CNO Documentation

Yann Le Franc, PhD
January 2013

Welcome to the Computational Neuroscience Ontology documentation.

Introduction

With the increased number of computational neuroscience studies comes the problem of reproducibility and reusability of either the whole or part of a model. The diversity of model implementations makes it unlikely that one laboratory can easily reproduce the results obtained by another group, even if the model is deposited in an openly accessible database such as ModelDB (http://senselab.med.yale.edu/ModelDB/default.asp). To solve this problem, the community needs to define and adopt standards for model description that will make model sharing more efficient. These standards should propose a common format of representation, independent of any given simulator that could be imported or exported by the different simulators.

To address this problem, standardized languages have been developed by and for the community, such as NeuroML (http://www.neuroml.org), PyNN (http://neuralensemble.org/trac/PyNN) and NineML (http://software.incf.org/software/nineml).

Why CNO?

Although these languages enable software interoperability and therefore model reuse and reproducibility, they lack semantic information that would facilitate efficient model retrieval.

The creation of XML standards such as NineML enables semantic web technologies. These recent technological developments aim at providing "meaning" in the web content that will be usable by software (i.e., machine-readable). This "semantic" information is contained in ontologies.

Ontologies are formal models of knowledge in a particular domain, and are composed of classes that represent concepts defining the field as well as logical relations that link these concepts together (for an example linked to Neurosciences, read Larson and al., 2009). These machine-interpretable solutions, using the mathematical framework of first-order logic, allow computers to make inferences on datasets, to reveal particular relations between different types of data and to provide a more intuitive and efficient information retrieval for domain-experts.

System Biology faced similar issues (De Schutter, 2008) and developed several tools to represent and annotate system biology models such as the XML standard SBML (http://sbml.org/Main_Page) and the System Biology Ontology (http://www.ebi.ac.uk/sbo/main/).
Semantic information from SBO is directly embedded into SBML with specific tags and is used in Biomodels database (http://www.ebi.ac.uk/biomodels-main/) to allow for the retrieval of major information concerning the model such as parameter values, equations used in the models. Few neurosciences model exist in Biomodels database, I invite you to visit the description of the Hodgkin and Huxley model to have an idea of

the type of information we can get with such representations: http://www.ebi.ac.uk/biomodels-main/BIOMD0000000020.
If you wish to know more, a recent review (Courtot M. and al., 2011) describes the different controlled vocabulary developed in System Biology and how they can be used.

The addition of similar information about models in ModelDB would be a real added value for the community. As part of the Senselab database, ModelDB has already made such integration with the Neurosciences ontologies (NIFSTD, http://www.neuinfo.org/) and other biomedical ontologies (Samwald, 2010). Despite the existence of resources like NIFSTD ontologies for Neurosciences, ModelDB ontologies or SBO for annotating system biology model, there is still little overlap with terms used in Computational Neurosciences.

Therefore, in the context of the INCF Multi-Scale Modeling (MSM) program, we have developed a Computational Neuroscience Ontology or CNO, to annotate Computational Neuroscience models described with NineML and other structured model description languages. CNO has been designed as a tool for the community and for this purpose has been thought to be fully interoperable with the semantic infrastructure of ModelDB and the NIFSTD ontologies providing neuroscience specific information.

What is CNO?

CNO is a controlled vocabulary composed of classes representing general concepts related to computational neuroscience. These concepts are organized in a hierarchy of concepts and related to each other using logical relations. One of the main relations is the hierarchical relation named “is a” relation.
Knowing the relations between classes allows us to create simple sentences such as: “a model is a thing”, “a leaky integrate-and-fire is a model”, “voltage is a elementary model component”, …

The first step for CNO development was to create a list of relevant terms and concepts from textbooks and from the different expertise of the MSM Task Force members.
Based on this list of terms, we developed a first version of the Computational Neuroscience Ontology (CNO), using the OWL-DL standard proposed by W3C (http://www.w3.org/TR/owl-features/). This particular form of the standard enables the use of inferences.

