NIRSPEC

UCLA Astrophysics Program **U.C. Berkeley** **W.M.Keck Observatory**

Tim Liu **February 12, 1997**

NIRSPEC Software Design Note 10.00
Interface Between Echelle Format Simulator and NIRSPEC Server

1 Introduction

The original purpose of Echelle Format Simulator (EFS) was to provide a graphical interface for positioning the Echelle mechanisms to give the desired wavelength coverage. It is agreed that EFS should also be able to perform as an independent graphical user-interface. Like quick-look, EFS will be coded in IDL widgets. To communicate with the rest of the NIRSPEC software system, mainly the NIRSPEC server which is programmed in C, an interface mechanism is developed and discussed in this document.

2 Structure

NIRSPEC software is a client-server based system. Under the client-server architecture, multiple user-interface clients like GUIs, command line interfaces (CLIs), and quick-look tools can be executed concurrently and remotely. The clients communicate with the NIRSPEC server, which will run on a SUN workstation computer at the Observatory, via TCP/IP network.

In order for EFS to function efficiently, it is best to treat it as an independent client with a direct inter-process communication (IPC) capability with the server, instead of relying on the other client such as the GUI to route a message, though the latter method is simpler in implementation.

EFS will be programmed in IDL. There are several mechanisms for IDL-based programs to communicate with other software. We will use the IDL CALL_EXTERNAL function to interface EFS to NIRSPEC software, similar to the IDL-coded quick look program, CALL_EXTERNAL allows one to call external functions written in C from inside IDL programs and thus pass values between the two different software systems. To make this happen, an IDL asynchronous event loop is constructed and the external C functions are called by EFS using time interrupt. The external function routines must be compiled as a shareable object library in order for IDL to call. In addition, there should be a dedicated process sitting between EFS and the NIRSPEC server to handle the bi-directional communications. This control program and the external functions library form a "gateway" to non-IDL software. Such an interface architecture is illustrated in Figure 1.

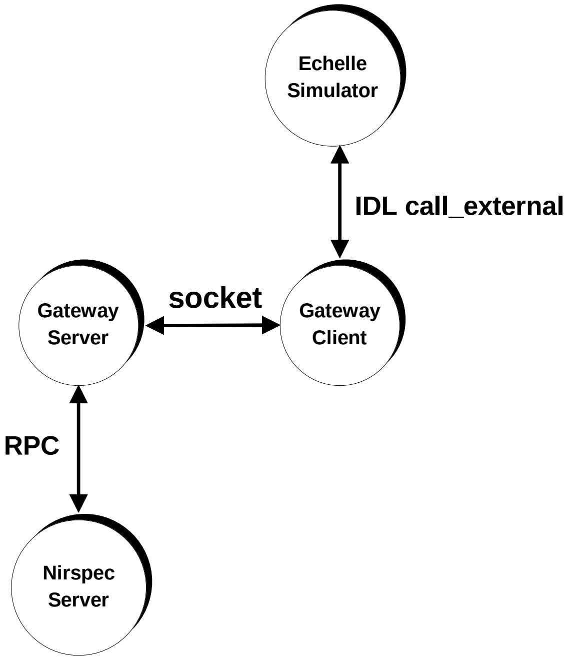

Figure 1 Interface scheme between EFS and NIRSPEC server

As can be seen, the gateway program consists of a server part and a client part which communicate with each other via a UNIX socket. The gateway client contains routines to talk to the server. To EFS, these socket routines are external functions and are called through IDL CALL_EXTERNAL function. The main part of the gateway server is an asynchronous event loop which processes socket events from EFS and KTL events from the NIRSPEC server. The gateway server and the NIRSPEC server are interfaced using remote procedure call (RPC) mechanism. For clarity, the functions of the gateway server and client are listed below:

Gateway server:

- listen to requests from both ends (KTL events and socket events)
- parse command/keyword strings from EFS and send to the NIRSPEC server
- handle NIRSPEC server broadcasts using callbacks and pass to EFS

Gateway client:

- provide a socket channel which links both the gateway server and EFS via CALL_EXTERNAL

Such a communication scheme works as follows: when EFS sends a command/keyword to the NIRSPEC server, the command message is passed to the gateway client via an CALL_EXTERNAL function call and then to the gateway socket server which in turn parses it and sends to the NIRSPEC server using a ktl_write() call. On the other hand, when the NIRSPEC server broadcasts a keyword, the gateway server will send it to the socket client by invoking a callback function and the client will then pass the keyword to EFS via CALL_EXTERNAL.

3 Other Issues

For simplicity in implementation, a command/keyword message sent by EFS is encoded in such a way that each CALL_EXTERNAL function call sends a single command/keyword. To start an exposure, about a dozen keywords need to be sent out in 1 second or shorter period of time. This requires that the EFS event loop be capable of making CALL_EXTERNAL calls at 10 or 20 Hz. Test shows that this rate can be achieved without noticeable downgrading of IDL GUI performance like mouse movement. In fact, it is possible to implement a variable loop rate such that the normal rate is set to 5 Hz and will be changed to 20 Hz or even higher when an exposure is to start.

The EFS gateway is a stand-alone program and will be launched along with GUI, CLI, quick-look, and EFS from a shell program.