COE

COE is an RDF/OWL ontology viewing/composing/editing tool built on top of the IHMC CmapTools concept mapping software suite. For more about CmapTools, see http://cmap.ihmc.us/.

With COE, you can:

- **import** OWL or RDF from RDF/XML files, or over the web from a URI, and display it in Cmap format, using graphical conventions to render OWL/RDF content in an intuitive form which is easy to learn.

- **Analyze** the ontology by viewing its triples in various formats, listing the concepts it uses, and finding concept relationships.

- **Edit** the ontology Cmap using CmapTools, creating new nodes and arcs by click-drag operations. Pieces of Cmaps can be cut and pasted from one ontology to another. All the editing and composing functions in CmapTools are also available in COE. COE also provides draggable templates for rapid composing of common ontology patterns, and drop-down contextual menus for selecting common link names.

- **Navigate** quickly through an ontology, rapidly finding concept instances, all triples containing a given concept, etc.; and also between ontologies. COE uses the base URI of a concept name to automatically locate, import and display the 'home' ontology of the concept, and autonavigates to the occurrence of the concept in that Cmap.

- **Save or publish** your Cmap as an image or an XML file.

- **Collaborate** with other COE users remotely in real time.

- **Record and re-play** your COE sessions for training or statistical purposes

- **Export** your Cmap to OWL/RDF in various formats, including RDF/XML, Turtle and N3.

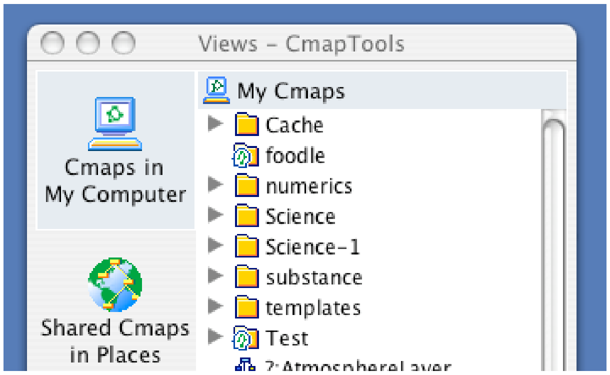
Getting started

Install COE by downloading it from http://cmap.ihmc.us/coe/beta/install.htm. COE is written in JAVA and will run on Windows, Apple or Sun platforms.

> NOTE. *Since COE is a version of CmapTools, it will not install on a machine on which a different version of CmapTools is installed. You will need to uninstall the earlier CmapTools version in order to use COE.*

The VIEWS window

On startup, COE displays a **Views** window.



This is the CmapTools access point to concept maps (including COEmaps) stored on your computer and elsewhere.

> TIP. *While working with COE, you should normally keep* **Cmaps in My Computer** *selected in this window, as many menu options are turned off when other selections are made. If nothing seems to be working properly and you cannot see a Views window, go to menu* **Window/Show Views***. A Views window will appear: check that* **Cmaps in My Computer** *is selected.*

Selecting **Shared Cmaps in Places** in this window provides access to several thousand public Cmap servers world-wide. However, most of the Cmaps in the world are not COEmaps; that is, they do not use the COE ontology conventions and would not export to useful RDF or OWL.

The Cmap server **IHMC Public Ontologies** is devoted to COEmaps; feel free to add a folder inside the **Users** folder there, if you would find this convenient, perhaps for file sharing with others. Or you may prefer to store your work locally. To open a Cmap from the menu, double-click on its name.

You can now: *create* a new blank COEmap; *open* a previously stored COEmap; or *import* an OWL or RDF/RDFS ontology and display it as a COEmap.

To **create** a new COEmap (make sure **Cmaps in My Computer** is selected in the Views window) choose menu **File/New Cmap**, or type control-N (mac: apple-N). A blank Cmap window will open.

To **open** a previously stored COEmap, locate it using the Views window, then double-click the name. A Cmap window will open displaying the COEmap.

To **import** a COEmap from a locally stored RDF or OWL file, choose menu **File/Import/ Ontology Map from XML File**, then use the search menu to find the file name on your computer's file system. A window of **import options** for expert users will open. Hit OK to use the recommended default settings. A Cmap window will open displaying the ontology as a COEmap.

To **import** a COEmap from a Web ontology, choose menu **File/Import/Ontology Map from URL**, then type or paste a URL into **Ontology URL** and hit OK. A window of **import options** for expert users will open. Hit OK. A Cmap window will open displaying the ontology as a COEmap.

> TIP. *Clicking on the small arrow tab at the bottom of this Importing window shows a list of URIs of test ontologies which illustrate aspects of the COE interface. Importing a few OWL ontologies and looking at the COEmaps is a good way to get a sense of what the COE interface feels like and how the graphical conventions work.*

The CMAP window

There is one CMAP window open for each COEmap being viewed. The Cmap window has several parts. When first opened, only the **main panel** is visible. This is a Cmap viewer which provides extra functionality for COEmaps.

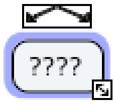
Clicking on one of the small logos on the top right opens the **side panel**. This side panel is used by a variety of navigation, documentation and editing tools, described later. To close the side panel, click on the minus-sign logo at its top right.

Manipulating nodes and links in the main panel

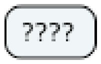
The main panel of the Cmap window is the heart of the COE GUI. You can do many things there. Nodes and links can be selected, created, joined or removed, deleted and selected. Selections can be cut and pasted; COE templates can be dragged into the COEmap and attached to existing nodes.

> NOTE. *Most of the functionality described in this section is the standard CmapTools interface, and can be skipped if you are familiar with CmapTools.*

To create a **new node**, double-click on the background surface. A new blank class node appears, which is *selected* (it has a blue outline and an arrow box on top).

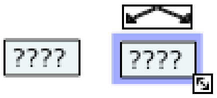


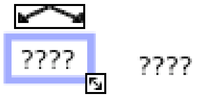
selected class node



unselected class node

By default, COE constructs a *class* node, as these are the most common nodes in OWL ontologies (see "OWL concepts"). Class nodes have a rounded outline box. To create an *individual* node, shift-double-click on the background; to create a new *literal* node, control-double-click on the background. Individual nodes have a square outline box; literal nodes have no outline. The selection highlighting is similar in all three cases.





