Possibilities for SED-ML

Jonathan Cooper Gary Mirams James Osborne

Computational Biology Group
Department of Computer Science
University of Oxford

August 15, 2012

Outline

What problem are we trying to solve?

- How can the whole modelling process be improved?

What problem are we trying to solve?

- How can the whole modelling process be improved?
- How can models be reused and composed reliably?

What problem are we trying to solve?

- How can the whole modelling process be improved?
- How can models be reused and composed reliably?
- Especially as models increase in size and complexity

What problem are we trying to solve?

- How can the whole modelling process be improved?
- How can models be reused and composed reliably?
- Especially as models increase in size and complexity

- We want . . .
  - A single model description that can be used/analysed in multiple ways

What problem are we trying to solve?

- How can the whole modelling process be improved?
- How can models be reused and composed reliably?
- Especially as models increase in size and complexity

- We want . . .
  - A single model description that can be used/analysed in multiple ways
  - Separation of model structure (i.e. the equations describing biological function) and experimental scenario

What problem are we trying to solve?

- How can the whole modelling process be improved?
- How can models be reused and composed reliably?
- Especially as models increase in size and complexity

- We want . . .
  - A single model description that can be used/analysed in multiple ways
  - Separation of model structure (i.e. the equations describing biological function) and experimental scenario
  - A uniform approach to model fitting, simulation, comparison and validation

Definitions

- A model is a purposeful simplification of reality
- An experiment is the process of stimulating a system to elicit observable responses
- A protocol is a set of detailed instructions for carrying out an experiment
  - Environmental conditions / parameters, interventions, recordings, filtering & post-processing, numerical algorithms, etc.