To allow for maximal interoperability with other biomedical ontology, the design of CNO follows some of the recommendations of the Open Biological and Biomedical Ontologies (OBO) community (http://www.obofoundry.org/).

CNO classes and relations have unique identifiers that allow unambiguous annotation of digital resources. These unique identifiers are of the form cno_XXXXXXX (7digits) and allow the creation of unique URI that references this particular class. The URI result from the concatenation of the namespace defined by CNO (http://purl.org/incf/ontology/Computational_Neurosciences/cno_alpha.owl#) and the unique identifier of the class (e.g. cno_0000066).

Descriptive information of terms such as human-readable label or definition has been added using annotation properties from the Dublin Core (http://dublincore.org/), SKOS (Simple Knowledge Organization System, http://www.w3.org/2004/02/skos/) and OBO (http://www.obofoundry.org/).

For each term or class, we provided 1- a label, which corresponds to a human-readable name, 2- a human-readable definition. Definitions were constructed using the particular format of Aristotelian definition, as recommended by OBO foundry. A class is defined by providing its genus (parent class) and its differentia, which tells what characterizes the Subclass within the parent class. When necessary we added additional information such as links to relevant resources used for creating the definition using the following SKOS annotation properties "definingCitationURI" and "definingCitationId".

A concrete example being often better than long explanations, here is one just for you. Let's look at the class describing the Leaky Integrate-and-Fire model:
Name (URI):
http://purl.org/incf/ontology/Computational_Neurosciences/cno_alpha.owl#cno_0000066
Label: "leaky integrate-and-fire"
Definition: a defined model (parent class) *that is an integrate-and-fire model with a leak component added to the membrane potential, reflecting the diffusion of ions that occurs through the membrane when some equilibrium is not reached in the cell (differentia).*
definingCitationURI:
http://en.wikipedia.org/wiki/Biological_neuron_model#Leaky_integrate-and-fire

CNO has been developed using the ontology editor Protégé (version 3.4.4 build 579) for the version up to CNO v0.2.4. For the development of CNO version 0.5, we use Protégé version 4.1.0 build 239.

The core of CNO is currently (January 2013) composed of 221 terms and 2 datatype properties.
As Computational Neuroscience model are very diverse, CNO is still in development to integrate the most relevant concepts for annotating as many Computational Neuroscience models as possible. However, CNO aims to be a tool for the community and therefore contribution of the community would be highly valuable to bring CNO to a more mature distribution.
In order to ease the participation of the community, CNO will be imported into Neurolex (http://neurolex.org/wiki/Main_Page) which provide a platform to 1- access the existing terms, 2- add new terms and 3- propose changes to existing terms (updating definition, associated resources, synonyms, ...).

If you are interested in contributing to this ontology, please contact Yann Le Franc.

Scope of CNO

CNO aims at providing a controlled vocabulary to annotate Computational Neurosciences models and describe the following aspects of models:

1- the model itself (parameter, equation, formalism, ...)
2- the model implementation (parameter values, algorithms, simulation software, ...).
3- the external resources that are related to the model such as publication, ModelDB accession number, ...

For the first versions, CNO doesn't aim to provide a detailed description for emerging functional properties of the models or mathematical functions and operators.

Where to find CNO?

An alpha version of CNO is currently accessible on bioportal, the biomedical ontology portal provided by NCBO (http://bioportal.bioontology.org/ontologies/3003).

You can download the owl file from the GitHub repository and open it as a text file to visualize the underlying owl structure. You can use Protégé 4.1 (http://protege.stanford.edu/) to navigate in the hierarchy and look at the annotations of the terms such as definition, label.

