The Network Architecture of CmapTools

Technical Report IHMC CmapTools 93-02

Alberto J. Cañas, Greg Hill, Adrián Granados, Carlos Pérez, Juan David Pérez
Institute for Human and Machine Cognition
40 South Alcaniz St.
Pensacola Fl 32501
www.ihmc.us

Introduction

The *CmapTools* software suite was designed with the purpose of providing an environment that would enable and encourage collaboration and sharing in the construction and use of knowledge models[1] based on concept maps (Cmaps). The client program works fine as a standalone tool, allowing the user to construct concept maps and store them on the user computer's hard drive, establish links among maps and to resources[2] in the hard drive, and print the maps. However, the ease with which knowledge models can be shared and collaboration can be established through *Places* (servers) is what makes *CmapTools* unique.

Client-Sever Architecture

The client program *CmapTools* is used construct and manipulate Cmaps. The program runs on the user's computer, allowing the user to create knowledge models and store them.

The storage of the knowledge model can take place in the computer's hard drive or on a *Place* (server). To store the knowledge models in the user computer's hard drive no software in addition to *CmapTools* is needed. However, to share a knowledge model with users in other computers, the client program needs to communicate through a computer

[1] A *Knowledge Model* is a collection of concept maps and associated resources about a particular domain of knowledge.
[2] Resources are files linked to concept maps that help explain and complement the information in the map. They can be of any type, including videos, images, text, web pages, documents, presentations, and other Cmaps, among others.

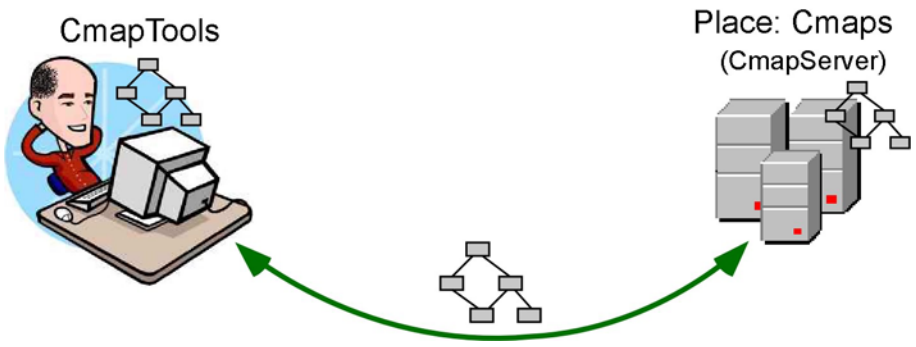

Figure 1: Client *CmapTools* saves and retrieves Cmaps from a *Place* (*CmapServer*)

network with a *Place*: a server computer running the *CmapServer* software. Figure 1 shows a user running the CmapTools software on his computer, storing and retrieving Cmaps from the "Place" server.

Several users can be accessing the same *Place* concurrently, storing and retrieving concept maps. This way, users can easily share and collaborate in the construction of their knowledge models.

Locating A Place

When the *CmapTools* program starts executing, it needs to locate available *Places*. First, the program tries to locate *Places* on the Local Area Network (LAN) where the client program's computer is running. Second, and most important, the program tries to locate *Places* available throughout the Internet. To facilitate collaboration and sharing, the Network Architecture of *CmapTools* has been designed in such a way that a user will automatically and transparently be able locate all *Places* that are available (that is, not "hidden" by their administrators), wherever they are located on the Internet. Third, the user can explicitly provide *CmapTools* with the Internet address of a particular *Place* he/she wants to locate. We discuss each of these in the following paragraphs.

a) *Servers in the Same Local Area Network*

The *CmapServer* is capable of advertising its presence on the LAN where it is installed, using the "Service Location Protocol" (SLP) [1, 2] mechanism, a protocol for locating resources in a network. When the *CmapTools* client starts executing -- and periodically while it is running -- it sends a broadcast message to locate the presence of *CmapServers* in the LAN. As shown in Figure 2, clients installed in the same LAN as a server will automatically find the *Place* located in that server through its broadcast. No configuration is needed on the client or server side, the discovery and location is automatic.