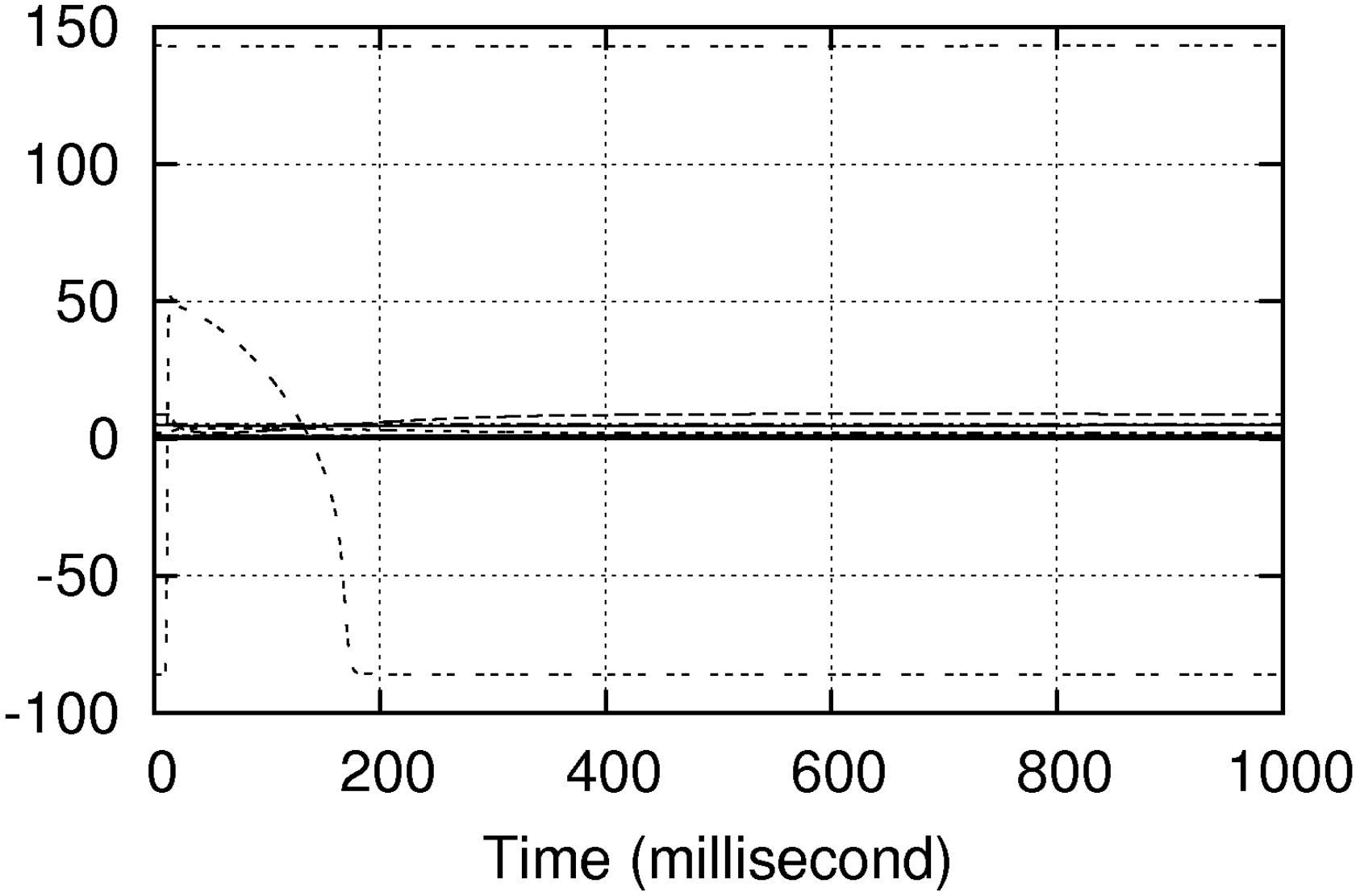
Use cases

- Functional curation of cardiac cell models
- Parameter sweeping models of multi-cellular tissue dynamics
- Experiments in immunology and synthetic biology