CNO internal model

To represent the large diversity of models, we considered that a given **model** is the results of the aggregation of different building blocks called **model component** (figure 1). These components should be general enough to allow the construction of any type of models de novo or based on previous models. This allows us to represent the typical case where a new model is build from an existing model with the addition of a new component or the change of a particular component (e.g. using a different cell model, adding a different type of synapse models or adding a model of a particular ionic conductance).
The figure 1 below shows the classes in CNO that were created to implement this particular model.

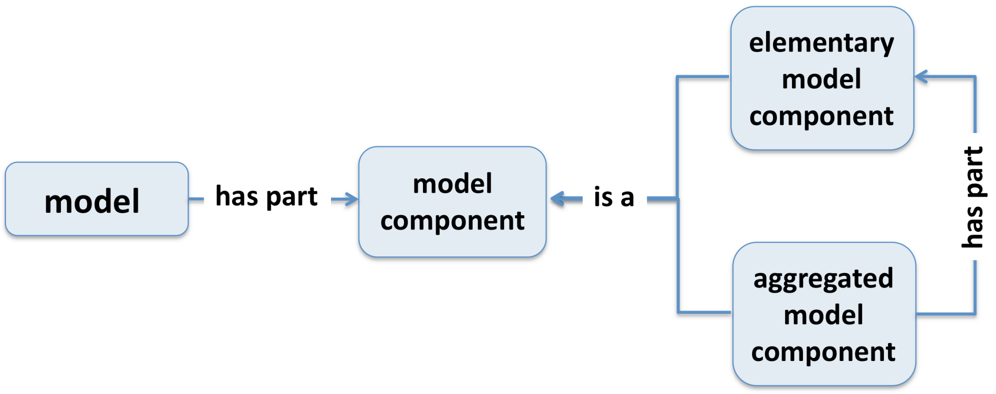

Figure 1: presentation of CNO internal model.

The **model** class is an upper class that represents all the models that could be encountered. Any model that will be annotated is then an instance of this class.
The **model component** class is also an upper class representing the set of components that can be used to build a model. It can be further subdivided into **elementary model component** and **aggregated model component**.
The **elementary model component** class includes the smallest components that can be used in models e.g. voltage, current, indices, kinetic rates, amplitude, …
The terms contained in this particular class have been for a large part imported from SBO. The procedure used to import these terms into CNO is described later (see "Importing multiple terms from multiple external ontologies."). These elementary components can be used as variable or as a parameter in the model.
The **aggregated model component** class represents more complex components such as ionic current model, morphology, network layout which result themselves from the aggregation of elementary components and mathematical operators or functions. It is important to mention that the scope of this ontology is not supposed to cover such mathematical functions and operators.

The link between the model and its component is made through the Object Property "*has part*" defined in the Relation Ontology (http://obofoundry.org/ro/).

This simple model seems to be efficient to represent any model structure. However a model has also some other important characteristics such as its description (publication, XML file, ModelDB entry, ...) or functional properties such as bursting, oscillation, synchronization emerging from the model structure.
To account for these important notions we added two upper classes as shown in figure 2.

