

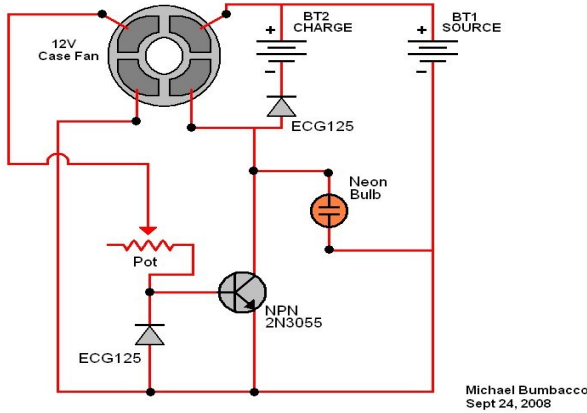
Proposed structure for NeuroML 2.0 and LEMS

- Motivation for LEMS
- Defining Component Types and Components
- Structure of models
- Interpreter
- Under development
- Open issues

Motivation

- From last year's report:
 - *“Discussion focused around two main topics: a possible structure for a more modular and flexible synapse specification; and the wide range of synaptic phenomena that might or might not be expressible in such a structure.”*
 - *“... would need to express a wide range of possible behaviors and could include kinetic scheme elements as for the channel specification, state variables governed by differential equations or reaction networks expressed as SBML”*
- Essentially, how can you keep the clarity and simplicity of domain specific “top level elements”, but also allow it to be easily extended with new types of component?

Electrical Circuit



Shared understanding
Not restated every time

Kirchoff's
Laws
etc, etc...



$$I_1 + I_2 + I_3 = 0$$

$$I_4 - I_3 + I_5 = 0$$

Etc

Abstracting Kirchoff's laws as shared knowledge allows a mode to be communicated at the diagram rather than equation level.

(eg) Hodgkin Huxley ion channel model



+



$$I_{Na} = 32 \cdot m^3 \cdot h \cdot i \cdot (v - 55)$$

$$m_{\infty} = \alpha_m / (\alpha_m + \beta_m); \quad \tau_m = 0.5 / (\alpha_m + \beta_m)$$

$$\alpha_m = 0.4(v + 30) / (1 - \exp(-(v + 30)/7.2))$$

$$\beta_m = 0.124(v + 30) / (\exp((v + 30)/7.2) - 1)$$

$$h_{\infty} = 1 / (1 + \exp((v + 50)/4))$$

$$\tau_h = 0.5 / (\alpha_h + \beta_h)$$

$$\alpha_h = 0.03(v + 45) / (1 - \exp(-(v + 45)/1.5))$$

$$\beta_h = 0.01(v + 45) / (\exp((v + 45)/1.5) - 1)$$

$$i_{\infty} = (1 + b_i \exp((v + 58)/2)) / (1 + \exp((v + 58)/2))$$

$$\tau_i = 3 \cdot 10^4 \beta_i / (1 + \alpha_i)$$

$$\alpha_i = \exp(0.45(v + 60))$$

$$\beta_i = \exp(0.09(v + 60))$$

14. Hightower et al.

and dissociation rates of active conductances in the distal portion of the dendritic tree (Brenner et al., 1984). In different brain regions, such as the visual cortex (Rockland and Vign, 1988) and hippocampus (Johnson and Axtell, 1990), this area of pyramidal neurons receives direct or indirect feedback projections. The elucidation of the membrane characteristics that could modulate such local signals may thus shed light on the operating principles of several brain circuits. In this article, we have shown how the β_{Ca} may play a key role in these processes.

Appendix

In the following we present the fit to the current, i_{Ca} , in nA, using constants in $\mu s/cm^2$ and α_i the time constant of a gating variable i in $\mu s/cm^2$. A temperature of 35°C was assumed for all calculations.

$$i_{Ca} = 32 \cdot m^3 \cdot h \cdot i \cdot (v - 55)$$

$$\alpha_m = 0.4(v + 30) / (1 - \exp(-(v + 30)/7.2))$$

$$\beta_m = 0.124(v + 30) / (\exp((v + 30)/7.2) - 1)$$

$$\alpha_h = 0.03(v + 45) / (1 - \exp(-(v + 45)/1.5))$$

$$\beta_h = 0.01(v + 45) / (\exp((v + 45)/1.5) - 1)$$

$$i_{\infty} = (1 + b_i \exp((v + 58)/2)) / (1 + \exp((v + 58)/2))$$

$$\tau_i = 3 \cdot 10^4 \beta_i / (1 + \alpha_i)$$

$$\alpha_i = \exp(0.45(v + 60))$$

$$\beta_i = \exp(0.09(v + 60))$$

$k_1 = 0.5$ in the rapid time scale, $k_2 = 0.8$ in the slow, and $k_3 = 1$ elsewhere.

