this hence the name change.

1.2 Scope

The Distributions package provides a mechanism for sampling from a probability distribution for use in parts of an SBML model that are **not** continuously evaluated. in particular the following are in scope:

- Sampling from a continuous distribution.
- Sampling from a discrete distribution.
- Sampling from user-defined probability density function.
- Sampling from user-defined discrete probability density.

At one point the following were considered for inclusion in this package but are now **out of scope**:

- Stochastic differential equations.
- Cumulative distribution functions.
- Description of the uncertainty of a model parameter: for example, standard deviation, standard error and similar descriptive statistics.[2]

1.3 This Document

The authors' expectation is that this is close to a final draft of the proposal with only a limited number of issues to be resolved. In its current state this document aims to establish a consensus about what has been agreed and what work remains to be carried out to complete the definition of this package.

This proposal is a working document and once finalised will be the first step towards the formal adoption of the distrib as a package in SBML Level 3. After this two implementations based on this proposal are required and then a vote on its adoption by the SBML community. The proposal will then provide the basis for a future package specification document. More details of the SBML package adoption process can be found at: http://sbml.org/Documents/SBML_Development_Process.

1.4 Conventions used in this document

As we are early in the package proposal process there will be some parts of this proposal where there is no clear consensus on the correct solution or only recent agreement or agreement by a group which may not be representative of the SBML community as a whole. These cases are indicated by the question mark in the left margin (illustrated). The reader should pay particular attention to these points and ideally provide feedback, especially if they disagree with what is proposed. Similarly there will be points — especially as the proposal is consolidated — which are agreed, but which the reader should take note of and perhaps read again. These points are emphasised by the hand pointer in the left margin (illustrated),

[1] for more information see: http://www.pharmpk.com/.

[2] It is proposed that this be provided descriptive statistics in a separate package.

2 Background

2.1 Problems with current SBML approaches

SBML Level 3 Core has no direct support for providing random values within a model. Currently there is no workaround and this makes it impossible to describe models that use probability distributions.

2.2 Past work on this problem or similar topics

2.2.1 The Newcastle Proposal

In 2005 there was a proposal from Colin Gillespie and others [3] to introduce support for probability distributions in the SBML core specification. This was based on their need to use such distributions to represent the models they were creating as part of the BASIS project http://www.basis.ncl.ac.uk).

They proposed that distributions could be referred to in SBML using the **csymbol** element in the MathML subset used by the SBML Core specification. An example is below:

```
<xmlns=''http://www.w3.org/1998/Math/MathML''>
  <apply>
    <csymbol encoding=''text''
        definitionURL=''http://www.sbml.org/sbml/symbols/uniformRandom''>
      uniformRandom
    </csymbol>
    <ci>mu</ci>
    <ci>sigma</ci>
  </apply>
</math>
```

This required that a library of definitions be maintained as part of the SBML standard and in their proposal they defined an initial small set of commonly used distributions. The proposal was never implemented.

2.2.2 Seattle 2010

The "distrib" package was discussed at the Seattle SBML Hackathon[4]. There one of the authors (DW) presented an overview of the problem[5][6], building on the old proposal from the Newcastle group (see above: 2.2.1). There was broad support at the meeting for development of such a package, and for the proposed feature set. Discussion following the presentation led to a consensus on the following points:

- There is an urgent need for such a package.
- It is important to make a distinction between a description of uncertainty regarding a model parameter and the mechanistic process of selecting a random number from a probability distribution, for applications such as parameter scans and experimental design
- It is probably worth including the definition of PMFs, PDFs and CDFs in the package
- It is worth including the definition of random distributions using particle representations within such a package, though some work still needs to be done on the precise representation
- It could be worth exploring the use of xinclude to point at particle representations held in a separate file
- Random numbers must not be used in rate laws or anywhere else that is continuously evaluated, as then simulation behaviour is not defined

[3] http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Distributions_and_Ranges
[4] http://sbml.org/Events/Hackathons/The_2010_SBML-BioModels.net_Hackathon
[5] Slides: http://sbml.org/images/3/3b/Djw-sbml-hackathon-2010-05-04.pdf
[6] Audio: http://sbml.org/images/6/67/Wilkinson-distributions-2010-05-04.mov

- Although there is a need for a package for describing extrinsic noise via stochastic differential equations in SBML, such mechanisms should not be included in this package due to the considerable implications for simulator developers
- We probably don't want to layer on top of UncertML (www.uncertml.org), as this spec is fairly heavy-weight, and somewhat tangential to our requirements
- A random number seed is not part of a model and should not be included in the package
- The definition of truncated distributions and the specification of hard upper and lower bounds on random quantities should be considered.