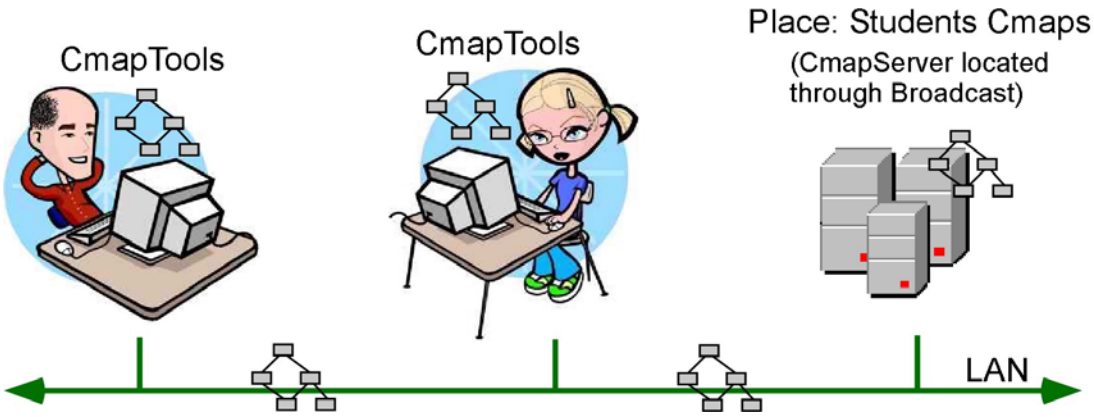

Figure 2: *CmapTools* clients and *CmapServer* running on the same LAN: the clients find the *Place* via broadcast.

The reply message from the server provides the client information on what "Services" are running in that particular server. For example, it will let the client know whether the server is capable of handling Discussion Threads, Knowledge Soups, or Synchronous Collaboration Sessions among other services. The administrator for the *CmapServer* can configure the server to run only a particular set of services.

c) Locating Places through the Directory of Places

Servers can be located anywhere in the world as long as they are accessible via Internet and not protected behind a firewall. However, there needs to be a mechanism by which the *CmapTools* client can find these servers. This need is handled by the *Directory of Places* using the SLP protocol.

The *Directory of Places* is a special type of server that handles "registrations" from *CmapServers*. When a *CmapServer* starts running it contacts the *Directory of Places* it has defined in its configuration or the *Directory of Places* that might be running on the same local area network and it has located through broadcast. In addition to its Internet address, the *CmapServer* provides other pertinent data, such as its name (*Place*) and other identifying information, and the services it is running (services include Discussion Threads, Knowledge Soups, Synchronous Collaboration, etc.). Periodically, the *CmapServer* contacts the *Directory of Places* providing a "heartbeat" – an indication that it is still up and running. If a *Directory of Places* does not receive a heartbeat from a *CmapServer* for a period of time, it removes it from its list of active servers.

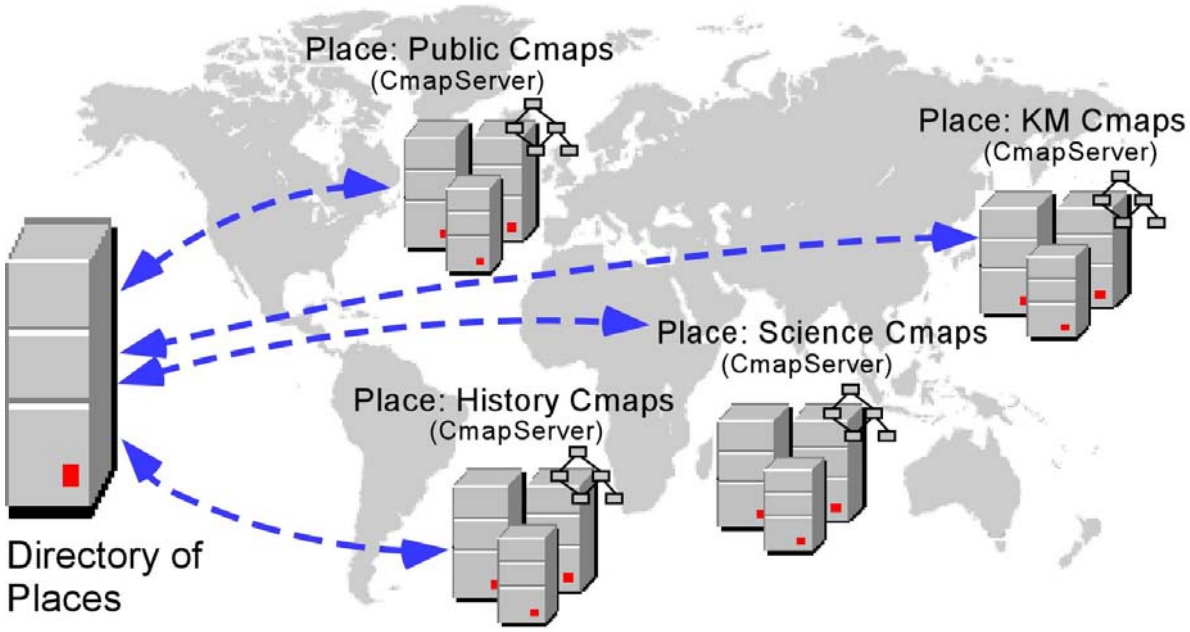

Figure 3: *CmapServers* register with a *Directory of Places* and periodically let it know they are up and running

Figure 3 shows four *CmapServers* reporting to a *Directory of Places*. Notice that the *Directory of Places* does not store any Cmaps. All it keeps is information about the *CmapServers* that are up and running and that have reported to it.