Once a node or link is selected, you can:

- **change its size** by clicking and dragging the arrow box in the lower right;

- **drag it** to a new position;

- **merge it with an existing node** by dragging it over the other node and then holding control-shift (mac: apple-shift) while releasing it. The label of the dragged node is lost; the new merged node has the same label as the target node.

- **draw a new arc** from it by clicking on the top arrow box, dragging away, and releasing. If you release on the background, a new node is created; if you release on a node, the new arc attaches to it. If this new arc came from a node, it will also have a link on it. (To attach an arc end to a *link*, use control-shift (mac: apple-shift) while releasing. This is only required in some special cases.)

> TIP. *If you meant to attach an arc to a node but miss it, so a new node is created by mistake, then select the link immediately back along the arc, drag a new arc from it to your intended node, then select and delete the unwanted new node.*

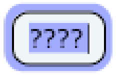
To **delete** a node or link, select it and hit 'delete'. Nodes can exist alone, without a link, but a link must be attached to nodes at its ends. If you delete the end node of an arc, and the link on that arc has no other arc coming from it, the link will also be deleted.

> TIP. *COEmaps allow 'branching' links which have several outgoing arcs from the link. To create one, simply make a single link between two nodes, then drag extra arcs from the link to other nodes or new nodes. These are often a good way to*

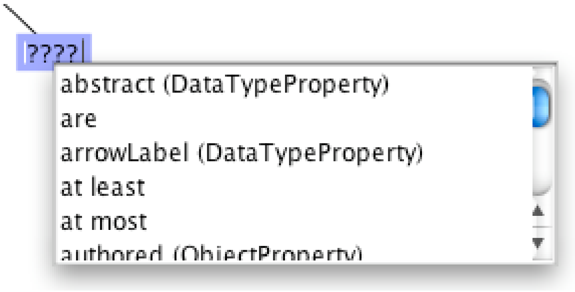
keep COEmaps more tightly organized. Sometimes they are created automatically.

To de-select a node or link, click elsewhere. To select a node or link, click on it. (To select several nodes, see below.)

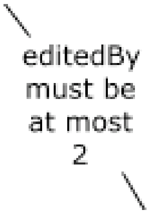
If you double-click on a node or link, its **interior** becomes selected.



You can then **type or paste a label** into the node or link. If you select the interior of a link, a **link-label menu** will appear, containing labels used elsewhere in this COEmap and also the COE **link label phrases**.


????
abstract (DataTypeProperty)
are
arrowLabel (DataTypeProperty)
at least
at most
authored (ObjectProperty)

To insert from this menu, click on the entry. You may insert several items from this menu, but not all combinations make sense. Each phrase appears on a separate line within the label, for example:



The meanings of the various COE arc label phrases are described later.

To dismiss the link-label menu and de-select the node or link, click elsewhere.

Sometimes it is necessary to delete just one arc between a link and a node, or to move one end of an arc. To select only the arc, click on it. It can then be deleted. (If this is the only arc to or from a link, the link will also be deleted.) The ends of a highlighted arc are indicated by small white squares: these can be dragged to another node or link. To attach an arc end to a *link*, use control-shift (mac: apple-shift) while releasing.

Arcs in COEmaps are drawn as straight lines by default. COE can also use various kinds of curved arcs. To make an arc into a curve, select the arc and then control-click (mac: apple-click) on it. A curve control point appears where you have clicked: drag this aside to make the arc curved.

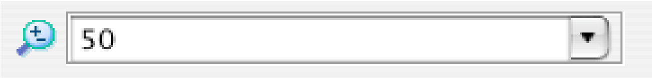
> TIP. *This is a good method for disentangling overlapping arcs.*

For more elaborate changes, select the arc or arcs (to select all arcs, use menu: **Edit/ Select Connections**), then menu: **Format/Styles**, then in the styles menu: **Line**, then **Shape**. To re-style a single selected arc, right-click (mac: control-click) on it and select **Line...** from the popup menu.

> TIP. *The CmapTools styles menu gives you great control over how nodes and arcs appear, but extensive use of it will interfere with the COE graphic conventions and is not recommended.*

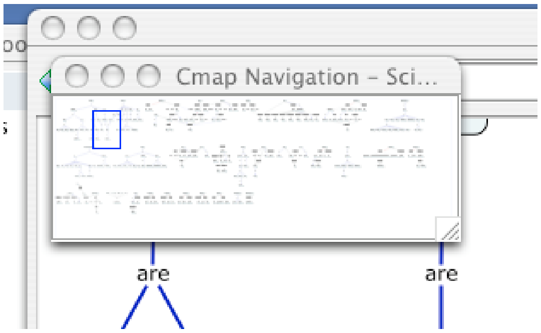
Main panel viewing options

The main panel view can be adjusted in various ways. To **zoom in or out**, use the size adjusting control in the window toolbar:



control-shift-Z (mac: apple-shift-Z) selects the 'fit to window' size option. To **move the view** across the COEmap, use the window slider bars or the **navigator tool**.

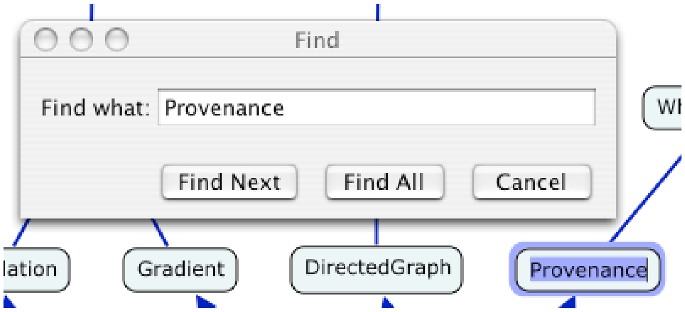
To open the navigator tool, use menu: **Window/Show Navigator Tool**. This provides a zoomed-out view of the whole COEmap surface, with the current position of the main panel shown as a blue outline box.



Dragging this box moves the focus of the main panel view. The navigator tool is a good way to get a sense of 'where you are' in a large COEmap. It is a separate window and can be dragged away from the main panel. To change the size and shape of the navigator tool window for better visibility, select and drag the foldover tab in its lower right-hand corner.