It was suggested that new constructs should be introduced into SBML by the package embedded as user-defined functions using the following syntax:

```
<listOfFunctionDefinitions>
  <functionDefinition id="myNormRand">
    <distrib:####>
      #### distrib binding information here ####
    </distrib:####>
    <math>
      <lambda>
        <bvar>
          <ci>mu</ci>
          <ci>sigma</ci>
        </bvar>
        <ci>mu</ci>
      </lambda>
    </math>
  </functionDefinition>
</listOfFunctionDefinitions>
```

which allows the use of a "default value" by simulators which do not understand the package (but simulators which do will ignore the <math> element). The package would nevertheless be "required", as it will not be simulated correctly by software which does not understand the package.

Informal discussions following the break-out covered topics such as how to work with vector random quantities in the absence of the vector element in the MathML subset used by SBML, and the care that must be taken with the semantics of random variables and the need to both a) reference multiple independent random quantities at a given time and b) make multiple references to the same random quantity at a given time.

2.2.3 Hinxton 2011

Detailed discussion was continued at the Statistical Models Workshop in Hinxton in June 2011[7]. There those interested in representing Statistical Models in SBML came together to work out the details of how this package would work in detail. Dan Cornford from the UncertML project[8] attended the meeting and described how that resource could be used to describe uncertainty and in particular probability distributions. Perhaps the most significant decision at this meeting was to adopt the UncertML resource as a controlled vocabulary that is referenced by the Distributions package.

Much of the detail and the examples in this proposal are based on the work of this workshop and so its outcomes are essentially presented below. However, at the end of the meeting there remained a number of unresolved issues that are still unresolved in this proposal. See section 7 on page 22 for more details.

2.2.4 HARMONY 2012: Maastricht

Two sessions were dedicated to discussion of Distributions at HARMONY based around the proposals described in version 0.5 of this document. In addition there was discussion about the Arrays and Sets proposal which was

[7] http://sbml.org/Events/Other_Events/statistical_models_workshop_2011
[8] http://www.uncertml.org/

very helpful in solving the problem of multivariate distributions in Distributions. The following were the agreed outcomes of the meeting:

- The original proposal included UncertML markup directly in the function definition. This proved unwieldy and confusing and has been replaced by a more elegant solution that eliminates the UncertML markup and integrates well with the fallback function (see details below).
- Multivariate distributions can be supported using the Arrays and Sets package to define a covariance matrix.
- User defined continuous distributions would define a PDF in MathML.
- Usage semantics were clarified so that invokation of a function definition implied a value was sampled from the specified distribution.
- It was agreed from which sections of an SBML model a distribution could be invoked.
- Statistical descriptors of variables (for example mean and standard deviation) would be separated from Distributions and either provided in a new package or in a later version of SBML L3 core.

3 Proposed syntax and semantics

3.1 Colour Conventions

Throughout this document, in all UML diagrams, classes that exists in SBML Level 3 core or another existing standard are displayed in black. If those elements are extended in this proposal, those extensions are displayed in green. Classes that are new to this proposal are shown in blue.

3.2 Defining Distributions

The Distributions package has a very simple purpose. It defines a statistical distribution, range or statistic from which you can obtain a value that is used in other parts of an SBML model. The distribution is defined using UncertML, which is a resource that defines probabilistic uncertainties Williams et al. (2009), or explicitly using MathML Ausbrooks et al. (2003).

SBML Core is extended at the **FunctionDefinition** class as can be seen in the UML representation in figure 1. The extended **FunctionDefinition** can optionally contain a single instance of a new class **Distribution** that is part of the distrib package. The **Distribution** class can then either refer to an element in UncertML using or alternatively a **Distribution** can contain a **Math** class that is used to define the distribution explicitly.