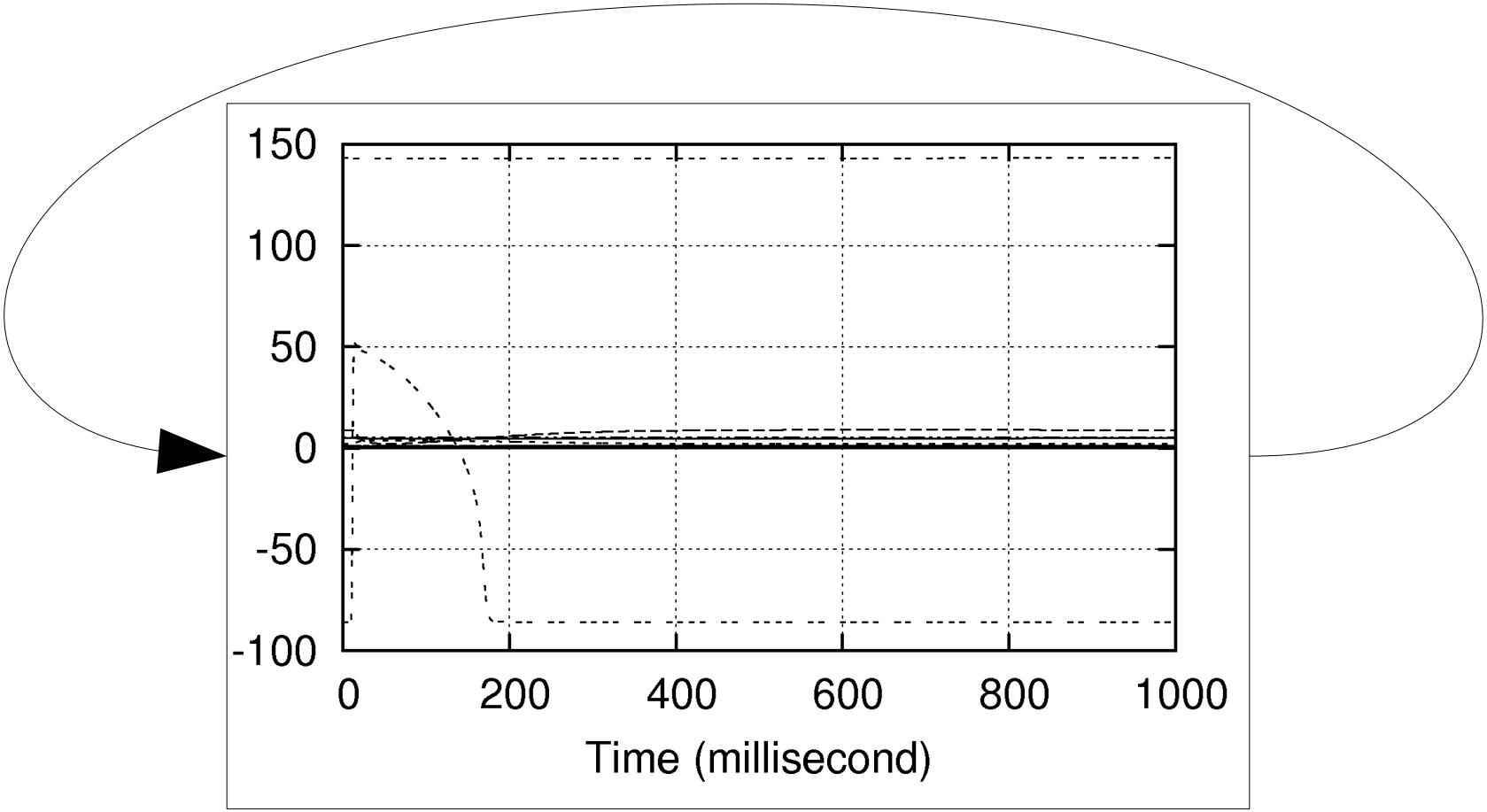
Functional curation of cardiac models

- Our original application; see also last year's talk
- Aim to create a comprehensive library of electrophysiology protocols, and screen the models in the CellML model repository
- One protocol that demonstrates new features since last year is finding a "steady state" for a given pacing rate

Pacing to “steady state”



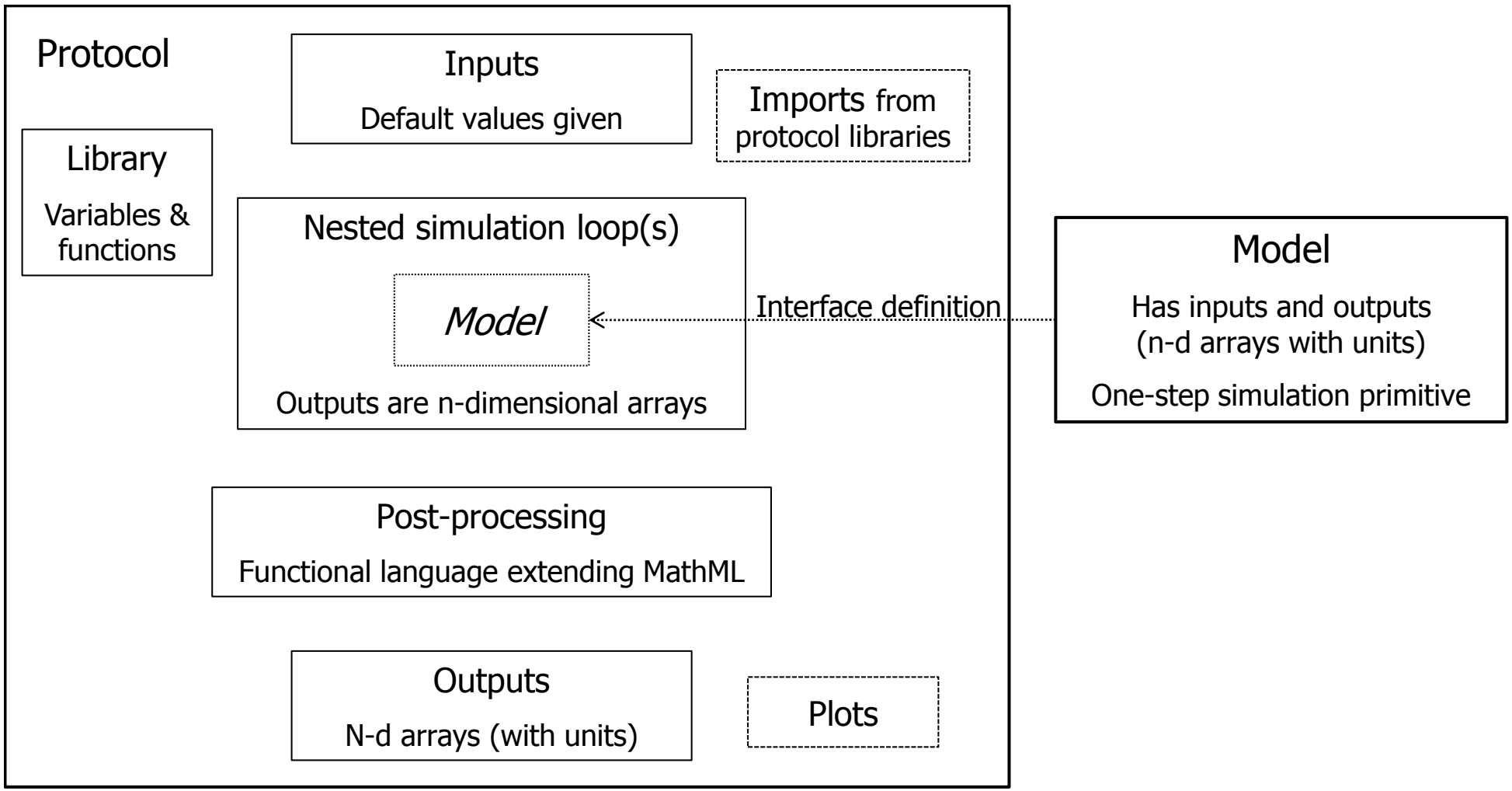
Pacing to “steady state”