> TIP. *The COE importation often produces 'wide, short' COEmaps, so making the navigator that shape is often a good idea.*

A concept name can be 'cloned' into several places in a COEmap. The **Find tool** (menu **Edit/Find**. or control-F (mac: apple-F); copy and paste, or type, a concept name; **Find Next**)



Find what: Provenance

finds the 'next' occurrence and moves the main panel to be centered over it. The order is arbitrary but fixed.

> TIP. *Watch the navigator tool while rapidly clicking on Find Next to cycle COE through the occurrences of a given concept in the COEmap. This can be used to quickly count the number of times it occurs.*

Selecting groups of nodes

There are several ways to **select a group of nodes and/or arcs**. The selected items are highlighted. Once selected, the group of items can be dragged across the COEmap as a rigid unit, or copied/cut and pasted into another COEmap.

> NOTE. *The operation applies only to the items actually selected. For example, moving two linked nodes does not automatically move the link, and moving a link does not move the nodes it links to. It is impossible to paste a link by itself.*

The simplest way to select is to make a single click on the background, then drag along a diagonal to create a **selection box**. You can also **select multiple items** by clicking on them individually while holding down the control (mac: apple) key.

For alternative selection tools, select a node or nodes and then right-click (mac: control-click) *on the selection* to bring up the **selection menu**. This provides several selection options which use the topology of the COEmap rather than the geometry of the surface.

Select children expands the selection from the selected nodes along the link direction one link step. **Select subtree** takes the selection all the way along the link directions until reaching the end of the branches in the COEmap. **Select parents** extends the selection one link 'backwards', i.e. in the reverse direction of the link. **Select suptree** extends this backward selection all the way back along branches in the COEmap. **Select connected component** extends the selection along all links in both directions. For example, applying this to one node in the subclass hierarchy will select the entire subgraph.

> NOTE. *COE draws the subclass hierarchy with links 'upside down', i.e. the link direction pointing upwards in the Cmap surface, in order to have the largest classes at the top of the page. This means that the parent/child and sub/sup terminology may seem unintuitive at first when navigating a subclass hierarchy: for example, parents are* ***below*** *their children. This only applies to the subclass hierarchy subgraph. Other parts of a COEmap are oriented with arrows pointing downwards.*

These selections can be used progressively, each building on a previous selection. Single nodes and links can be erased from, or added to, any selection by clicking on them while holding the control (mac: apple) key.

It is also possible to select based on concept names by using the **Cmap List View**: see the next section.

Viewing a COEmap in list form

Click on the top logo on the right-hand side of the main panel

A side panel opens showing the **Cmap List View**. There are four options, selectable by clicking on the small black arrows.

Selecting items in this window also selects the corresponding occurrence in the COEmap. For single selections this also moves the main panel to center on that occurrence. To perform multiple selections, hold down the control (mac: apple) key while selecting.

> TIP. *You can use the selection menu to expand a selection made using the Cmap List view.*

Export as Text creates a .txt file of the list currently displayed.

‣ **Concepts**. This is a list of all the node names used in the COEmap, ordered alphabetically, showing the number of links in and out of the node where it occurs. (To reverse, or re-order by number of links, click on a column header.) Concept names are listed here once for each occurrence in the COEmap.

‣ **LInking Phrases**. This is a list of link names which occur in the COEmap. It behaves just like the Concepts view.

‣ **Propositions**. This is a list of all the node-link-node triples in the COEmap. Selecting here selects the corresponding triple(s) in the COEmap. Clicking on a column header re-orders by that column name.

‣ **Cmap Outline**. This is a summary of all the connected paths through the COEmap, organized as a folding hierarchy with superclass relationships indicated by indenting.

URIs and namespaces

The OWL and RDF specifications require that all concept names used in ontologies are full URI references such as

http://sweet.jpl.nasa.gov/ontology/earthrealm.owl#Troposphere

The part preceding the # sign is the *base URI*, which is often (though not always) the Web address of the source ontology for the concept - i.e. the place where it was first published and which is supposed to be authoritative - and the remainder following the # sign is the *local name*. As these full names are not very readable, COE uses the XML *Qname convention* where the base URI is replaced by a brief prefix attached to the local name by a colon, for example:

earth:Troposphere

To see the full URI reference name of any concept, simply hover the mouse over a node or link containing the name: a yellow box will appear showing the full URI reference.

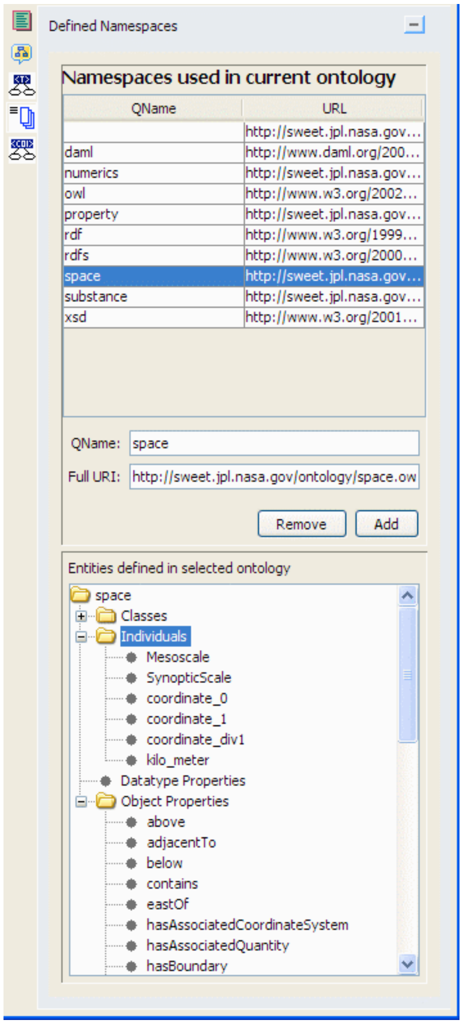
If the source XML document defines any Qnames (usually the case) then these are also used in the imported COEmap; for other base URIs COE generates prefixes *n1*, *n2*, etc.. By convention, the prefix corresponding to the COEmap source itself is left blank, so that the ontology's 'own' concept names appear without a prefix. In this case the colon is also omitted.