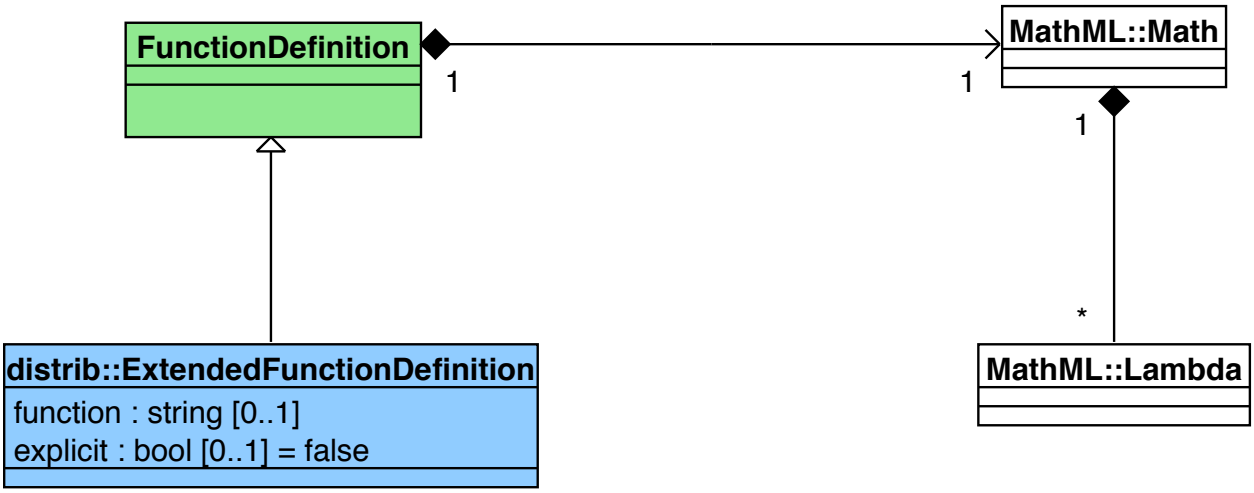

Figure 1: *UML class diagram for the Distributions package. This diagram describes the classes involved and not instances of those classes. Note the UML packages indicate that the classes belong to different namespaces. The namespace of the SBML Core is not shown.*

3.3 Extension Handling

The Distributions package is defined as a required package, which means that an SBML model that uses that extension will not be correctly defined unless the features of the extension are used and interpreted correctly by supporting software.

3.4 FunctionDefinition Extension

As outlined above the **FunctionDefinition** is extended to include the attributes defined below.

`function`

This provides the external definition of the distribution to be invoked. The value of the attribute should be a URI that uniquely identifies a definition from an external resource. The exact nature of this URI and the external

resource to use is discussed below (section 3.7). The attribute is optional.

`explicit`

This attribute indicates that a user-defined (or explicit) continuous distribution will be defined. If it has a value of `true` then the MathML expression in the body of the function definition should define a probability definition function. In this case this also serves as the fallback function. This attribute is optional and a default value of `false` is assumed if it is not set explicitly. Note that a user define discrete distribution is defined in a different way (see section 5.4).

3.4.1 Usage Rules

The `function` cannot have a value (i.e. it must be `null`) when the `explicit` attribute has a value of `true`.

3.5 Using the distrib package

The distribution defined by a **FunctionDefinition** instance can be used by instances of the following SBML classes:

- InitialAssignment
- EventAssignments
- Delays
- Priorities

An invocation of a distribution behaves like any other function call in SMBL and the function is executed. For a statistical distribution this means that one or more values are sampled from the distribution each time the function definition is invoked.

3.6 Permitted Types

The Distributions package will support arrays and matrices via the Arrays and Sets package. This means that paramaters passed to a function definition can be arrays, matrices or scalar values and that a functions return type can be either a scalar or the above non-scalar types. Type consistency and conversion behaviour between scalar and non-scalar types is defined by the Arrays and Sets package.