Intestinal crypt — cell-based Chaste

Simulation movie

Our framework



Some proposals for SED-ML

- Applying a protocol to any model
  - Using ontological annotations
- New task hierarchy
  - Split up repeatedTask
- Extending variable references
  - Chaining post-processing
- (Handling n-dimensional data)

Proposal 1: Applying a protocol to any model

Problem: In order to compare models under a given protocol, we require that the protocol does not hardcode the model to use.

Proposal: Allow the source attribute on model to hold a special SED-ML URN, urn:sedml:anymodel say, that signals to processing software that the specific model must be supplied by some external mechanism.

Proposal 1: Accessing model variables

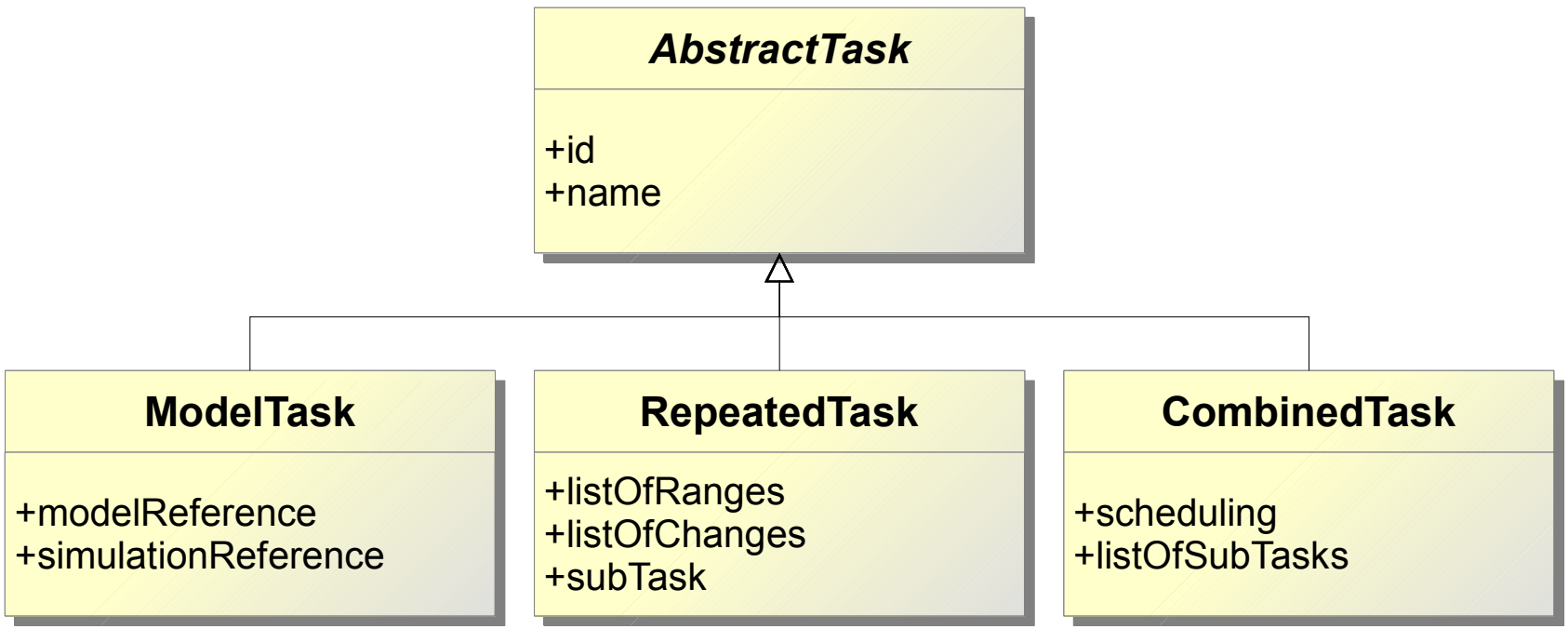
Problem: Different models may use different names for the same concept. How can a single protocol be applied to both?
Ontological annotation can provide a consistent nomenclature.
Model and protocol need to agree on ontology to use.
Proposal: Allow the variable element to use an ontology term instead of an XPath expression.
Either in the target attribute itself, or a new annotatedTarget attribute.