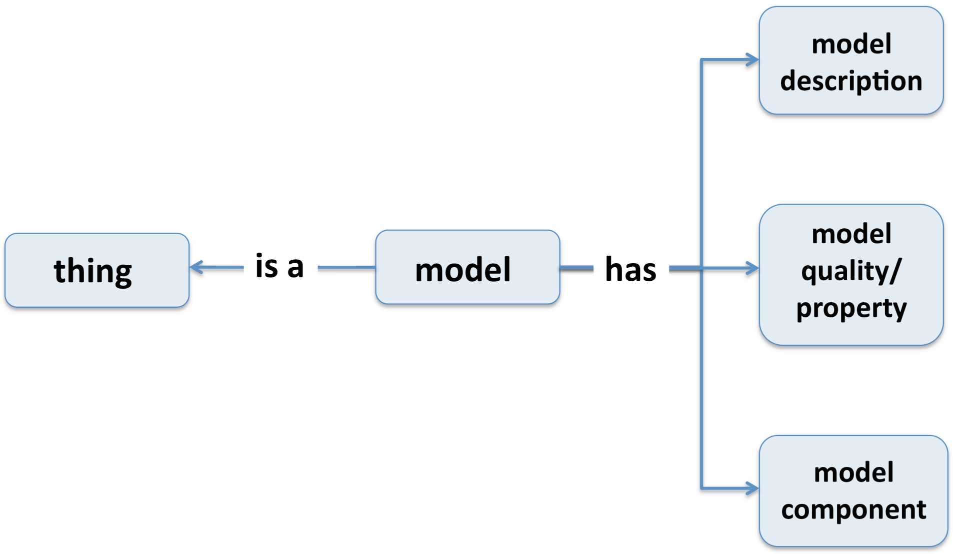

Figure 2: attaching other information to the model

The **model description** class represents the different possible description for the model: publication, a particular implementation, a XML representation, ...
As this class is rather broad, we only considered the PMID and the ModelDB accession number in CNO version 0.5.
The **model quality**/property class represents the properties emerging from the particular structure of the model. This class is rather complex and broad, we chose to focus on populating it in later versions of CNO. To have an example of the type of concept that should be included, you can look at the list proposed on ModelDB (http://senselab.med.yale.edu/ModelDB/FindByConcept.asp).

Finally, we created two additional upper classes: **defined model** and **defined model type** (figure 3).
The **defined class** represents the set of models that we could consider as "standard" models e.g. Leaky Integrate-And-Fire, Izhikevitch model, Markram and Tosdyks model, Fitzhugh Nagumo, Morris-Lecar, Hodgkin and Huxley and many more. These different models have well known characteristics and are often used as a base for building up different models.
The **defined model type** is a classification of the different models. It represents the different general types of models such as artificial neural network, spiking network, point process, ...

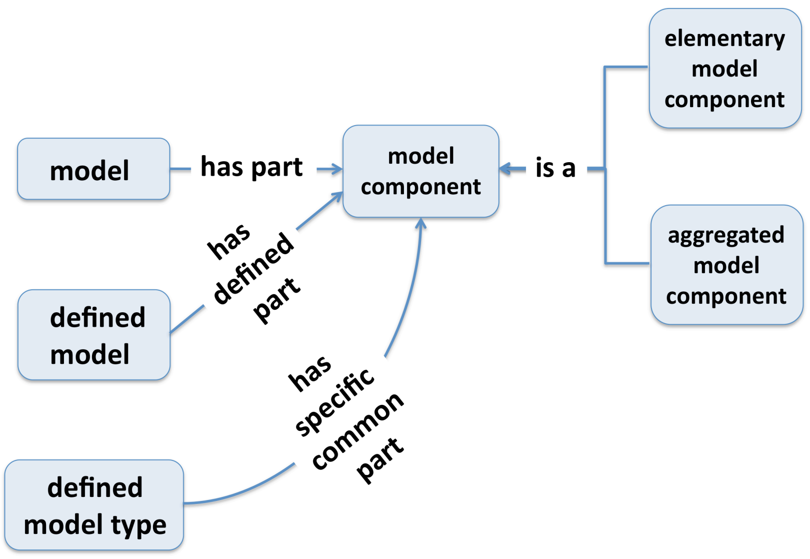

Figure 3: presentation of the defined classes

The **defined model** class is linked to the **model component** class using the relation *"has part"*. These relations are refined with restrictions to express that one particular model has some defined components.
To give an example, the current definition of a Leaky Integrate-and-Fire describes the model with the following components: a fixed spiking threshold, a stimulation current or a synaptic current, a point morphology, a refractory period and a leak current.
We are planning to create these restrictions in next versions of CNO.

Work done on CNO version 0.5:

The main objective of this update of CNO is to propose an ontology that 1- could be integrated into the Neuroscience ontologies developed by the Neuroscience Information Framework (NIFSTD ontologies) and interoperable with other Biomedical Ontologies and 2- includes terms from existing ontologies such as the System Biology Ontology (SBO), the Ontology for Biomedical Investigation (OBI) or the Information Artifact Ontology (IAO, http://code.google.com/p/information-artifact-ontology/).