3.7 External Definition of Distributions

The external definition is a URI that should refer to a standardised dictionary of distributions. It should unambiguously define the following:

- A globally unique URI by which to refer to it.
- A meaningful name
- The parameter arguments used by the function, including for each argument: its name and type (scalar or non-scalar). Note that all arguments are mandatory.
- The value and its type returned by the function.
- A detailed mathematics description of the statistical distribution being sampled from.

3.8 Equivalence with Fallback Function

The MathML definition contained in the function definition is not used if distrib is supported and the `explicit` attribute is `false`. However, in order to ensure the SBML model is still valid when distrib is not supported the fallback function must be present and it should conform to the following rules:

- Match the number of arguments in the external function definition.
- Each argument should match the type of the equivalent argument in the external function.
- Have the same return type as the external function.

Clearly these rules can only be enforced when Distributions is supported.

4 Package dependencies

This package is dependent on the Arrays and Sets package to provide array and matrix structures. It is also dependent on MathML Ausbrooks et al. (2003) to define distributions explicitly. It uses the the subset of MathML set out in the SMBL Level 3 Core Specification Hucka et al. (2010).

5 Use-cases and examples

5.0.1 Sampling from a distribution: PK/PD Model

This is a very straightforward use of an externally defined distribution. The key point to note is that a value is sampled from the distribution and assigned to a variable when it is invoked. In the initialAssignments element in this example. Later use of the variable does not result in re-sampling from the distribution. This is consistent with current SBML semantics.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
    xmlns:distrib="http://www.sbml.org/sbml/level3/version1/distrib/version1"
    distrib:required="true">
    <model id="PKModel" name="PKModel" substanceUnits="item"
          timeUnits="second" volumeUnits="litre" extentUnits="item">
        <listOfFunctionDefinitions>
            <functionDefinition id="logNormal"
         distrib:function="http:///identifiers.org/Distribution/LogNormal">
          <!-- below is a dummy function provided to ensure valid SBML Core
        for software that do not understand distrib. -->
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <lambda>
                        <bvar><ci>mu</ci></bvar>
                        <bvar><ci>sigma</ci></bvar>
                        <apply>
                            <ci>mu</ci>
                        </apply>
                    </lambda>
                </math>
            </functionDefinition>
        </listOfFunctionDefinitions>
        <listOfCompartments>
            <compartment id="central" name="central" size="0" constant="true"/>
            <compartment id="gut" name="gut" size="0" constant="true"/>
        </listOfCompartments>
        <listOfSpecies>
            <species id="Qc" compartment="central" initialAmount="1"
               hasOnlySubstanceUnits="true" boundaryCondition="false" constant="false"/>
            <species id="Qg" compartment="gut" initialAmount="1"
               hasOnlySubstanceUnits="true" boundaryCondition="false" constant="false"/>
        </listOfSpecies>
        <listOfParameters>
            <parameter id="ka" value="1" constant="true"/>
            <parameter id="ke" value="1" constant="true"/>
            <parameter id="Cc" value="1" constant="false"/>
            <parameter id="Cc_obs" value="1" constant="false"/>
        </listOfParameters>
        <listOfInitialAssignments>
            <initialAssignment symbol="central">
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <apply>
                        <ci>logNormal</ci>
                        <cn>0.5</cn>
                        <cn>0.1</cn>
                    </apply>
                </math>
            </initialAssignment>
            <initialAssignment symbol="ka">
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <apply>
                        <ci>logNormal</ci>
                        <cn>0.5</cn>
                        <cn>0.1</cn>
                    </apply>
```

