

2 Connecting The Server

Chapter Index

1. [General Procedure](#)

2.1 General Procedure

The API for the SCF server connection is defined in [DataConnection](#) class. This is a singleton, and its shared instance can be obtained with `getInstance()` static method. `DataConnection` can have only one connection to the SCF server at a time. The class is used by all underlying services, such as the [naming service](#) and the [data acquisition service](#).

The connection can be in three possible states: *connected*, *disconnected*, and *pending*. The *pending* status is used during transitions between two others statuses. The connection process can take a while, because before the server gets connected, it sends a request to the [Kerberos Module](#) on the client side to reads the user's ticket. `DataConnection` class monitors the communications and can automatically restore the link if it gets broken on some lower level (e.g., because of a network failure). If the server can not be reached for a certain period of time, the connection is closed. Thus, even if the server is not available on-line, the status can be *connected* until a timeout expires. The naming service requires the server to be on-line: all remote naming requests will block otherwise, or will fail if the server is disconnected. However, the data acquisition service does not require on-line communications at all: event if the status is *disconnected*, you can manipulate jobs and settings (of course, without getting any actual data...) `DataConnection` is invariant to the server name: if there is a connection to the server *A* and a bunch of data acquisition jobs running on it, you can disconnect *A* and connect another server *B*, without stopping and loosing these jobs. All the necessary information will be moved between *A* and *B* automatically.

Besides connecting the server, this class is used to enable and disable settings, and register [JobStatusListeners](#).

There are several means to control the connection.

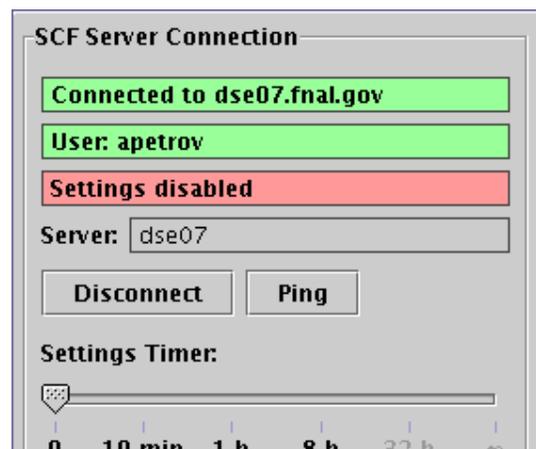
From a program, `connect()` and `disconnect()` methods can be used to open and close the connection; `enableSettings()` and `disableSettings()` enables and disables setting data to the data acquisition system.

For Application Framework, in the classes like [JControlsFrame](#) you can have a connection button similar to that for DAE. The button opens a connection dialog. The current SCF server name is defined by `scf.server.name` application property. The default and an actual user's values are stored in the property database on the server.

The SCF connection dialog can be set up in few steps:

- In the application property file (one per application), disable DAE connection support:

```
| dae.enabled=0
```



- In order to open an SCF connection automatically during the application startup, add one more line:

```
| scf.connect=1
```

- In the XML frame descriptors (one per frame) set a new status bar ID in the root `frame` element:

```
| <frame statusBarID="ScfStatusBar">
```

The SCF connection dialog can also be opened through *Tools|SCF Connection* menu item.

[< prev](#) [contents](#) [next >](#)
[security, privacy, legal](#)