When the client *CmapTools* program starts executing, it contacts the *Directory of Places* defined in its configuration and retrieves the list of *CmapServers* that are available. It then proceeds to contact each of those servers to make sure it is accessible and retrieve information about the *Place*. Recently installed *CmapServers* will be located and show up in the *CmapTools* client. Figure 4 shows a *CmapTools* client obtaining the servers information from the *Directory of Places* and accessing the various servers.

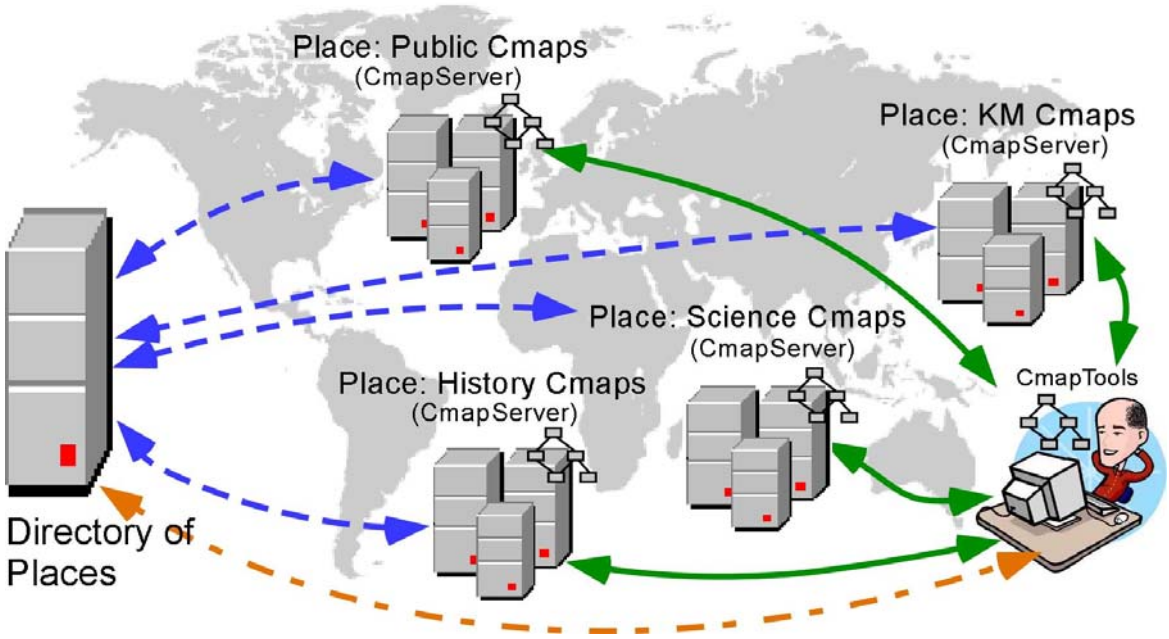

Figure 4: *CmapTools* client contacts the *Directory of Places* to obtain a list of *CmapServers*, and then contacts each *Place* separately

b) *Locating "Private Places"*

CmapServers may be configured not to register with a *Directory of Places*, or may be behind a firewall that prevents the registration from taking place. If the client *CmapTools* program is not on the same LAN as the server, then the user must explicitly provide the Internet address (IP or domain address) of the server in the program's Preferences dialogue box (Edit/Preferences menu item in *CmapTools*). In addition to the IP address, the user must provide the "Port" on which the server is listening. By default, port 4447 is used. We refer to *Places* that are located via the explicit Internet address provided by the user as *Private Places*. Once the address of a *Private Place* is provided, the client will attempt to locate that *Place* every time it is executed. The same dialogue box can be used to remove or edit a *Private Place* from the list of servers to be contacted.

Figure 5 shows two *CmapTools* users. The one on the left is accessing the "Students Cmaps" *Place* via broadcast, while the user on the right is also accessing the "Corporate Cmaps" *Place* that was located by explicitly giving its Internet address. Additionally, both of these users could be working on *Places* located via the *Directory of Places* described above.