```
                </math>
            </initialAssignment>
            <initialAssignment symbol="ke">
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <apply>
                        <ci>logNormal</ci>
                        <cn>0.5</cn>
                        <cn>0.1</cn>
                    </apply>
                </math>
            </initialAssignment>
        </listOfInitialAssignments>
        <listOfRules>
            <assignmentRule variable="Cc">
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <apply>
                        <divide/>
                        <ci> Qc </ci>
                        <ci> central </ci>
                    </apply>
                </math>
            </assignmentRule>
            <assignmentRule variable="Cc_obs">
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <apply>
                        <plus/>
                        <ci> Cc </ci>
                        <cn type="integer"> 1 </cn>
                    </apply>
                </math>
            </assignmentRule>
        </listOfRules>
        <listOfReactions>
            <reaction id="absorption" reversible="false" fast="false">
                <listOfReactants>
                    <speciesReference species="Qg" stoichiometry="1" constant="false"/>
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Qc" stoichiometry="1" constant="false"/>
                </listOfProducts>
                <kineticLaw>
                    <math xmlns="http://www.w3.org/1998/Math/MathML">
                        <apply>
                            <times/>
                            <ci> ka </ci>
                            <ci> Qg </ci>
                        </apply>
                    </math>
                </kineticLaw>
            </reaction>
            <reaction id="excretion" reversible="false" fast="false">
                <listOfReactants>
                    <speciesReference species="Qc" stoichiometry="1" constant="false"/>
                </listOfReactants>
                <kineticLaw>
                    <math xmlns="http://www.w3.org/1998/Math/MathML">
                        <apply>
                            <divide/>
                            <apply>
                                <times/>
                                <ci> ke </ci>
                                <ci> Qc </ci>
                            </apply>
                            <ci> central </ci>
                        </apply>
                    </math>
                </kineticLaw>
```

```
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

5.1 Truncated distribution

To encode a truncated distribution we rely on external definitions. Clearly it would be cumbersome if every distribution and multivariate distribution required a truncated equivalent definition, but at present this is what is required. Perhaps this problem could be solved if optional function arguments were permitted, but this cases problems with the fallback function (see section 7).

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
      xmlns:distrib="http://www.sbml.org/sbml/level3/version1/distrib/version1"
      distrib:required="true">
  <!-- Note that this is a code snippet and not a valid model -->
    <model>
        <listOfFunctionDefinitions>
      <!-- We treat truncated distributions as any other distribution. The external
    defined function handles the truncation correctly. -->
            <functionDefinition id="truncNormal"
          distib:function="http://identifiers.org/truncatedNormal">
                <math xmlns="http://www.w3.org/1998/Math/MathML"
          xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
                    <lambda>
                        <bvar><ci>mu</ci></bvar>
                        <bvar><ci>sigma</ci></bvar>
                        <bvar><ci>lower</ci></bvar>
                        <bvar><ci>upper</ci></bvar>
      <apply>
                            <ci>mu</ci>
      </apply>
                    </lambda>
                </math>
            </functionDefinition>
            </listOfFunctionDefinitions>
    <listOfParameters>
      <parameter id="V" constant="true"/>
      <parameter id="V_pop" constant="true"/>
      <parameter id="V_omega" constant="true"/>
      <parameter id="V_upper" constant="true"/>
      <parameter id="V_lower" constant="true"/>
    </listOfParameters>
    <initialAssignments>
      <initialAssignment symbol="V_pop">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <cn>105</cn>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="V_upper">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <cn>150</cn>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="V_lower">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
```