If $\tau_m < 0.02$, then $\tau_m = 0.02$ ms.
If $\tau_h < 0.5$, then $\tau_h = 0.5$ ms.
If $\tau_i < 10$, then $\tau_i = 10$ ms.

$i_{Ca} = 10 \cdot i \cdot (v + 90)$
 $\tau_m = 1/(1 + \alpha_m); \quad \tau_h = 30 \beta_h / (1 + \alpha_h)$
 $\alpha_m = \exp(-0.11(v - 17))$
 $\beta_h = \exp(-0.08(v - 17))$

If $\tau_m < 2$, then $\tau_m = 2$ ms.
 $i_{Ca} = 32 \cdot m^3 \cdot h \cdot i \cdot (v - 55)$
 $\tau_m = 0.5 / (\alpha_m + \beta_m)$
 $\alpha_m = 0.4(v + 30) / (1 - \exp(-(v + 30)/7.2))$
 $\beta_m = 0.124(v + 30) / (\exp((v + 30)/7.2) - 1)$
 $\alpha_h = 0.03(v + 45) / (1 - \exp(-(v + 45)/1.5))$
 $\beta_h = 0.01(v + 45) / (\exp((v + 45)/1.5) - 1)$
 $i_{\infty} = (1 + b_i \exp((v + 58)/2)) / (1 + \exp((v + 58)/2))$
 $\tau_i = 3 \cdot 10^4 \beta_i / (1 + \alpha_i)$
 $\alpha_i = \exp(0.45(v + 60))$
 $\beta_i = \exp(0.09(v + 60))$

If $\tau_m < 0.1$, then $\tau_m = 0.1$ ms.
If $\tau_h < 2$, then $\tau_h = 2$ ms.

Acknowledgments

This work was supported in part by the NATO CNR Senior Fellowship Programme (MM), MH grants MH44474 and MH48452, the Shodor Foundation, and the Human Frontiers Science Program (DI). MM thanks S. Pappalardo for technical assistance.

What not to do...

```

<variable name="F" units="coulomb_per_mole" initial_value="96845" public_interface="out"/>
<variable name="Cm" units="picoF" initial_value="7"/>
<variable name="time" units="millisecond" public_interface="in"/>
<variable name="i_Ca_L" units="picoA" public_interface="in"/>
<variable name="i_Ca_T" units="picoA" public_interface="in"/>
<variable name="i_K_DR" units="picoA" public_interface="in"/>
<variable name="i_K_Ca" units="picoA" public_interface="in"/>
<variable name="i_leak" units="picoA" public_interface="in"/>
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <apply>
    <eq/>
    <apply>
      <diff/>
      <bvar>
        <ci>time</ci>
      </bvar>
      <ci>V</ci>
    </apply>
    <apply>
      <divide/>
      <apply>
        <minus/>
        <apply>
          <plus/>
          <ci>i_Ca_L</ci>
          <ci>i_Ca_T</ci>
          <ci>i_K_DR</ci>
          <ci>i_K_Ca</ci>
          <ci>i_leak</ci>
        </apply>
        <ci>Cm</ci>
      </apply>
    </apply>
  </math>
</component>

