THE CASE FOR USING MARKUP FOR BIOMECHANICAL MODELLING

David Johnson and Steve McKeever
Oxford University Computing Laboratory; email: david.johnson@comlab.ox.ac.uk,
web: www.comlab.ox.ac.uk

INTRODUCTION

Approaches to modelling biological systems take many forms. The mathematical, computational and encapsulated components of biomechanical models can be diverse in terms of complexity and scale, as well as in published implementation (mathematics, source code, and executable files). Markup languages for biomodelling have emerged as a solution to unifying approaches to publishing models to the research community.

We have previously discussed the potential for applying markup languages to the field of cancer modelling [1]. Specifically we discussed how model markup could be used to generate software code on-the-fly that can be used by computational execution frameworks such as Chaste [2], and also how generated code could be automatically optimised. We build on and generalize this vision and submit that the development of markup languages for biomechanical models will provide three key benefits. Firstly, by providing an expressive metadata vocabulary researchers will be able to appropriately curate their models and publish them to an audience of research peers and clinical scientists wishing to validate, trial, and apply published models. Secondly, automated validation of models will be possible where markup incorporating enough expressiveness to describe the underlying mathematics and logic of models can be interpreted and checked by computer-based tools. Thirdly, markup will be developed to describe abstract interfaces to the computational execution of the models. These abstractions can then be mapped to the appropriate biological entities that could be used to couple domain-specific models together. In this paper, we briefly review the current efforts of applying markup languages to computational biology, and argue that domain-specific markup languages are the way forward in biomechanical modelling.

METHODS

Existing markup languages of note include the Systems Biology Markup Language (SBML) [3], CellML [4], which emerged from efforts within cardiac modelling, FieldML [5] that aims to tackle geometric expression of spatial geometry in biological structures and the *In Silico* Markup Language (ISML) [6] developed out of the Japanese Physiome project.

Each of these markup languages addresses biological modelling by encapsulating the mathematics that underpins each component part of a model. However it should be noted that the expressiveness in each respective modelling language lacks somewhat when applied to more complex scenarios. For example, each modelling language bases its mathematical descriptions on MathML [7], a markup language developed by the World Wide Web Consortium (W3C) for describing mathematical formulae. While MathML consists of a mature vocabulary, it does not provide any way of expressing logic and control flow or complex data constructs. The models based on markup using MathML are typically simulated through solving ODEs and DAEs. SBML is a very specialised language and represents models through describing low-level molecular components and their relationships with each other. CellML relies on declarative mathematics that is interpreted and processed by numerical solvers, mainly to model biological cell function. The domain concepts in CellML are decoupled from the language and included as metadata. FieldML is being developed as a language to compliment CellML in modelling physiological structures based on geometric meshes and fields. ISML is similar to CellML is its application to a wide range of biology, and also demonstrates multi-scale application.

RESULTS AND DISCUSSION

CellML serves as an excellent exemplar of how to tackle biological modelling with markup. However we feel that CellML, although appropriate for describing a wide range of models, is limited in expressiveness when considering more complex phenomenon found in the biological modelling literature. Each of the markup languages reviewed describes models based mainly on solving mathematical formulae, however other modelling techniques include more algorithmic approaches. Control flow constructs, such as loops and conditional behaviours, data type collections, such as arrays and matrices, and domain-specific data objects, would give a markup language more expressive power, especially where models are developed using an *in silico* methodology rather than a pure mathematical approach.

As a very generic example, consider a model that may be based on finite-state machines (FSMs) whereby each biological cell's internal state is determined by events external to itself. States, transitions, conditions, and actions (all fundamental elements of FSMs) cannot be coded using any of SBML, CellML or FieldML. The definition of FSMs would require the use of formal logic that is not considered in any of these markup languages. We submit that the introduction of such constructs, which are commonly found in programming languages, would compliment pure mathematical modelling.