If part of a COEmap is pasted into another COEmap, then any blank prefixes are replaced by a generated prefix name linked to the base URI of the originating ontology of the first COEmap.

The association of prefixes to URIs is called a ' namespace'. To **inspect the namespaces** of a COEmap, click on the bottom logo on the right side of the main panel

A side panel appears listing the namespaces. To **change any URI or prefix**, select a namespace and use the lower panels to insert the new entry. To see a **summary of all the concept names** originating from each ontology, select one of the namespaces, right click (mac: control click), then select **Show ontology contents in this panel**. The concepts will be displayed in file-folder format at the bottom of the panel, organized into *classes*, *individuals*, *datatype properties* and *object properties*. (See "Basic OWL ideas"). To enlarge the view, drag the side panel divider upwards.

(NOTE, this is *all* the concepts in the selected ontology, which may be more than occur in the COEmap itself.)



Namespaces used in current ontology

QName	URL
	http://sweet.jpl.nasa.gov...
daml	http://www.daml.org/200...
numerics	http://sweet.jpl.nasa.gov...
owl	http://www.w3.org/2002...
property	http://sweet.jpl.nasa.gov...
rdf	http://www.w3.org/1999...
rdfs	http://www.w3.org/2000...
space	http://sweet.jpl.nasa.gov...
substance	http://sweet.jpl.nasa.gov...
xsd	http://www.w3.org/2001...

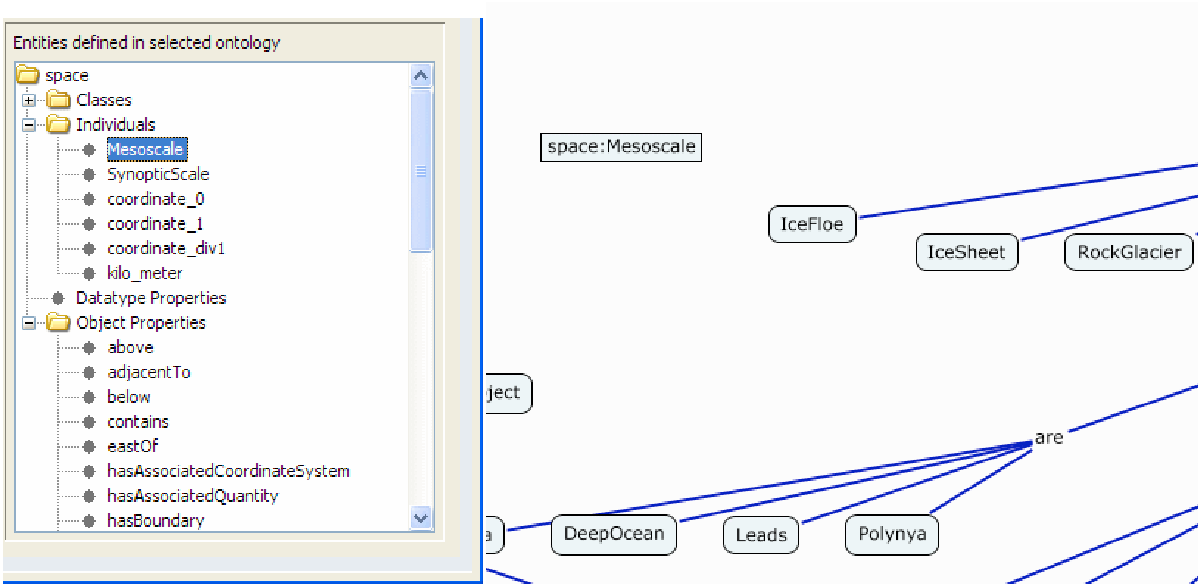
QName: space

Full URI: http://sweet.jpl.nasa.gov/ontology/space.ow

Entities defined in selected ontology

- space
  - Classes
  - Individuals
    - Mesoscale
    - SynopticScale
    - coordinate_0
    - coordinate_1
    - coordinate_div1
    - kilo_meter
  - Datatype Properties
  - Object Properties
    - above
    - adjacentTo
    - below
    - contains
    - eastOf
    - hasAssociatedCoordinateSystem
    - hasAssociatedQuantity
    - hasBoundary

Here we have selected the Space ontology and right clicked to **Show ontology contents in this panel.** The classes, individuals, and properties are listed, and can be dragged into the current ontology.



Entities defined in selected ontology

- space
  - Classes
  - Individuals
    - Mesoscale
    - SynopticScale
    - coordinate_0
    - coordinate_1
    - coordinate_div1
    - kilo_meter
  - Datatype Properties
  - Object Properties
    - above
    - adjacentTo
    - below
    - contains
    - eastOf
    - hasAssociatedCoordinateSystem
    - hasAssociatedQuantity
    - hasBoundary

Here we have selected the **Mesoscale** individual and dropped it on the map, creating an individual node.

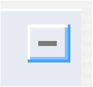
KNOWN BUG. Dragging a property to the COEmap will create a COEmap fragment which does not export correctly.

Selecting **Import ontology to concept map** will import the RDF or OWL file found at the selected base URI, if there is one, and display it in a new COE window. If there is no readable file at that Web address, nothing happens.

> TIP. *This can be used to 'surf' between Web ontologies, searching for how concepts are used in them and how these uses relate to the assertions in their originating documents.*

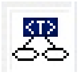
(Both of these operations require internet access.)

To **close the side panel**, click on the minus logo at the top right corner:



Templates

COE provides **templates** for a number of common node-link combinations. To see the template menu, click on the third logo at the right of the main panel:



A side panel opens showing the menu. Drag a template from here into the main panel. If you drop it on the background, the template will appear as an isolated subgraph. If you drop it on a node, it will be merged with that node. In this way one can rapidly construct common COEmap structures. Once in the COEmap, templates are simply parts of the map and can be edited using the COE interface.

The meanings of the various templates are explained in the next section.

Making new templates

You can construct your own templates, or keep alternative template sets for different purposes. To create a template, use menu **File/New Template**. A COE window opens: construct the template using COE, then hit menu **File/Save Template**. Select a name for the template, and save it somewhere in your *My Cmaps* folder. It is convenient to use a Template sub-folder, but this is not required. Folders of templates outside your *My Cmaps* folder will not be indexed, so you can switch template options by moving template folders in and out of *My Cmaps*.