```

```

<component xmlns="http://www.cellml.org/cellml/1.0#" name="L_type_calcium_current">
  <variable name="i_Ca_L" units="picoA" public_interface="out"/>
  <variable name="phi_Ca" units="millivolt_millimolar" public_interface="out"/>
  <variable name="g_Ca_L" units="nanoS_per_millimolar" initial_value="9"/>
  <variable name="time" units="millisecond" public_interface="in" private_interface="out"/>
  <variable name="V" units="millivolt" public_interface="in" private_interface="out"/>
  <variable name="V_tau" units="millivolt" public_interface="in" private_interface="out"/>
  <variable name="k_tau" units="millivolt" public_interface="in" private_interface="out"/>
  <variable name="R" units="joule_per_kilomole_kelvin" public_interface="in"/>
  <variable name="T" units="kelvin" public_interface="in"/>
  <variable name="F" units="coulomb_per_mole" public_interface="in"/>
  <variable name="Ca_e" units="millimolar" public_interface="in"/>
  <variable name="Ca_i" units="millimolar" public_interface="in"/>
  <variable name="m_L" units="dimensionless" private_interface="in"/>
  <math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
      <eq/>
      <ci>i_Ca_L</ci>
      <apply>
        <times/>
        <ci>g_Ca_L</ci>
        <apply>
          <power/>
          <ci>m_L</ci>
          <cn cellml:units="dimensionless">2</cn>
        </apply>
        <ci>phi_Ca</ci>
      </apply>
    </math>
  </math>
  <apply>
    <eq/>
    <ci>phi_Ca</ci>
    <apply>
      <divide/>
      <apply>

```

```

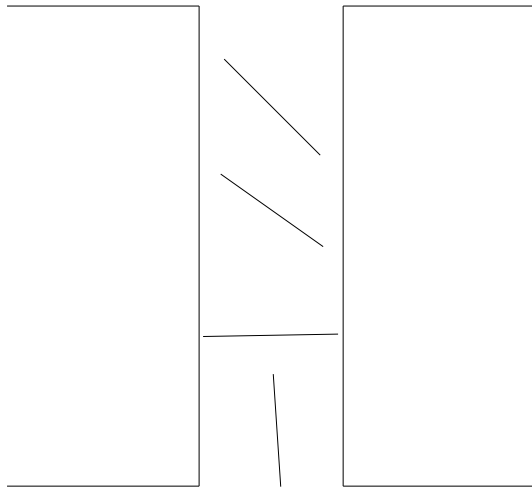
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="leak_current" component_2="environment"/>
  <map_variables variable_1="time" variable_2="time"/>
</connection>
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="ER_calcium" component_2="environment"/>
  <map_variables variable_1="time" variable_2="time"/>
  <map_variables variable_1="V_cell" variable_2="V_cell"/>
</connection>
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="cytosolic_calcium" component_2="environment"/>
  <map_variables variable_1="time" variable_2="time"/>
  <map_variables variable_1="V_cell" variable_2="V_cell"/>
</connection>
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="membrane" component_2="L_type_calcium_current"/>
  <map_variables variable_1="V" variable_2="V"/>
  <map_variables variable_1="i_Ca_L" variable_2="i_Ca_L"/>
  <map_variables variable_1="R" variable_2="R"/>
  <map_variables variable_1="F" variable_2="F"/>
  <map_variables variable_1="T" variable_2="T"/>
</connection>
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="membrane" component_2="T_type_calcium_current"/>
  <map_variables variable_1="V" variable_2="V"/>
  <map_variables variable_1="i_Ca_T" variable_2="i_Ca_T"/>
</connection>
<connection xmlns="http://www.cellml.org/cellml/1.0#">
  <map_components component_1="membrane" component_2="voltage_sensitive_K_current"/>
  <map_variables variable_1="V" variable_2="V"/>
  <map_variables variable_1="i_K_DR" variable_2="i_K_DR"/>
  <map_variables variable_1="R" variable_2="R"/>
  <map_variables variable_1="F" variable_2="F"/>
  <map_variables variable_1="T" variable_2="T"/>
</connection>

```

3 of about 30 pages that constitute the CellML representation of a fairly simple HH based cell model.