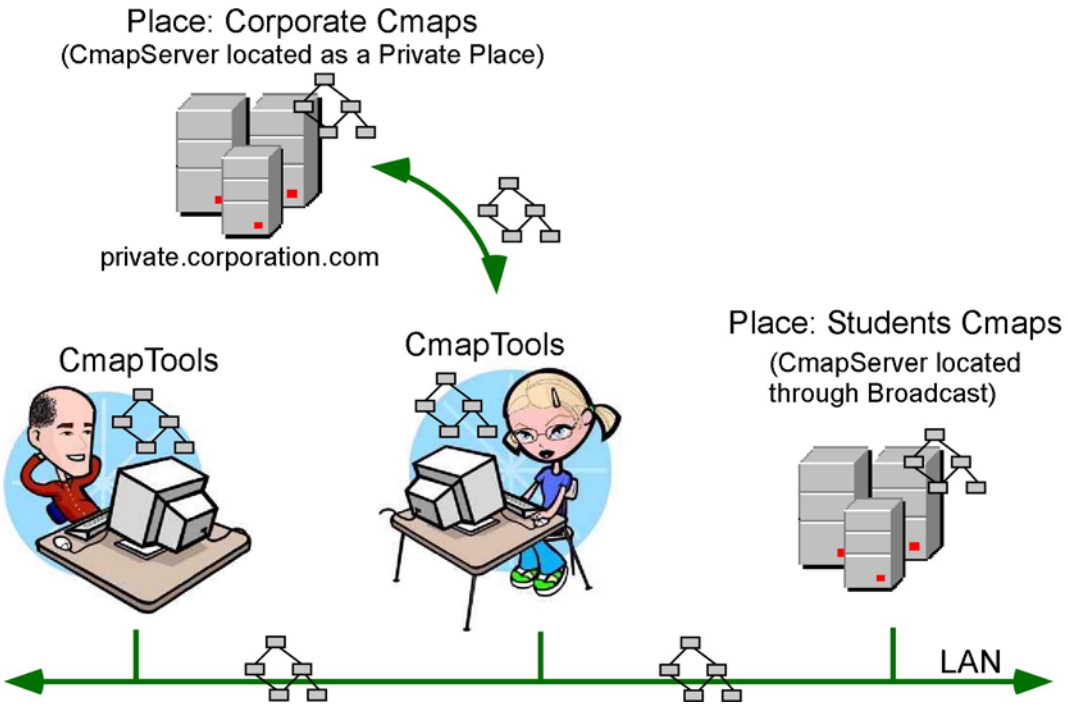

Figure 5: The *CmapTools* user on the left accesses the LAN server (Place: Students Cmaps) while the other user accesses Students Cmaps and the *Private Place* Science Cmaps.

A *Private Place* does not have to be behind a firewall or inaccessible via broadcast or through the *Directory of Places*. A server on the same LAN as the client, or one that is registered with the *Directory of Places* can be added as a *Private Place* in a client. The client program will locate the server by various means and will know that it corresponds to the same *Place*.

Figure 6 shows a *CmapTools* client connected to a variety of *Places* through the three mechanisms described above. The “Students Cmaps” Place is located via broadcast since the user’s computer is on the same LAN as that CmapServer. “Corporate Cmaps” is located as a *Private Place* by explicitly providing the Internet address of the *CmapServer*. “Public Cmaps”, “Science Cmaps”, and “History Cmaps” were located via the *Directory of Places*.

As the client program executes, it periodically checks with the Directory of Places for CmapServers that have become available or are no longer online.

Indexing and Searching

The mechanism for locating *Places* described in the previous section allows the *CmapTools* client to access *Places* located throughout the Internet, within an Intranet or in the same LAN. Looking for a particular resource within this collection of servers can take time. Each *Place* can contain a large number of knowledge models, and knowledge