```
    <apply>
      <cn>15</cn>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="V_omega">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <cn>0.70</cn>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="V">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <ci>multivariateNormal</ci>
      <ci>V_pop</ci>
      <ci>V_omega</ci>
      <ci>V_upper</ci>
      <ci>V_lower</ci>
    </apply>
  </math>
      </initialAssignment>
    </initialAssignments>
    </model>
</sbml>
```

5.2 Multivariate distribution

In this example two correlated parameters are sampled from a multivariate distribution. The correlation is defined using a covariance matrix and the sampled values are returned as a vector of 2 values, which are then assigned to the individual parameters. This example relies heavily on the arrays package, which will be a dependency of the distrib package.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
      xmlns:distrib="http://www.sbml.org/sbml/level3/version1/distrib/version1"
      distrib:required="true"
      xmlns:arrays="http://www.sbml.org/sbml/level3/version1/arrays/version1"
      arrays:required="true">
  <!-- NOTE: This requires the arrays package! -->
  <model id="MultivariateExample" name="Multivariate␣Example" substanceUnits="item"
   timeUnits="second" volumeUnits="litre" extentUnits="item">
    <listOfFunctionDefinitions>
      <functionDefinition id="multivariateNormal"
  distrib:function="http://identifiers.org/Distribution/MultivariateNormal">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <lambda>
      <bvar><ci>meanVector</ci></bvar>
      <bvar><ci>covarianceMatrix</ci></bvar>
      <apply>
        <ci>meanVector</ci>
      </apply>
    </lambda>
  </math>
      </functionDefinition>
    </listOfFunctionDefinitions>
    <listOfParameters>
      <parameter id="V" constant="true"/>
      <parameter id="V_pop" constant="true"/>
      <parameter id="V_omega" constant="true"/>
```

```
    <parameter id="Cl" constant="true"/>
    <parameter id="Cl_pop" constant="true"/>
    <parameter id="Cl_omega" constant="true"/>
    <parameter id="covariance" constant="true">
<arrays:listOfDimensions>
  <arrays:dimension id="i" lowerLimit="1" upperLimit="2" />
  <arrays:dimension id="j" lowerLimit="1" upperLimit="2" />
</arrays:listOfDimensions>
    </parameter>
    <parameter id="correlated_means" constant="true">
<arrays:listOfDimensions>
  <arrays:dimension id="i" lowerLimit="1" upperLimit="2" />
</arrays:listOfDimensions>
    </parameter>
    <parameter id="correlated_params" constant="true">
<arrays:listOfDimensions>
  <arrays:dimension id="i" lowerLimit="1" upperLimit="2" />
</arrays:listOfDimensions>
    </parameter>
  </listOfParameters>

  <listOfInitialAssignments>
    <initialAssignment symbol="V_pop">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
  <apply>
    <cn>105</cn>
  </apply>
</math>
    </initialAssignment>
    <initialAssignment symbol="V_omega">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
  <apply>
    <cn>0.70</cn>
  </apply>
</math>
    </initialAssignment>
    <initialAssignment symbol="Cl_pop">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
  <apply>
    <cn>73</cn>
  </apply>
</math>
    </initialAssignment>
    <initialAssignment symbol="Cl_omega">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
  <apply>
    <cn>0.70</cn>
  </apply>
</math>
    </initialAssignment>
    <initialAssignment symbol="covariance">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/arraymaths/version1">
  <!-- This is an unresolved issue. The l3 V1 Core mathml subset does not support
       matrices. One solution - as above is to use a different MathML subset definition.
  -->
  <matrix>
    <matrixrow>
      <apply><times/><ci>V_omega</ci><ci>V_omega</ci></apply>
      <apply><times/><ci>V_omega</ci><ci>C_omega</ci><ci>V_C_rho</ci></apply>
    </matrixrow>
    <matrixrow>
      <ci>0</ci>
```

```
        <apply><times/><ci>C_omega</ci><ci>C_omega</ci></apply>
      </matrixrow>
    </matrix>
  </math>
      </initialAssignment>
      <initialAssignment symbol="correlated_means">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <vector>
      <ci>V_pop</ci>
      <ci>C_pop</ci>
    </vector>
  </math>
      </initialAssignment>
      <initialAssignment symbol="correlated_params" >
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <ci>multivariateNormal</ci><ci>correlated_means</ci><ci>covariance</ci>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="Cl">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <selector/>
      <ci>correlated_params</ci>
      <cn type="integer">2</cn>
    </apply>
  </math>
      </initialAssignment>
      <initialAssignment symbol="V">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
        xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
    <apply>
      <selector/>
      <ci>correlated_params</ci>
      <cn type="integer">1</cn>
    </apply>
  </math>
      </initialAssignment>
    </listOfInitialAssignments>
    <-- This is an incomplete model snippet, sufficient to illustrate the use of
   a multivariate distribution. -->
  </model>
</sbml>
```

5.3 User-defined continuous distribution

In this example an additional construct is used to indicate:

- that the MathML within the function definition defines a PDF and so is not a fallback.
- that when invoked a value should be sampled from this PDF.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
      xmlns:distrib="http://www.sbml.org/sbml/level3/version1/distrib/version1"
      distrib:required="true">
  <!-- Code snippet. Not a valid model. -->
  <model id="UserDefined" name="User␣Defined␣Example" substanceUnits="item"
         timeUnits="second" volumeUnits="litre" extentUnits="item">
    <listOfFunctionDefinitions>
```