As first step, we reconsidered all the definitions proposed in CNO version 0.2.4 and added most of the missing definitions. In the meantime, we refined the terminology and the internal CNO classes. These changes are documented in the Changelog.

The second step was to integrate CNO classes into the Basic Formal Ontology super classes (see below).

The third step was to import terms that are defined in other ontologies (see "Importing multiple terms from multiple external ontologies"). For this, we used the MIREOT format (http://obi-ontology.org/page/MIREOT) and the Ontofox service (http://ontofox.hegroup.org/).

Integration into the Basic Formal Ontology

The Basic Formal Ontology (BFO, http://www.ifomis.org/bfo) is an upper-level ontology that describes material entities bounded in space and time. BFO is currently used in several biomedical ontologies including the NIFSTD ontologies.

Although a model can be considered as a material entity through it implementation or the publication that describes it, models cannot really be bounded in space.

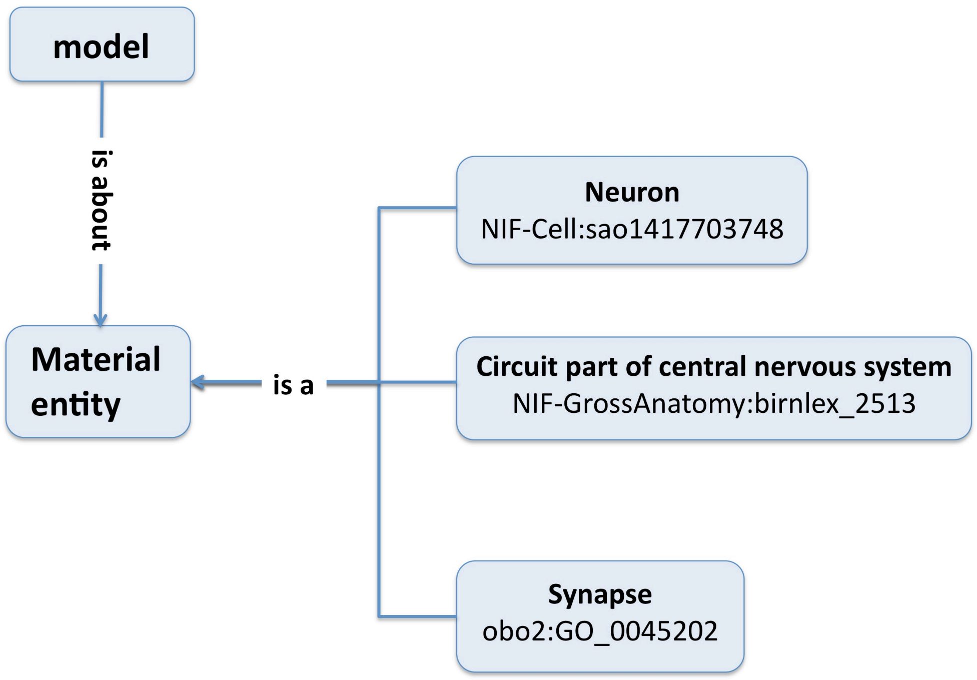

Figure 4: Integration into BFO: how to relate a mathematical model to the corresponding biological entity?

However, models aim to describe material entities (real neurons, synapses, or part of the brain). IAO has worked on a similar problem to deal with the representation of information content entities such as data label or document.
A particular relation, the "is about" relation has been created to link a **material entity** named an **information bearer** and **generically dependent continuant** named **information content entity**.

For the integration of CNO into BFO we used similar approach, considering that a **model** "is about" a **material entity**. Figure 5 shows the integration CNO upper classes into BFO.