We do not propose that such an expansion on mathematical notation is done to produce general-purpose modelling markup. Using general-purpose languages for describing biological processes could introduce ambiguity in how models are developed and described. We also submit that the mathematics and computational descriptions should be encapsulated in domain specific components, and consequently models described using domain specific markup. Creating languages that are designed for particular biological applications, and using terminology that will be familiar to the field, will allow modellers to more easily focus on the scientific questions at hand rather than spending significant effort on learning how to use yet another general purpose programming language. Models will also become more modular allowing component reuse by restricting the concepts and biological terminology used in the markup descriptions. General purposes languages are not suited to domain-specific modularization as the specification of interfaces between model components is left entirely open to ambiguous definition.

For example, we are actively involved with the development of a markup language to describe computational cancer models within the European Commission funded Transatlantic Tumor Model Repositories project (www.tumor-project.eu). The challenges posed to developing markup for cancer modelling (we will refer to this markup language as *TumorML*) include formalizing cancer terminology, linking biological entities with computational and mathematical elements of models, and incorporating features to allow for curating models in online repositories. Paired with ontologies of how entities of cancer biology are related and interact, we may be able to package models with metadata that automates coupling different cancer models together, regardless of scale and source. Coupling models together may produce more accurate compound models as a result of combining approaches from different research teams.

As a first step, we will develop metadata for curation and for describing the public interfaces with existing models that have been developed and published as source code and executable files. This will allow us to investigate how to fuse models of different scales together through their exposed parametric inputs and outputs; an initial 'black box' approach to computational model execution and coupling. While the cancer modelling community adopts TumorML for publishing existing models, we will work with modellers to develop the next level of more detailed abstraction in the structural, mathematical, and algorithmic descriptions of the inner workings of models. Significant effort might be required to port existing models to TumorML, so by providing multiple levels of abstractive notation in our markup, we can wrap existing models in TumorML as well as develop new models with the same markup specification. Like SBML, CellML, FieldML, and ISML, TumorML will utilize existing metadata vocabularies such as Dublin Core (www.dublincore.org) for document curation and MathML for providing validated mathematical content where possible, and where existing vocabularies do not exist we will specify our own cancer-specific metadata descriptions.

CONCLUSIONS

In this paper we reviewed the current state-of-the-art methods in using markup languages for biological modelling with SBML, CellML, FieldML, and ISML. We described how although each of these markup languages are relatively mature and are still being refined, each approach is limited by either being too specific to a single scale (SBML), by being unrestricted in its application to different biological modelling domains (CellML, ISML), or by being directed at a particular aspect of modelling (FieldML). Also each of these markup languages uses declarative mathematics as their basis, again limiting the kinds of models that can be developed. We argue that introducing control flow constructs, complex data types, and domain-specific encapsulation is the way forward in modelling with markup. Models can then be more easily shared, reused, and combined to further the state of the art.

ACKNOWLEDGEMENTS

This work was supported in part by the EC under the Transatlantic Tumor Model Repositories project (#FP7-ICT-2009.5.4-247754). The authors would also like to thank the participating institutions of the project consortium and its members.

REFERENCES

1. Johnson D, et al. Markup Languages for In Silico Oncology. In *Proc. 4th Int. Adv. Res. Workshop on In Silico Oncology and Cancer Investigation (4th IARWISOCI) – The ContraCancrum Workshop.* 108–110, September 2010.
2. Pitt-Francis J, et al. Chaste: a Test driven Approach to Software Development for Biological Modelling. *Computer Physics Communications*, **180**(12), 2452-2471, December 2009.
3. Hucka M, et al, Evolving a Lingua Franca and Associated Software Infrastructure for Computational Systems Biology: The Systems Biology Markup Language (SBML) Project. *Systems Biology*, **1**(1), 41-53, June 2004.
4. Lloyd CM, et al. CellML: its future, present and past. *Progress in Biophysics and Molecular Biology*, **85**, 433-450, 2004.
5. Christie GR, et al. FieldML: Concepts and Implementation. *Philosophical Transactions of the Royal Society A.*, **367**(1895), 1869-1884, May 2009.
6. Asai Y, et al. Specifications of insilicoML 1.0 : A Multilevel Biophysical Model Description Language. *Journal of Physiological Sciences,* **58**(7), 447-458. December 2008.
7. Carlisle D (ed.), et al, Mathematical Markup Language (MathML) Version 3.0, *W3C Recommendation 21 October 2010*, October 2010.