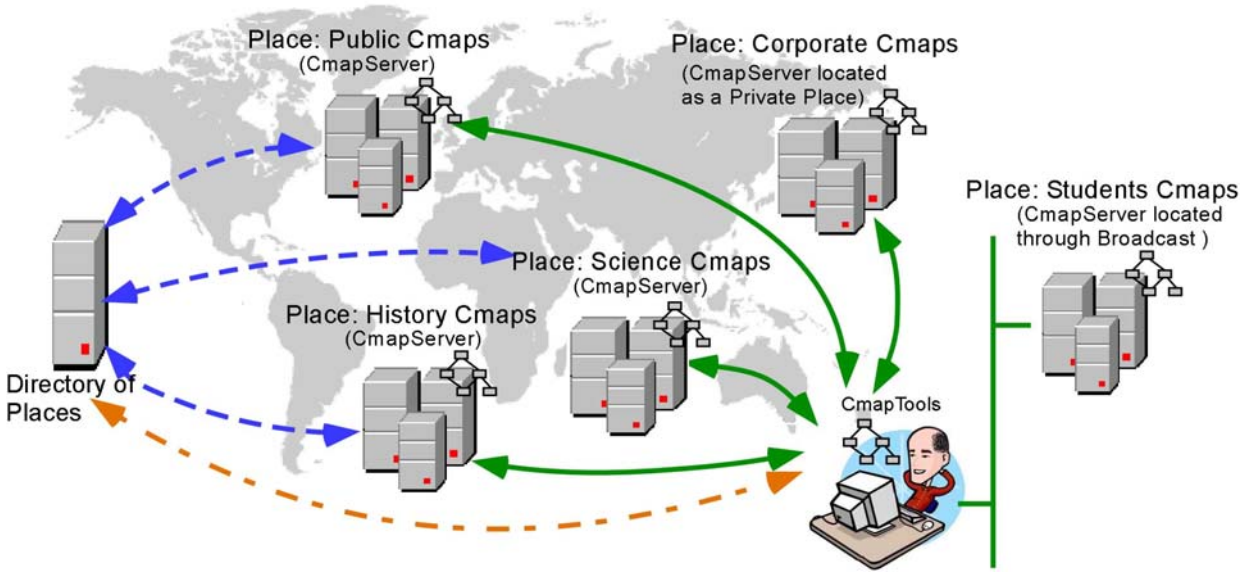

Figure 6: A *CmapTools* user takes advantage of the three mechanisms to locate *Places* (broadcast, *Private Places* and *Directory of Places*) to access Cmaps from a variety of *CmapServers*.

models can vary in size from a single Cmap to hundreds or thousands of resources (Cmaps, images, videos, text, links to Web pages, etc.).

To speedup and simplify searches for Cmaps and resources, all Cmaps and resources at the *CmapTools* client and at each *CmapServer* are indexed. The index includes: for each Cmap, all concepts, linking phrases, properties (owner, focus question, etc) notes (whether hidden or not); for all resources, the names and captions (images, videos, sounds, etc.), and in the case of links to texts or web pages, their contents. All this information is readily available for searching. Indexes and searches are implemented using the Jakarta Lucene engine [3], which provides fast response and a powerful query language.

With the number of *CmapServers* increasing and their location distributed throughout the world, searching for Cmaps about a particular topic would imply searching each *CmapServer* individually. To speed up global searches, *CmapServers* periodically send copies of their indices to an *IndexServer*, as shown in Figure 7.

When it starts up, the *IndexServer* registers with the *Directory of Places* it is configured to register with, and periodically reports that it is still running. The availability of the *IndexServer* is passed-on to the *CmapTools* clients and *CmapServers* that contact the *Directory of Places*.

The *CmapServers* contact the *Directory of Places* to find the address of the *IndexServer*. From there on, they periodically send a copy of their index to the *IndexServer*. By accumulating all these indices, the *IndexServer* acts as a central indexer for all *CmapServers*.

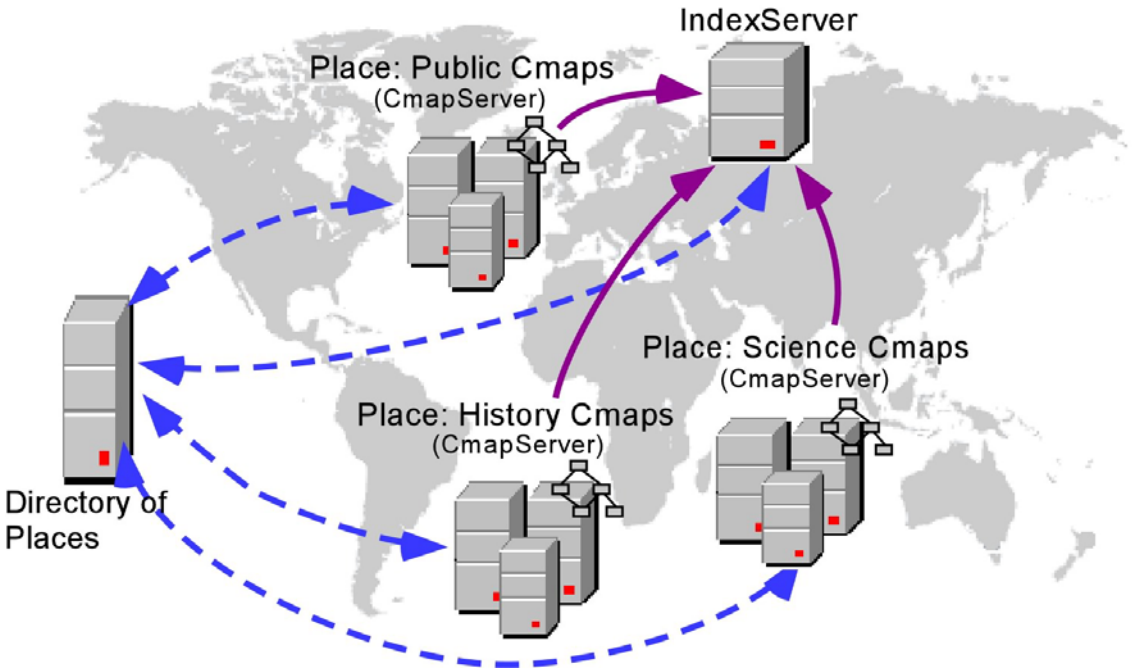

Figure 7: *CmapServers* periodically send copies of their Indices to the *IndexServer*

The *CmapTools* clients also receive from the *Directory of Places* the Internet address of the *IndexServer*. By sending a Query to the *IndexServer* instead of to a particular *CmapServer*, the user is covering all *CmapServers* registered with the *Directory of Places* in a single search.