COE notation and conventions

COE uses lexical and graphical conventions to express OWL, RDF or RDFS content in a COEmap. These conventions have been chosen to be humanly readable and intuitive rather than to mirror OWL/RDF syntax. This section explains the meanings of the COE conventions in intuitive terms.

Individuals are the atomic 'things' or objects that the ontology is about. For example, we might have an ontology about people, and then people would be our individuals.

Classes are named groups or categories of individuals: for example, Mothers and Males might be two classes of people.

Object properties - sometimes just called **properties** - are binary relationships between two individuals. For example, *isMarriedTo* and *isMotherOf* might be properties over people. Properties come in various flavors: for example, *isMarriedTo* is **symmetric** but *isMotherOf* is not.

Literals are just like individuals except they have 'fixed' conventional names, usually defined using a datatype. Examples of literals include numbers, dates and strings.

Datatype properties are just like object properties except that they relate an individual to a literal. An example for people might be *age*, relating people to a number, or *dateOfBirth* relating them to a date. (The reason for distinguishing two kinds of property is to make life easier for the reasoning engines, rather than because of any significant difference in meaning.)

KNOWN BUG. COE currently has no way to explicitly indicate the presence of an RDF blank node. A blank COEmap individual node, i.e. one without a label or with the node label ????, may not export correctly

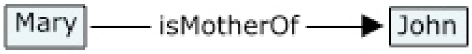
COE always represents individuals, classes and literals as nodes (distinguished by their graphic style) and properties as links.

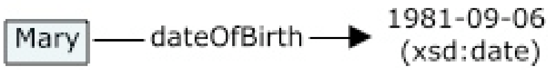
> NOTE. *Some ontologies don't make all these distinctions. OWL-DL, the commonest dialect of OWL, requires all of them to be clearly distinguished; OWL-Full allows classes to also be individuals; RDFS allows complete freedom to mix the various categories. When displaying an RDFS ontology, COE may draw an individual as a class node, since the distinction becomes meaningless in RDFS, and COE treats classes as a default. This is not necessarily an error, though exporting such a COEmap into OWL may cause an OWL-DL error.*

Several relationships can hold between classes, individuals and properties.

Relationships between individuals and classes

The simplest kind of COE statement is that a property holds between two individuals or an individual and a literal. Simple assertions such as these are called *instance data.*

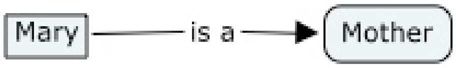




> NOTE. *It is illegal to have a property link* from *a literal.*

> TIP. *Ontologies containing large amounts of instance data are best handled using one of the advanced importing options.*

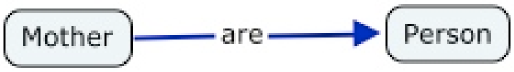
An individual can be *a member of* a class, indicated in COE by a link from the individual to the class with the label **is a***,* For example,



says that Mary **is a** member of the class Mother. To construct this, use the **A is B template**, or draw a link from the individual node to the class node, select the link interior and choose **is a** from the popup link menu.

Relationships between classes

One class can be a *subclass* of another, indicated by a link with the label **are**. Mother is a subclass of Person; that is, mothers **are** persons:

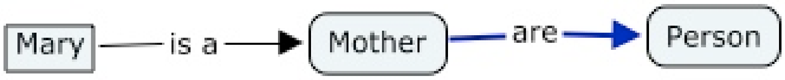


To create this, use the **A are B template**, or make a link between the nodes, select the link, and choose **are** from the popup link menu. (The second method does not color the link blue.)

> TIP. *If you get the direction wrong by mistake, right-click (mac: control-click) the link and choose* ***Reverse Connections*** *from the popup menu.*

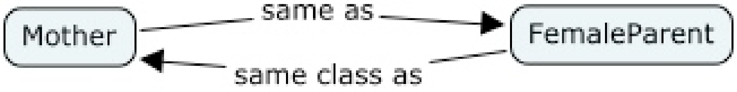
When importing and laying out an ontology, COE uses a blue color, and attempts to group all the **are** links into a single connected subgraph, called the **subclass hierarchy**. This is drawn, following a widely used convention, with the 'largest' classes at the top, so that the **are** links point *upwards* on the page. The rest of the COEmap is drawn with all links pointing downwards. The orientation of links on the page makes no difference to what they mean: only the actual link direction is meaningful.

> NOTE. *The two facts shown here can be unified into a single COEmap*:



> *which has exactly the same meaning as the two COEmaps shown above. In general, two COEmaps with the same node-arc-node triples mean exactly the same, and will export to the same ontology.*

There are several simple relationships between classes which are indicated by special link labels. Two classes might be identical, i.e. there is one class with two different names. This can be expressed using the link labels **same class as** or **same as**:

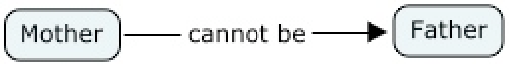


As both of these are symmetric in meaning, the direction of this link is unimportant.

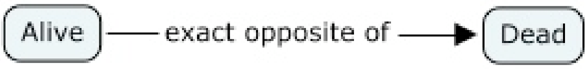
> NOTE. *These differ subtly in their exact OWL meanings, but such subtleties are beyond the scope of this primer. See the OWL specification documents. The label* **same as** *can be used between individual nodes also*.

The simple denial of sameness is indicated by the link label **different from**.