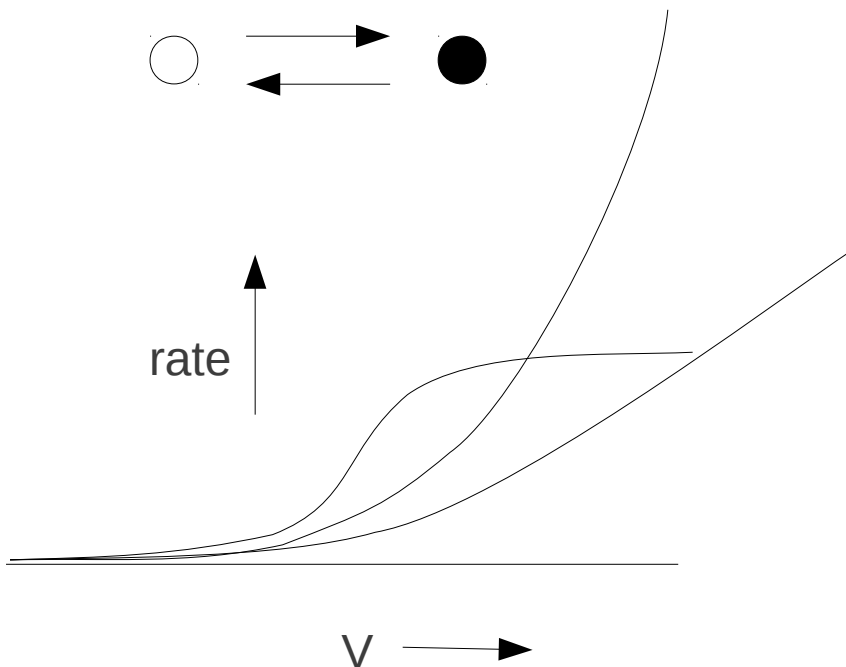
Says $C \frac{dV}{dt} = \sum i$,
 ... but without using \sum
 All the currents are enumerated explicitly

What actually is the model?



Original HH model and almost all derivatives have:

- Serial independent gates
- Gate opening and closing governed by a rate expression with one of three forms:
 - $\exp(V)$
 - $\exp(V) / (1 + \exp(v))$
 - $V / (1 - \exp(-V))$
- Each rate expression has three parameters – V scale, Rate scale, V offset



NeuroML exploits this structure to allow concise expression of HH style models: (this is actually the PSICS form, but its pretty much equivalent)

```
<KSChannel id="HH_Na" permeantIon="Na" gSingle="20pS">
  <KSComplex id="m" instances="3">
    <ClosedState id="c"/>
    <OpenState id="o"/>
    <ExpLinearTransition from="c" to="o" rate="1.per_ms" midpoint = "-40.mV" scale="10mV"/>
    <ExpTransition from="o" to = "c" rate="4.per_ms" midpoint="-65.mV" scale="-18mV"/>
  </KSComplex>
  <KSComplex id="h">
    <ClosedState id="c"/>
    <OpenState id="o"/>
    <ExpTransition from="c" to="o" rate="0.07per_ms" midpoint="-65.mV" scale="-20.mV"/>
    <SigmoidTransition from="o" to="c" rate="1per_ms" midpoint="-35mV" scale="10mV"/>
  </KSComplex>
</KSChannel>

<KSChannel id="HH_K" permeantIon="K" gSingle="20pS">
  <KSComplex id="n" instances="4">
    <ClosedState id="c"/>
    <OpenState id="o"/>
    <ExpLinearTransition from="c" to="o" rate="0.1per_ms" midpoint = "-55.mV" scale="10mV"/>
    <ExpTransition from="o" to = "c" rate="0.125per_ms" midpoint="-65.mV" scale="-80mV"/>
  </KSComplex>
</KSChannel>
```

But this depends on external definitions for the element types.
What if we want to express the whole lot from scratch?

Need a way to express -

- The structures shared by many models, once
- For a particular model, just the parts unique to that model, with a reference to the shared structure

Without editing the schema/specification every time – neuroscience models are just too diverse.

Similar principle to:
Modelica (mechanical systems)
VHDL (electronic design)
NineML (networks and more)
...and others

Syntactic fiddles:

“<XXX .../>”

is shorthand for

“<Component type='XXX'/>”