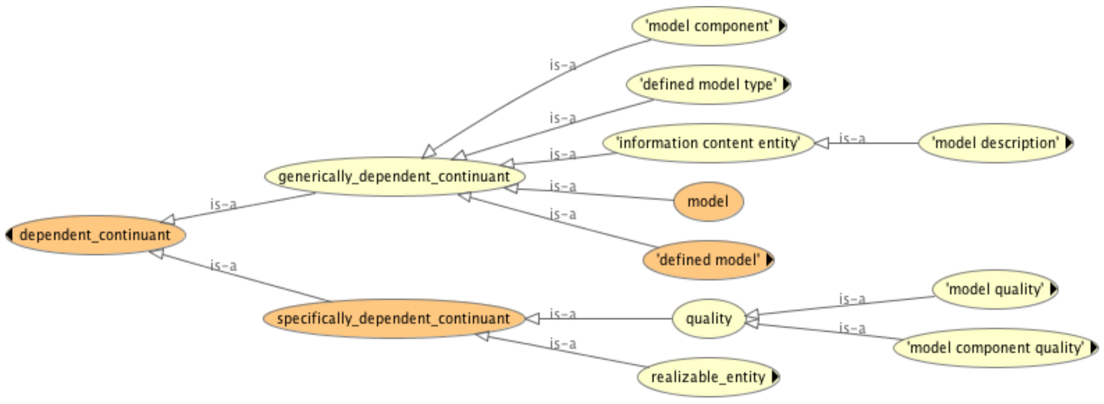

Figure 5: Integration of CNO upper classes into BFO

In BFO, a **material_entity** is an **independent continuant**. As we assume that a model depends on a material_entity, CNO classes are considered as **dependent_continuant**. By definition, a **dependent_continuant** is *"a continuant [snap:Continuant] that is either dependent on one or other independent continuant [snap:IndependentContinuant] bearers or inheres in or is borne by other entities"*.

As expressed in the definition, a **dependent_continuant** can be of two sorts: either a **generically_dependent_continuant** that depends on an **independent_continuant** bearer or a **specifically_dependent_continuant**, which inheres in or is born by other entities.

CNO classes describing the model and its structure are generically_dependent_continuants as they describe and therefore depend on material entities such as a neuron, a brain region, ionic channel, neuron subcellular entities, neuron morphology, ... In contrast, model qualities and role are specifically dependent continuant.

This sounds quite complicated so here is an example. Let's consider a neural network model with a recurrent connectivity pattern defined by a particular connectivity rule. The recurrent connectivity pattern is a quality of the model (as the red color would be for a tomato), which is born by a particular connectivity rule.
Therefore **connectivity rule** is a **generically_dependent_continuant** (which depends on the biological network connectivity) and the **recurrent** connectivity pattern is a **specifically_dependent_continuant** and more precisely a **quality**.

Specifically_dependent_continuants can be either a **quality** as we just saw or a **realizable_entity**. Realizable_entities can be a **role**, a **disposition** or a **function**.
In the design of CNO, we faced the issue of representing parameters and variables in the ontology. Parameters and variable represent physical qualities such as voltage for instance. In the design of a model, nothing prevents to have voltage as a parameter or a variable. This means that if we create two classes to represent model parameters and model variable, voltage should be a subclass of both classes.
To prevent this redundancy, we created the class **elementary model component** that will contain the references to physical qualities and allow for a particular elementary model component to be either a variable or a parameter depending on the model (e.g. the membrane voltage could be a variable in the sub-threshold equation of an IAF or could be a parameter in some other model).
For this, we considered that being a **parameter** or a **variable** is a **role** played by an elementary model component in the model structure (see figure 6).