The user has the option of specifying the location where a particular search will take place. A search on the user's computer will locate Cmaps and resources in MyCmaps. The user can also specifically select one or more *CmapServers* as targets of the search. The default search in the *CmapTools* client includes searching through the *IndexServer* (which covers all *CmapServers* registered with the *Directory of Places*), searching directly through other *CmapServers* known to the *CmapTools* client but not covered by the *IndexServer* (e.g. *Private Places* or *CmapServers* located only through broadcast), plus searching MyCmaps.

Replication of Directory of Places

The *Directory of Places* plays a key role in the communication between nodes of the *CmapTools* network. Through the *Directory of Places*, the *CmapTools* client finds *CmapServers* and *IndexServers*. It is therefore essential that the *Directory of Places* be always available and respond fast to requests.

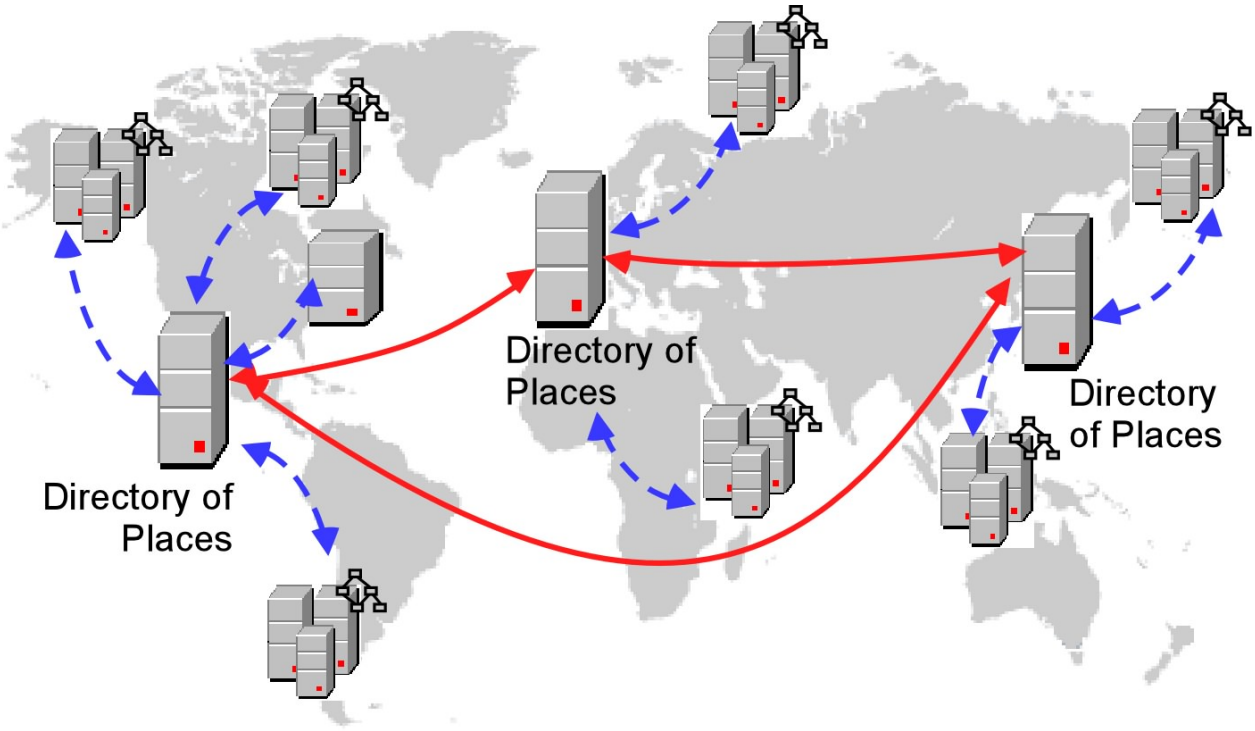

Figure 8: *Directory of Places* may be configured to replicate, making it indifferent which *Directory of Places* a *CmapServer* or *IndexServer* registers with.

To guarantee the service provided by the *Directory of Places*, the network allows for replication of *Directory of Places*. This means that in a network, there can be more than one *Directory of Places*, and they will all exchange information and keep each other up-to-date. Figure 8 shows seven *CmapServers* and an *IndexServer* registering via three *Directories of Places*. In this case, all *CmapServers* will receive the address of the IndexServer in order to send their indices for searches.

A *CmapTools* client contacting one of a group of replicated *Directories of Places* will receive information regarding all servers (*CmapServers* and *IndexServers*) registered with any of the *Directories of Places*. Figure 9 shows a user's *CmapTools* client registering with a single *Directory of Places* but interacting with all *CmapServers* in the network.