```
      <functionDefinition id="normal" distrib:explicit="true">
        <!-- Function defines a PDF using MathML. The implied behaviour is that
            A value will be sampled from the distribution described by this PDF -->
        <math xmlns="http://www.w3.org/1998/Math/MathML">
          <lambda>
            <bvar><ci>mu</ci></bvar>
            <bvar><ci>omega</ci></bvar>
            <apply>
              <apply>
                <times />
                <apply>
                  <power />
                  <exponentiale />
                  <apply>
                    <times />
                    <apply>
                      <times />
                      <cn type='integer'>-1</cn>
                      <cn type='rational'>1<sep />2</cn>
                    </apply>
                    <apply>
                      <power />
                      <apply>
                        <times />
                        <apply>
                          <plus />
                          <ci>x</ci>
                          <apply>
                            <times />
                            <cn type='integer'>-1</cn>
                            <ci>mu</ci>
                          </apply>
                        </apply>
                        <apply>
                          <power />
                          <ci>sigma</ci>
                          <cn type='integer'>-1</cn>
                        </apply>
                      </apply>
                      <cn type='integer'>2</cn>
                    </apply>
                  </apply>
                </apply>
                <apply>
                  <power />
                  <apply>
                    <times />
                    <ci>omega</ci>
                    <apply>
                      <ci>sqrt</ci>
                      <apply>
                        <times />
                        <cn type='integer'>2</cn>
                        <pi/>
                      </apply>
                    </apply>
                  </apply>
                  <cn type='integer'>-1</cn>
                </apply>
              </apply>
            </apply>
          </lambda>
        </math>
      </functionDefinition>
      <listOfParameters>
        <parameter id="V" constant="true"/>
        <parameter id="V_pop" value="100" constant="true"/>
```

