### Defining Operations

**// class member function:**

```cpp
bool checkFoo(double arg) { .... }
```

**// in constructor or configureHook():**

```cpp
addOperation("checkFoo", &MyComp::checkFoo, this).doc("..."��);
```

**// Add C function:**

```cpp
addOperation("cFoo", &cFoo).doc("..."��);
```

**// Execute in own thread:**

```cpp
addOperation("checkFoo", &MyComp::checkFoo, this->OwnThread).doc("..."��);
```

### Calling Operations

**// class member:**

```cpp
OperationCaller<bool(double)> cFoo;
```

**// in configureHook():**

```cpp
if (getPeer("Foo") )
    cFoo = getPeer("Foo") ->getOperation("checkFoo");
```

```cpp
if (cFoo.ready() )
    bool ret = cFoo( 1.234 );
```

### Use Orocos-RTT cmake

You may mix these Orocos specific macros with standard CMake commands. A CMake TARGET name is created for each 'name' argument.

- `orocos_component( name files.cpp )`
  - Creates a component library

- `orocos_install_headers( headers.hpp )`
  - Installs headers in include/orocos/projectname during 'make install'

- `orocos_service( name files.cpp )`
  - Loads a service in a component

- `orocos_plugin( name files.cpp )`
  - Creates a plugin library containing one plugin

- `orocos_typekit( name files.cpp )`
  - Compiles a hand written typekit

- `orocos_generate_package( )`
  - Last statement which generates & installs a .pc file

### Deployment scripts

- `import("package")`
  - Imports all component libraries from a package located in your component path

- `path("prefix/lib/orocos")`
  - Adds a directory to your component path

- `displayComponentTypes()`
  - Prints all imported component types

- `loadComponent("Name","Type")`
  - Creates a new component (or proxy to existing component if Type is "CORBA")

- `loadService("Component","Service")`
  - Loads a service in a component

### TaskBrowser

- `.types`
  - Prints all known types

- `.services`
  - Prints all known services

- `cd Name`
  - Changes to a component

- `help [service|operation]`
  - Help

- `ls [Peer]`
  - Lists interface of current or peer component

- `provide <servicename>`
  - Adds a service to the current component