Scopes

The network described above assumes a set of *CmapTools* clients and servers (*CmapServers*, *IndexServer*, *Directories of Places*) linked together to allow sharing and collaboration between all of them.

An organization may need to have two parallel sets of clients and servers that it wants to handle as separate "logical" networks, even if they are running, for instance, on the same LAN. For example, at the Institute for Human and Machine Cognition the "development" network runs on computers standing side by side with the "Public" network (which

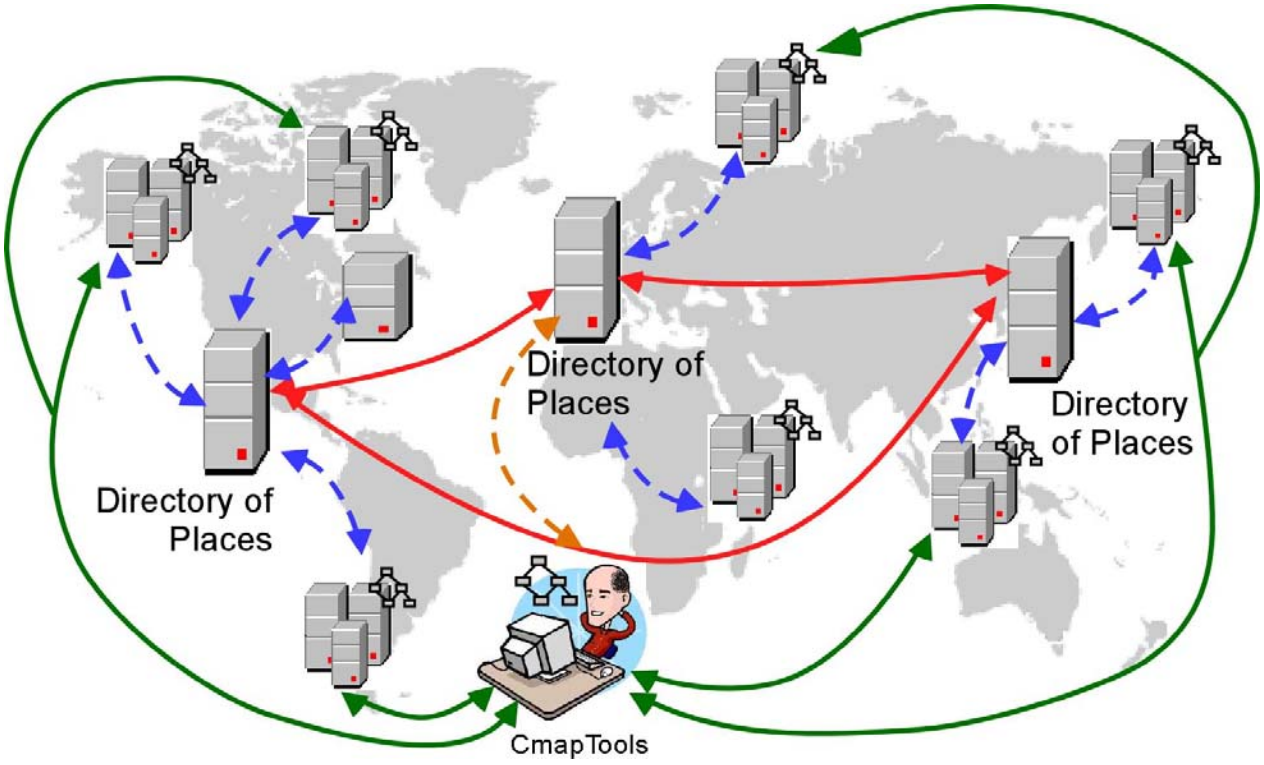

Figure 9: *CmapTools* client accessing various *CmapServers* located via the replication of the *Directories of Places*

includes, among others, the publicly available *CmapServer* "Public Cmaps"). Users developing knowledge models need not see the "development servers" and developers do not need to see "public" servers.

The separation between these two "logical" networks is achieved by defining each network to have its own "scope". A scope is the "identifier" of the network. Each *CmapTools* client, *CmapServer*, *IndexServe*r, and *Directory of Places* can belong to one or more scopes. They ignore any information that does not correspond to a scope to which they belong. The "Public" CmapTools network, for example, has scope "cmapdp".

The *CmapTools* software (client, servers, etc.) by default is configured to use scope "cmapdp", so any client installed will by default be part of this scope. Any of the programs can be configured to use another scope instead of, or in addition to, "cmapdp". So a user who decides to install a *CmapTools* client and a *CmapServer* in a LAN where there are other *CmapTools* clients running can change his/her *CmapServer's* scope, and add that new scope to his/her *CmapTools* client, and that way the client would see all the other servers, but his/her server would be "hidden" from the other clients. Similarly, and organization may decide to have all its clients and servers under a different scope, with no access to "cmapdp".