Two classes may be exclusive, in that anything in one cannot be in the other. This is indicated by a **cannot be** link:

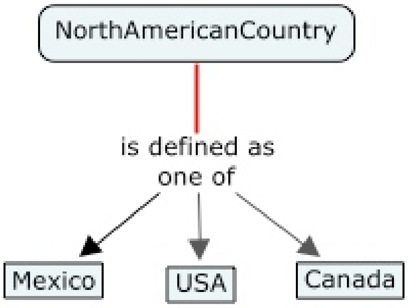


Finally, the strongest kind of mutual exclusion is between a class and its *complement*, i.e. the class of *all* things that are not in it. Two complimentary classes share no elements but together exhaust the universe. Complimentarity is indicated in COE by the link label **exact opposite of**:



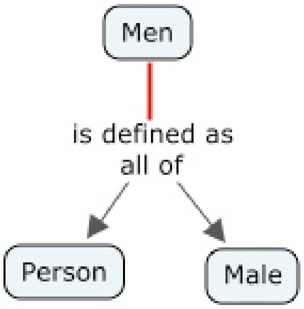
Describing classes

Sometimes a class is best defined simply by listing all its members. COE uses the OWL phrase **one of** to indicate this, for example:

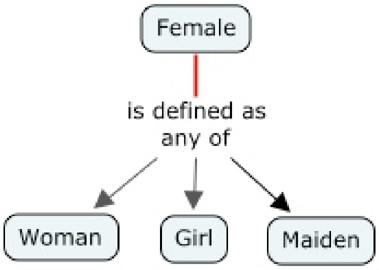


Note the red line: this color is used in COE to highlight a *definition*, i.e. a case where the conditions on the class are exactly necessary and sufficient. To create this, use the **New Class Def One Of Individuals template**. This comes with two blank individual nodes ready to use: to add more, create new individual nodes by shift-clicking on the background, then drag arcs from the link to them.

A class may be defined by a combination of other class conditions. For example, the class Men consists of things which are satisfy **all of** the conditions of being Persons and Male:



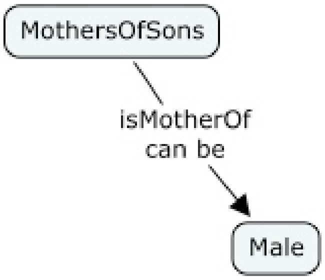
This is often called the *intersection* of the classes. To create this, use the **New Class All Of Definition template**. As the name suggests, there can be more than two classes involved: simply drag new arcs from the link to add more. Other combinations include *unions*, i.e. classes whose members meet **any of** several conditions, for example:



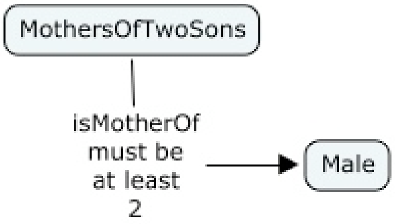
To create this, use the **New Any Of Class Definition template**. Another is a class being defined to be all the things that are *not* in another class, which is indicated by an **exact**

opposite of link between the class nodes; to create this, select from the popup link menu.

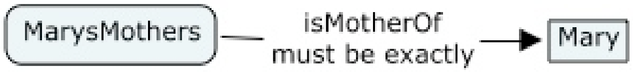
A class may be defined by a *restriction* on a property. There are several kinds of restriction. For example, the class of all mothers who have a son can be defined by saying that it is the class of things such that the *isMotherOf* property, when applied to that thing, must have at least one value in the class *Male*. To say this in COE, use the special link label **can be,** after the property name, on a link to the class:



To create this, type the property name into the link, then select **can be** from the link popup menu, then insert the property name. Other restriction types include **must be**, meaning that *all* the values are found in the named class (used here, this would define mothers who have no daughters) and numerical restrictions which specify bounds on the numbers of values, defined using the link labels **must be** and one of **at least**, **at most** and **exactly** from the popup link label menu. For example, mothers with at least two sons:

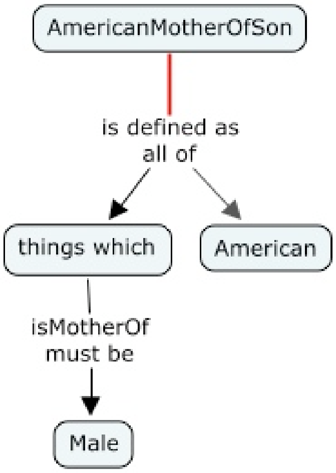


Some restrictions can be applied to individuals rather than classes, indicating that the property value must be this particular thing:

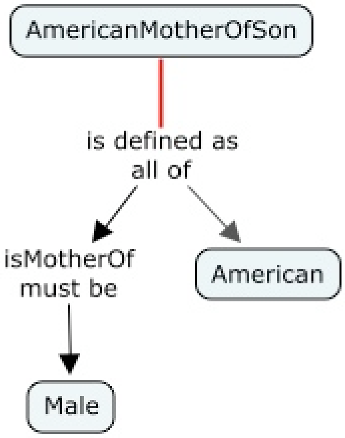


NOTE. *Only these combinations should be used in restriction link labels. The other link label menu items have different uses. In particular,* **cannot be** *is* not *the opposite of* **can be**.

Restrictions are always treated by COE as *necessary* conditions on a class, rather than as exact definitions, which is why they are not drawn with a red arc. This is the most usual way that restrictions are applied. A typical combination is an **all of** definition with one component being a restriction:



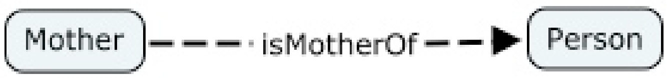
The node label **things which** indicates an 'anonymous' class which is used only within this definition. COE standardly abbreviates this pattern:



To create this, either use the **New All Of Restriction template** directly, or drag an arc from the top link and control-shift (mac: apple-shift) drop it onto the lower link, making a direct link-to-link arc, then delete the intervening class node.

Relationships between properties

An object property can have an associated **domain** and **range**, which are classes. Anything to which the property applies must be in the domain, and any value that it has must be in the range. For example, the property *isMotherOf* has range *Person* but domain *Mother*. This is indicated in COE by putting the property on a special link, indicated by a bold dashed line, from the domain to the range:



To create this, either use the **Property Domain and Range template**, or draw a normal link from the domain to the range, right-click (mac: control-click) on it, and select **Change to Domain and Range** from the pop-up menu.