“a='value unit'”

is shorthand for

“<value parameter='a'
size='val'
unit='unit'/>”

```
<Include file="hhchannel.xml" />  
<Unit symbol="mV" dimension="voltage" powTen="-3" />  
<Unit symbol="per_ms" dimension="per_time" powTen="3" />  
<Unit symbol="pS" dimension="conductance" powTen="-12" />
```

```
<HHChannel id="na" conductance="20pS">  
  <HHGate id="m" power="3">  
    <Forward type="HHExpLinearRate" rate="1.per_ms" midpoint="-40mV" scale="10mV" />  
    <Reverse type="HHExpRate" rate="4per_ms" midpoint="-65mV" scale="-18mV" />  
  </HHGate>  
  <HHGate id="h" power="1">  
    <Forward type="HHExpRate" rate="0.07per_ms" midpoint="-65.mV" scale="-20.mV" />  
    <Reverse type="HHSigmoidRate" rate="1per_ms" midpoint="-35mV" scale="10mV" />  
  </HHGate>  
</HHChannel>
```

```
<HHChannel id="k" conductance="20pS">  
  <HHGate id="n" power="4">  
    <Forward type="HHExpLinearRate" rate="0.1per_ms" midpoint="-55mV" scale="10mV" />  
    <Reverse type="HHExpRate" rate="0.125per_ms" midpoint="-65mV" scale="-80mV" />  
  </HHGate>  
</HHChannel>
```

Desired content of the
top layer of the model
specification

```

<Dimension name="voltage" m="1" l="2" t="-3" i="-1" />
<Dimension name="time" t="1" />
<Dimension name="per_time" t="-1" />
<Dimension name="conductance" m="-1" l="-2" t="3" i="2" />
<Dimension name="capacitance" m="-1" l="-2" t="4" i="2" />
<Dimension name="current" i="1" />

<ComponentType name="HHRate">
  <Parameter name="rate" dimension="per_time" />
  <Parameter name="midpoint" dimension="voltage" />
  <Parameter name="scale" dimension="voltage" />
  <Behavior>
    <IndependentVariable name="v" dimension="voltage" />
    <DerivedVariable name="r" dimension="per_time" />
  </Behavior>
</ComponentType>

<ComponentType name="HHExpRate" extends="HHRate">
  <Behavior inherit="variables">
    <DerivedVariable name="r" value="rate * exp((v - midpoint)/scale)" />
  </Behavior>
</ComponentType>

<ComponentType name="HHSigmoidRate" extends="HHRate">
  <Behavior inherit="variables">
    <DerivedVariable name="r" value="rate / (1 + exp(0 - (v - midpoint)/scale))" />
  </Behavior>
</ComponentType>

<ComponentType name="HHExpLinearRate" extends="HHRate">
  <Behavior inherit="variables">
    <DerivedVariable name="x" value="(v - midpoint) / scale" />
    <DerivedVariable name="r" value="rate * x / (1 - exp(0 - x))" />
  </Behavior>
</ComponentType>

```

```

<ComponentType name="HHGate">
  <Parameter name="power" dimension="none" />
  <Child name="Forward" type="HHRate" />
  <Child name="Reverse" type="HHRate" />
  <Behavior>
    <IndependentVariable name="v" dimension="voltage" />
    <StateVariable name="q" dimension="none" />
    <ExternalVariable name="rf" dimension="per_time" select="Forward/r" />
    <ExternalVariable name="rr" dimension="per_time" select="Reverse/r" />
    <TimeDerivative variable="q" value="rf * (1 - q) - rr * q" />
    <DerivedVariable name="fcond" dimension="none" value="q^power" />
  </Behavior>
</ComponentType>

```

```

<ComponentType name="HHChannel">
  <Parameter name="conductance" dimension="conductance" />
  <Children name="gates" type="HHGate" min="0" max="4" />
  <Behavior>
    <IndependentVariable name="v" dimension="voltage" />
    <ExternalVariable name="gatefeff" dimension="none"
      select="product(gates[*]/fcond)" />
    <DerivedVariable name="g" value="conductance * gatefeff" />
  </Behavior>
</ComponentType>

```

With these definitions in place, most of the ion channel models in the modeling literature can be expressed in 10 or 15 lines of XML.

What is there so far

- Point process models
 - Dimensions, units, parameters, state variables
 - First order ODEs
 - Event generation and handling
- Hierarchical structures
 - Eg, “a channel has n gates; a gate has children for the forward and reverse rates”
- References between elements
 - Eg “Channels is permeable to Na; the reversal potential of Na in this simulation is 60mV → so, channel reversal potential is 60mV ”
- Elements for defining simulations and outputs
- Expressions with paths and predicates operating on components
 - Selecting and filtering sets of instances
 - Selecting and filtering sets of instance pairs
 - Adding new instances based on sets
 - Per-instance properties

Under development

- Control of model “instantiation”
 - Needed for extended cells and networks
- Selection operators and paths across an “instantiated” model
 - Operating on synapses on a cell, or cells in a population
- Better structures for representing component hierarchies and behaviors

Beyond the horizon

- Spatial structure and PDEs

What can you do with it?