Proposal 2: New task hierarchy

Problem: Frank's repeatedTask is great, but includes too much:

- Associating a model with a simulation
- Repeating a task (potentially with changes)
- Grouping tasks to be run together (in sequence or parallelisable)

Proposal: Separate these concepts into separate classes:



Proposal 2: Examples (1)

```
<!-- Test when order matters -->
<combinedTask id="seq" scheduling="sequential">
    <listOfSubTasks>
        <subTask task="task1" resetModel="false"/>
        <subTask task="task2" resetModel="false"/>
    </listOfSubTasks>
</combinedTask>

<!-- Test when order does not matter -->
<combinedTask id="para" scheduling="parallel">
    <listOfSubTasks>
        <subTask task="task1" resetModel="true"/>
        <subTask task="task2" resetModel="true"/>
    </listOfSubTasks>
</combinedTask>
```

Proposal 2: Examples (2)

```
<!-- Test nesting a combined task -->
<nestedTask id="nested" resetModel="false"
            range="loop_counter">
    <listOfRanges>
        <vectorRange id="loop_counter">
            <value>0</value>
            <value>1</value>
            <value>2</value>
        </vectorRange>
    </listOfRanges>
    <subTask task="para"/>
</nestedTask>

<!-- Test combined inside combined -->
<combinedTask id="c_in_c" scheduling="sequential">
    <listOfSubTasks>
        <subTask task="seq" resetModel="false"/>
    </listOfSubTasks>
</combinedTask>
```

Proposal 2: Getting the results out

Problem: Repeated and combined tasks lead to more complex results data structures than the vectors in SED-ML L1V1.

- Repeated tasks yield n-dimensional arrays — see later
- Combined tasks yield “sub-results”

Proposal: Enhance the taskReference attribute on variable to be able to indicate sub-tasks, e.g. taskReference="task:subtask".
Continuing the examples from earlier:

```
<variable id="para_V1" taskReference="para:task1"
          target="..."/>
<variable id="nested_V1" taskReference="nested:task1"
          target="..."/>
```

Proposal 3: Accessing protocol variables

Problem: Sometimes we want to refer to variables defined in the *protocol*, not the model.

- Data generators cannot currently take the outputs of other data generators as inputs.
- Changes in nested tasks need to refer to range values.

Proposal: Most SED-ML elements already have an id attribute. Allow the variable element to reference these, either with target="#id", or a new attribute idref.

Proposal 3: Changing algorithm parameters

Problem: We might want to vary algorithm parameters over a repeated task.
Proposal: If algorithmParameter gains an (optional) id, the referencing scheme proposed earlier suffices.

Example: a timecourse simulation implemented using a repeated task.

```
modelTask id="singlestep"
  simulationReference to
    oneStep with algorithmParameter id="solveTo"

repeatedTask id="timecourse" resetModel="false"
  uniformRange id="time" start="0.0" end="10.0"
  setValue target="#solveTo" range="time"
  subTask task="singlestep"
```

Proposal 4: Handling n-dimensional data

- With nesting of tasks, results may have arbitrary dimensionality
- Standard MathML has no facilities for dealing with such data types
- We can define new `csymbol`s for operations such as extracting sub-arrays, creating new arrays, explicit maps, and 'folds' (see last year's talk)
- This should ideally be done in concert with NuML

Looking further ahead

- Units conversions
  - Requires a units standard across modelling languages

Looking further ahead

- Units conversions
  - Requires a units standard across modelling languages
- Protocol libraries

Looking further ahead

- Units conversions
  - Requires a units standard across modelling languages
- Protocol libraries
- Dealing with irregular data
  - e.g. arising from cell birth and death

Conclusions

- We believe the concepts described above from our protocol language would contribute more widely useful functionality to SED-ML
- Please discuss!
- Various strands of ongoing work
  - Encoding protocols for cardiac electrophysiology, cell-based Chaste, immunology, synthetic biology, ...
  - Developing new protocol concepts as needed by applications
  - Developing SED-ML extension proposals
  - Improving implementation, especially user-friendliness
  - Textual syntax for protocol language

Acknowledgments

Chaste team
Alan Garny, Steven Niederer, Mark Slaymaker
Reference publication: Prog Biophys Mol Biol 107:11-20, 2011
Web site: `https://chaste.cs.ox.ac.uk/cgi-bin/trac.cgi/wiki/FunctionalCuration`


Chaste

EPSRC Engineering and Physical Sciences Research Council

Microsoft Research 2020 SCIENCE

SEVENTH FRAMEWORK PROGRAMME