> TIP. *To define just one of the domain or range, leave an empty node label* *at the other end of the domain/range link.*
>
> TIP*. The domain and range may be the same class. It is quite OK to write the class name twice in two different places in a case like this. When importing such an ontology, however, COE might well create a 'folded' link with overlapping arcs. To make such a COEmap look better, select the arc, control-click (mac: apple-click) on it to create a curve control point, and drag the point aside.*

Properties can be declared to satisfy some useful mathematical conditions. A **symmetric** property applies both ways round, so the link direction is unimportant. *Being a sibling of* is a symmetric property. A **transitive** property is inherited along chains of relationships. For example, *being an ancestor of* is transitive, since an ancestor of an ancestor is itself an ancestor. A **functional** property is one whose value is always unique, so that nothing can have two or more distinct values for the property. *hasMother*, the property whose value is a person's natural mother, is functional. Finally, a property is **inverse functional** if no two things have the same value of this property. Inverse functional properties are like database keys; their values can be used to identify what it is that the property applies to. The property whose value is a person's social security number should be inverse functional. All of these are indicated in COE by attaching one of a series of property signs to the property name on a link. Exactly where in the COEmap the label occurs is not important, only that it occurs on the same link label as the property name somewhere in the COEmap. Once is enough.

These signs are selectable from the link label menu.

> **<=>** means **symmetric**

> **=>=>** means **transitive**
>
> **>>=** means **functional**
>
> **=<<** means **inverse functional**.

A property may have any combination of these labels.

In addition, a property may be declared to be an **object property** or a **datatype property** by appending one of these labels to the name of the property. This, again, only needs to be done once per COEmap, and is required only for OWL-DL ontologies.

Finally, there are two binary relationships between properties. Two properties may be **inverses** of one another, meaning that they are the same property with the direction of the link reversed. *isMotherOf* and *hasMother* are inverse properties. Or, one property may be a **subproperty** of another. This means that if the subproperty holds between a thing and a value, then the other property must also hold between them. For example, *isMotherOf* is a subproperty of *isParentOf*.

Both of these require drawing a link between two other links. To do this, drag an arc from one link and control-shift (mac: apple-shift) drop it on the other link. Then select the link, right-click (mac: control-click) on it, and select **Change to Subproperty** or **Change to Inverse** from the popup menu.

KNOWN BUG. **Change to Inverse** sometimes does not appear in the menu.

Saving and exporting COEmaps

Any COEmap may be simply saved as a CmapTools file. (Menu **File/Save** or **Save As**.) COEmaps can also be output in a number of formats (menu: **File/Export Cmap As>**) or as an ontology (menu **File/Export Cmap As/Ontology File in Format>**). Selecting **OWL** exports into OWL/RDF/XML, the recommended standard interchange format for the Semantic Web.

> NOTE. *The **XML File ...** option under **Export Cmap As** is **not** an ontology format: it is used by CmapTools to transmit Cmaps in XML.*
>
> TIP. *To automatically layout a large ontology, export it to an OWL file then re-import it.*

COE does *not* check that COEmaps are 'legal' or 'wellformed'. It exports into legal OWL-DL where possible, and into RDF any node-link-node triples which are not legal OWL. If it finds constructions which cannot be exported, it ignores them in the exported file but issues a warning.

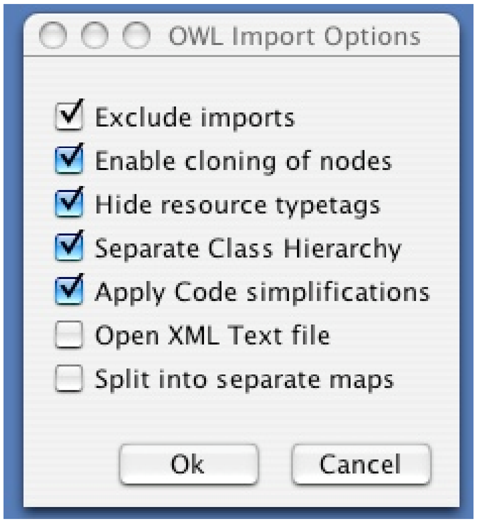
KNOWN BUG. In some cases it does not issue a warning.

Exporting with namespaces

When constructing a new ontology, COE inserts http://localhost/default as a default base URI corresponding the blank Qname. This is sufficient to enable COE to export and re-import any COEmap, but before exporting a completed ontology intended for public use, this should be replaced by an appropriate base URI for the ontology in its intended location of publication as a Web resource. For advice on how to choose an appropriate URI for an OWLontology, see the W3C website.

Importing Options

When importing an OWL ontology into COE, a menu of options appears:



☑ Exclude imports
☑ Enable cloning of nodes
☑ Hide resource typetags
☑ Separate Class Hierarchy
☑ Apply Code simplifications
□ Open XML Text file
□ Split into separate maps

Unchecking **Exclude Imports** will search through the ontology looking for *owl:imports* statements and build a COEmap of the resulting imports closure. This can be a very large COEmap.

Unchecking **Enable cloning of nodes** will draw a COEmap in which each node label appears in only one place in the COEmap. Such maps are often extremely hard to read. Try it and see.

Unchecking **HIde resource typetags** will produce a COEmap in which every node has its OWL type category added to the label. This can be useful for debugging purposes.

Unchecking **Separate Class Hierarchy** switches off the layout preference to collect all subclass links into a connected tree.

Unchecking **Apply Code Simplifications** removes some of the abbreviations and short-cuts in the standard COEmap style.

Checking **Open XML Text file** opens a second window showing the ontology source file

Checking **Split into separate maps** produces a more sophisticated input option recommended for large ontologies, described in the next section.

Dealing with large ontologies

The input option **Split into separate maps** imports an ontology into a group of COEmaps organized into a folder. COE divides the ontology into up to five pieces, though not all ontologies will provide material for all of them:

Class Hierarchies (subclass information) in a COEmap

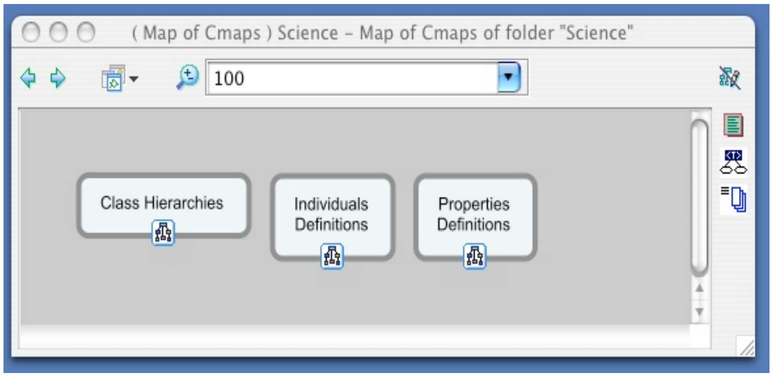
Property Definitions (domain and range information and subproperty and inverse-of assertions) in a COEmap