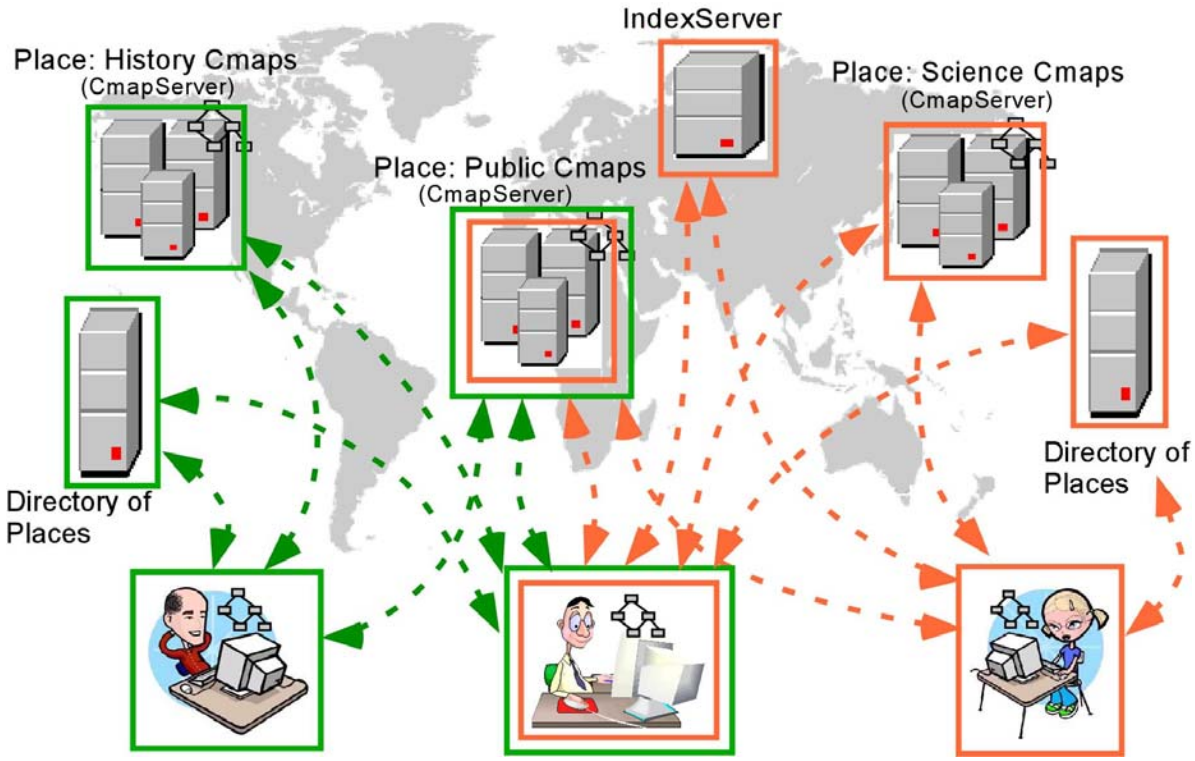

Figure 10. Two scopes (shown as green and orange) allow the users at the left and right to access different networks, while the user in the middle to access both networks. The "Public Cmaps" *Place* is part of both scopes.

Just as a *CmapTools* client can be in more than one scope, a *Directory of Places* can be configured to serve more than one scope, and *CmapServers* can be configured to belong to more than one scope.

Figure 10 shows two "scopes", represented by the orange and green colors (scopes are identified by their "name" in *CmapTools*, color is used in Figure 12 for illustration purposes). The green scope consists of the *Directory of Places* and a *CmapServer* at the left of the Figure, plus the "Public Cmaps" *CmapServer* in the middle. The orange scope includes the *Directory of Places* and *CmapServer* at the right, and shares the "Public Cmaps" *CmapServer* with the green scope. The user on the left has his *CmapTools* client configured for the green scope, and can access the two *CmapServers* and the *Directory of Places* registered in that scope. The user at the right can access the *Directory of Places* and *CmapServers* registered with the orange scope. The user in the middle has *CmapTools* configured for both green and orange scopes and can thus access all *CmapServers* and *Directories of Places* on both scopes.

The scope is not meant to be a security mechanism. Users can easily add and delete scopes to their *CmapTools* client or administrators to their *CmapServers* or *Directory of Places*, and there is one means to protect a scope (e.g. via passwords or other mechanism).

It is intended to be a means of partitioning or organizing the different sets of clients and servers according to the users' needs.

References

[1] Service Location Protocol Project (http://www.srvloc.org/)

[2] Service Location Protocol (http://www.ietf.org/html.charters/svrloc-charter.html)

[3] Jakarta Lucene (http://jakarta.apache.org/lucene/docs/index.html)