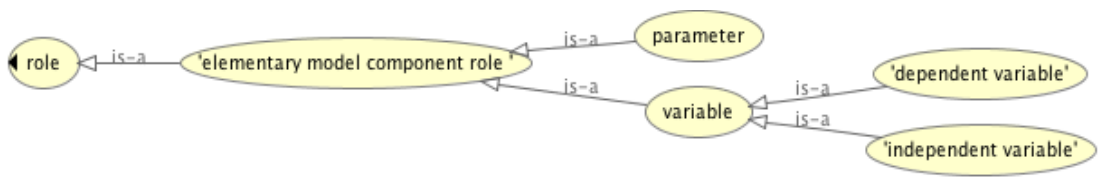

Figure 6: Using the structure of BFO to define parameters and variables

We thus created a subclass of role named **elementary model component role**, which contains the subclasses **parameter** and **variable**.

Importing multiple terms from multiple external ontologies.

One of the major principle in ontology design is reuse terms defined in other ontologies. To comply with this good practice, we started reusing terms already existing in other ontologies and we focused on the IAO ontology for terms related to model description and on SBO for terms describing model parameters.
To import terms from another ontology, you have two main possible options: 1- importing the whole external ontology or 2- importing selected terms from this ontology.
Although the option 1 seems to be the easiest, importing large ontologies into CNO could be troublesome and would add up terms that are not necessary in the context of CNO. Therefore we decided to import selected terms.

For this purpose, the Minimum Information to Reference an External Ontology Term or MIREOT format has been proposed (http://obi-ontology.org/page/MIREOT). To support such method of importing terms, a web interface, Ontofox (http://ontofox.hegroup.org/) has been designed to export selected terms using the MIREOT format.

We used Ontofox to extract the terms of interest from SBO and IAO (Xiang et al., 2010). For this, we created an input file with the support of the tutorial (http://ontofox.hegroup.org/tutorial/index.php).
This file is build to import multiple terms This file has been build to import multiple terms (for information regarding merging OntoFox input files for different source ontologies, see here: http://ontofox.hegroup.org/tutorial/index.php#merge) from SBO, IAO and OBI.

The input file is then submitted to the Ontofox web service, which then gather the requested information using dedicated SPARQL queries and concatenate the result into an owl file.

The output owl file containing the imported terms has been added in the folder /external_import and was named external.owl.

In order to keep track of the import, we also added the input file used to generate the owl file in the folder /external_import.

To include the external.owl ontology into CNO, we used a simple ontology import in Protégé.

To add new terms from other ontologies, we just need to add the necessary information into the input file and generate a new owl file.

References

De Schutter E. (2008) **Why are computational neuroscience and system biology so separate?**, PloS Computational Biology, May 30; 4(5):e1000078 (Review)

Larson S.D., Martone M.E. (2009) **Ontologies for Neuroscience: What are they and What are they Good for?** *Front Neurosci*, **3**(1):60-67.

Gardner D., Akil H., Ascoli G.A., Bowden D.M., Bug W., Donohue D.E., Goldberg D.H., Grafstein B., Grethe J.S., Gupta A. *et al* (2008) **The Neuroscience Information Framework: a data and knowledge environment for neuroscience**. *Neuroinformatics*, **6**(3):149-160.

Courtot M., Juty N., Knüpfer C., Waltemath D., Zhukova A., Dräger A., Dumontier M., Finney A., Golebiewski M., Hastings J., Hoops S., Keating S., Kell D.B., Kerrien S., Lawson J., Lister A., Lu J., Machne R., Mendes P., Pocock M., Rodriguez N., Villeger A., Wilkinson D.J., Wimalaratne S., Laibe C., Hucka M., Le Novère N. (2011) **Controlled vocabularies and semantics in systems biology**, Molecular System Biology, 7:543

Samwald M., Chen H., Ruttenberg A., Lim E., Marenco L., Miller P., Shepherd G., Cheung K.H. (2010) **Semantic Senselab: Implementing the vision of the Semantic Web in neuroscience.**, Artif. Intell. Med., Jan;48(1):21-8

Xiang Z., Courtot M., Brinkman R.R., Ruttenberg A., He Y. (2010) **Ontofox: web-based support for ontology reuse.**, BMX Research Notes, 3:175