```
        <parameter id="V_omega" value="0.25" constant="true"/>
      </listOfParameters>
      <listOfInitialAssignments>
        <initialAssignment symbol="V">
          <math xmlns="http://www.w3.org/1998/Math/MathML">
            <apply>
              <ci>normal</ci>
              <cn>V_pop</cn>
              <cn>V_omega</cn>
            </apply>
          </math>
        </initialAssignment>
      </listOfInitialAssignments>
    </model>
  </sbml>
```

5.4 User-defined discrete distribution

In this example we don't use any special distrib features, by invoke an externally defined function that constructs a PMF from a matrix of values and their associated probabilities. The example is not biologically meaningful, but illustrates that the matrix contains the values and probabilities possible when throwing two dice. These values are encoded in a matrix, which is passed as a parameter in a function, however, it may be that this information is defined using NuML or by referring to an external file.

? ?

? There has been very little discussion about how to treat user-defined PMFs so this example should be regarded as a starting point for discussion rather than a final proposal.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
      xmlns:distrib="http://www.sbml.org/sbml/level3/version1/distrib/version1"
      distrib:required="true"
      xmlns:arrays="http://www.sbml.org/sbml/level3/version1/arrays/version1"
      arrays:required="true">
  <!-- Code snippet. Not a valid model. -->
  <model id="UserDefined" name="User␣Defined␣Example" substanceUnits="item"
         timeUnits="second" volumeUnits="litre" extentUnits="item">
    <listOfFunctionDefinitions>
      <functionDefinition id="myDistribution"
         distrib:function="http:///identifiers.org/Distribution/UserDefinedDiscrete">
          <!-- below is a dummy function provided to ensure valid SBML Core
        for software that do not understand distrib. -->
                <math xmlns="http://www.w3.org/1998/Math/MathML">
                    <lambda>
                        <bvar><ci>param1</ci></bvar>
          <apply>
                        <ci>param1</ci>
          </apply>
                    </lambda>
                </math>
      </functionDefinition>
      </listOfFunctionDefinitions>
      <listOfParameters>
        <parameter id="W" constant="true"/>
      <parameter id="outcomes" constant="true">
  <arrays:listOfDimensions>
    <arrays:dimension id="i" lowerLimit="1" upperLimit="11" />
    <arrays:dimension id="j" lowerLimit="1" upperLimit="11" />
  </arrays:listOfDimensions>
      </parameter>
      </listOfParameters>
      <listOfInitialAssignments>
        <initialAssignment symbol="W">
  <math xmlns="http://www.w3.org/1998/Math/MathML"
```

```
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/core">
  <apply>
    <ci>myDistribution</ci>
    <ci>outcomes</ci>
  </apply>
</math>
</initialAssignment>
<initialAssignment symbol="outcomes">
<math xmlns="http://www.w3.org/1998/Math/MathML"
      xmlns:sbml="http://www.sbml.org/sbml/level3/version1/arraymaths/version1">
  <!-- This is an unresolved issue. The l3 V1 Core mathml subset does not support
       matrices. One solution - as above is to use a different MathML subset definition.
       We may choose to use numml here too.
  -->
  <matrix>
    <matrixrow>
      <cn>2</cn>
      <cn>3</cn>
      <cn>4</cn>
      <cn>5</cn>
      <cn>6</cn>
      <cn>7</cn>
      <cn>8</cn>
      <cn>9</cn>
      <cn>10</cn>
      <cn>11</cn>
      <cn>12</cn>
    </matrixrow>
    <matrixrow>
      <apply><divide/><cn>1</cn><cn>21</cn></apply>
      <apply><divide/><cn>1</cn><cn>21</cn></apply>
      <apply><divide/><cn>2</cn><cn>21</cn></apply>
      <apply><divide/><cn>2</cn><cn>21</cn></apply>
      <apply><divide/><cn>3</cn><cn>21</cn></apply>
      <apply><divide/><cn>3</cn><cn>21</cn></apply>
      <apply><divide/><cn>3</cn><cn>21</cn></apply>
      <apply><divide/><cn>2</cn><cn>21</cn></apply>
      <apply><divide/><cn>2</cn><cn>21</cn></apply>
      <apply><divide/><cn>1</cn><cn>21</cn></apply>
      <apply><divide/><cn>1</cn><cn>21</cn></apply>
    </matrixrow>
  </matrix>
</math>
      </initialAssignment>
    </listOfInitialAssignments>
  </model>
</sbml>
```

6 Prototype implementations

None as yet.

7 Unresolved issues

UncertML to snapshot or not A key decision is what form the external definition of the distributions will take. In particular should we use UncertML or a definition based on a snapshot of it.

Optional function arguments It would be useful to .

8 Acknowledgements

Much of the initial concrete work leading to this proposal document was carried out at the Statistical Models Workshop in Hinxton this year (2011). A list of participants and recordings of the discussion is available from http://sbml.org/Events/Other_Events/statistical_models_workshop_2011. The authors would also like to thank the participants of the distrib sessions during HARMONY 2012 for their excellent contributions in helping revising this proposal into something a lot closer to the finished version.

The authors would like to thank Sarah Keating and Maciej Swat for useful discussions; and Mike Hucka for LaTeX advice and his beautiful template upon which this document is based.

References

Ausbrooks, R., Buswell, S., Carlisle, D., Dalmas, S., Devitt, S., Diaz, A., Froumentin, M., Hunter, R., Ion, P., Kohlhase, M., Miner, R., Poppelier, N., Smith, B., Soiffer, N., Sutor, R., and Watt, S. (2003). Mathematical Markup Language (MathML) Version 2.0 (Second Edition). Available online via the Worldwide Web at http://www.w3.org/TR/MathML/.

Hucka, M., Bergmann, F. T., Hoops, S., Keating, S. M., Sahle, S., Schaff, J. C., Smith, L. P., and Wilkinson, D. J. (2010). The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core. Available via the World Wide Web at http://sbml.org/Documents/Specifications.

Williams, M., Cornford, D., Bastin, L., and Pebesma, E. (2009). Uncertainty Markup Language (UnCertML). Available online via the Worldwide Web at http://portal.opengeospatial.org/files/?artifact_id=33234.