Individual Definitions (membership of individuals in classes) in a COEmap

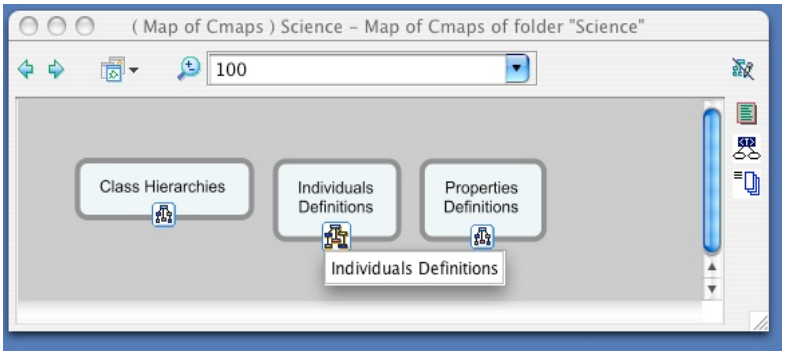
instance data in an **Instance File**

anything else left over as a COEmap.

After importing, the folder is created in My Cmaps and a **map of maps** is opened in a new Cmap window:



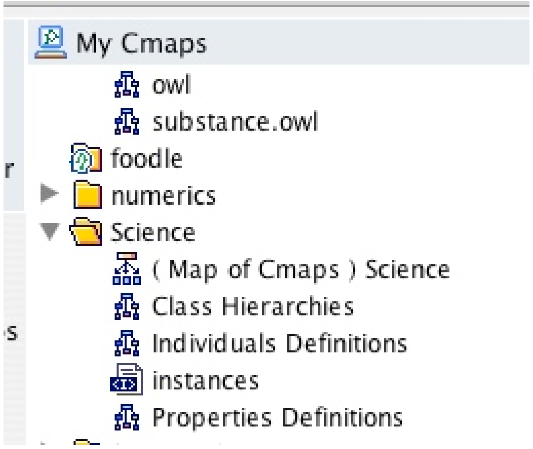
To open one of the component COEmaps, select the Cmap tag:



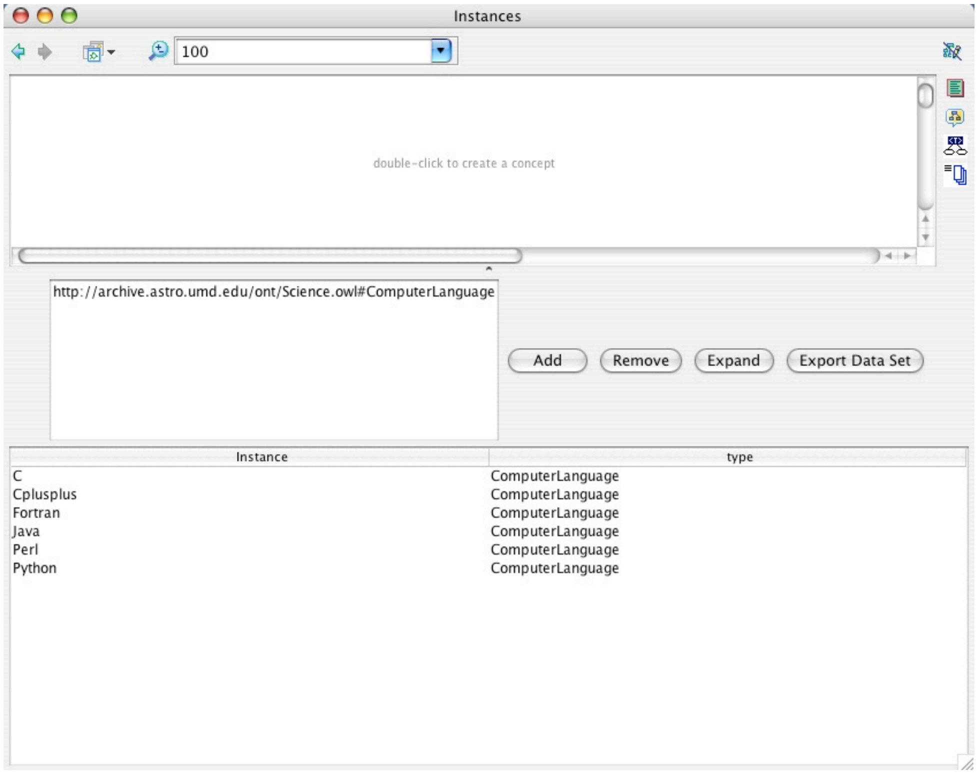
then click on the name. The component COEmap will open in a new window. The separate COEmaps can be edited and processed normally, but will be exported as a single ontology file with a common namespace.

The instance viewer

To view the instance data of the ontology, go to *My Cmaps* in the **Views** window, open the folder and double-click on **Instances**:



An **instance viewer window** opens. This has a blank COEmap main window at the top, but displays all the instance data from the ontology as a table of individuals with their immediately enclosing classes.



http://archive.astro.umd.edu/ont/Science.owl#ComputerLanguage

Instance	type
C	ComputerLanguage
Cplusplus	ComputerLanguage
Fortran	ComputerLanguage
Java	ComputerLanguage
Perl	ComputerLanguage
Python	ComputerLanguage

Double Right Click (mac: **control double click**) on a selection in the table causes it to be displayed as a COEmap in the main window.

Shift Double Right Click (mac: **shift control double click**) on a selection in the table adds it to the already displayed instance in the COEmap.

Left Click on a class in the middle panel filters the list of instances according to the class(es) selected, i.e. displays only instances which are in the class(es).

Add adds a new instance of the type selected in the middle panel filter.

Remove removes the currently selected row(s).

Expand elaborates the instance shown in the COEmap by finding other instances in the graph and importing them into a map. In this way, hierarchical structure of an instance can be viewed while showing only instances of one particular type in the table.

Export Data Set exports data that can be used by the Command Data Mining Engine. See the COE/DME integration manual for more details on this interaction mode.