SBML Level 3 package: Hierarchical Model Composition, Version 1 Release 3

Lucian P. Smith[1], Michael Hucka[2], Stefan Hoops[3], Andrew Finney[4], Martin Ginkel[5], Chris J. Myers[6], Ion Moraru[7], Wolfram Liebermeister[8]

[1]Department of Bioengineering, University of Washington, Seattle, WA, USA

[2]Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, USA

[3]Virginia Bioinformatics Institute, Virginia Tech, Blacksburg, VA, USA

[4]No affiliation

[5]Dynamics of Complex Technical Systems, Max Planck Institute, Magdeburg, Germany

[6]Electrical and Computer Engineering, University of Utah, Salt Lake City, UT, USA

[7]University of Connecticut Health Center, University of Connecticut, Farmington, CT, USA

[8]Molecular Genetics, Max Planck Institute, Berlin, Germany

Summary

Constructing a model in a hierarchical fashion is a natural approach to managing model complexity, and offers additional opportunities such as the potential to re-use model components. The *SBML Level 3 Version 1 Core* specification does not directly provide a mechanism for defining hierarchical models, but it does provide a mechanism for SBML *packages* to extend the Core specification and add additional syntactical constructs. The SBML *Hierarchical Model Composition* package for SBML Level 3 adds the necessary features to SBML to support hierarchical modeling. The package enables a modeler to include submodels within an enclosing SBML model, delete unneeded or redundant elements of that submodel, replace elements of that submodel with element of the containing model, and replace elements of the containing model with elements of the submodel. In addition, the package defines an optional "port" construct, allowing a model to be defined with suggested interfaces between hierarchical components; modelers can chose to use these interfaces, but they are not required to do so and can still interact directly with model elements if they so chose. Finally, the SBML Hierarchical Model Composition package is defined in such a way that a hierarchical model can be "flattened" to an equivalent, non-hierarchical version that uses only plain SBML constructs, thus enabling software tools that do not yet support hierarchy to nevertheless work with SBML hierarchical models.