- Reference interpreter - will build and run models defined in LEMS.
- Can retrofit existing NeuroML element types with LEMS component type definitions
- Tools have a choice of recognizing the component type, or processing the LEMS definition

What you can't do

It is tempting to suggest that the behavior definitions say something about the semantics of a model, but they really don't help much. Still need annotations and documentation to express the significance behind a component type.

Component type definitions

- **Dimensions and Equations**
- Hand-written in XML
- Possibly machine generated in some cases
- Concise, and relatively few of them
- Need referencing and selection mechanisms operating across types.
 - *eg to say the relative conductance of a channel is the product of the relative conductances of its gates*

Component definitions

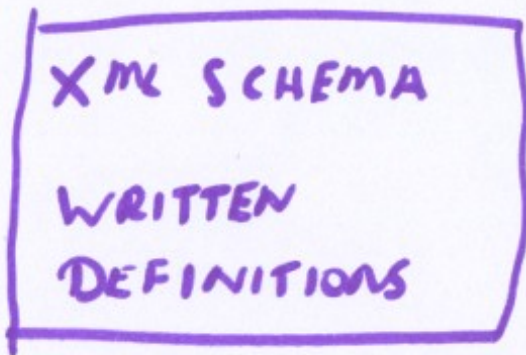
- **Units and Parameter values**
- Can be hand-written in XML by a modeler, possibly generated
- Need references and selections across a model
 - *Eg to specify the channel model to use for a particular collection of channels*
- **Also** need references and selections across the (hypothetical) instantiated model
 - *Eg to select cells for a particular connectivity pattern (**targets don't exist in the XML**)*

Hypothetical fully-expanded model

- **Simulator state as nesting structure and dimensional quantities**
- One entry for every cell in a network, every state variable in a simulation.
- *Could* be used to simulate a model (but not a very efficient way of doing it)
- In general, may never be generated
- But can still write xpath or equivalent to operate over it

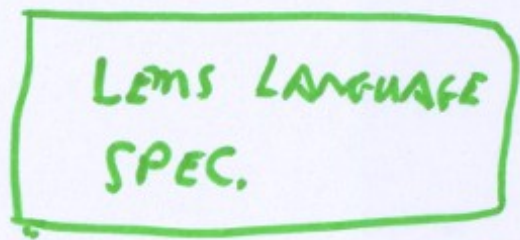
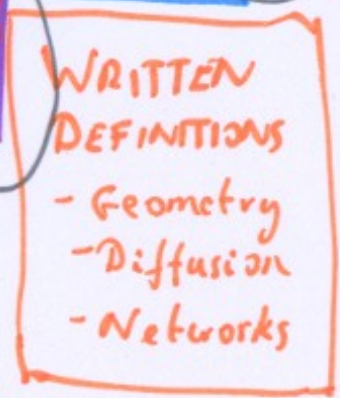
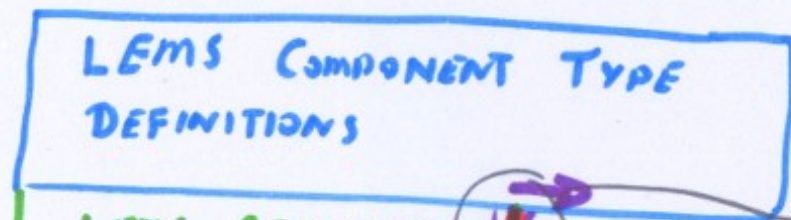
NEURO ML 1.

PRIMARY ELEMENT
TYPES



NEURO ML 2.

PRIMARY TYPES ↔ USER-DEFINED
TYPES



Risk of inventing
UD types even
where primary
ones exist **1**

Standardization
process
UD Type → Primary **2**

3 Getting more
components
fully expressed
with LEMS
Behaviors