Keck Adaptive Optics Note 652

Keck Next Generation Adaptive Optics
Real Time Computer Interface Concept

Jimmy Johnson, Erik Johansson

Revision 1: April 14, 2009

1. Introduction

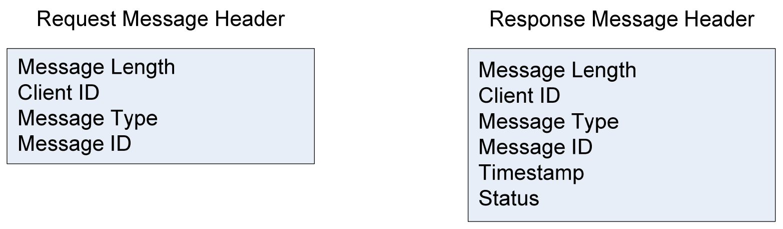
This document describes a concept for the interface between the AO Control System and the Real Time Controller (RTC) for the NGAO system. A similar type of interface has been successfully used in the past by Jimmy Johnson at Measurex/Honeywell and was adopted as a standard by the company. It provides a robust and flexible means of bidirectional communication of commands, status, and parameter data. The concept is presented here in a primarily pictorial fashion as a starting point for more detailed design efforts.

2. Basic Interface Description

The interface was originally designed for a client-server architecture with multiple clients. For the case of NGAO, the RTC implements the server and the AO Controls function implements a single client. This discussion assumes a generic communications layer between the two, which will most likely be implemented using Ethernet and TCP/IP. The basic interface consists of messages that are sent back and forth between the client and the server. The message sent from the client to the server is called a "Request" message and the message sent from the server to the client is called a "Response" message. All communication is initiated by the client, and the server only sends messages in response to client requests. Each message consists of a message header and a message body. The message headers are standardized, one type for request messages and another type for response messages. The message body contents are specific to the particular request or response, and typically consist of symbols (parameters), their associated meta-data, and their values. It is the responsibility of the client to ensure that all data are formatted appropriately for the server (data type, byte order, etc.) and to reformat data received from the server, as required.

3. Message Headers

The structure of the message headers is shown below in Figure 1. For simplicity, we will not concern ourselves with the data types or field alignment in this discussion.



Request Message Header	Response Message Header
Message Length	Message Length
Client ID	Client ID
Message Type	Message Type
Message ID	Message ID
	Timestamp
	Status

Figure 1: The structure of the message headers.

The message header fields are defined as follows:

- Message Length: The length of the entire message (message header and message body) in bytes. This includes the message length field itself. A single message may be sent out in

- Client ID: A unique identifier representing the client. The server does not use this field and merely copies it to the Client ID field when generating a response message. For our purposes, this field may not be needed. Our current thinking is that there will only be a single client, the AO Controls function.
- Message type: This field identifies the message type. The server uses this field to identify the requested service. Message types are described in more detail below.
- Message ID: A unique ID assigned by the client to each request message. The server copies this field to all messages generated in response to a request. The client may use the Message ID field as desired, the most obvious use being synchronization between requests and responses. Other possible uses include storing pointers or hash code values for use by the client when it processes the response messages.
- Timestamp: The timestamp field contains the server time at which the response was generated.
- Status: A field generated by the server to indicate the completion status of the request. The value will either indicate success or an error condition. Note that an error condition need not imply a fatal error, but could indicate a partial success.

4. Message Types

The message body is dependent on the type of message being sent, so we now turn our attention to the different message types used in the interface. The message types described below are candidate types and may not all be required or implemented for the RTC interface. Moreover, we may choose to implement additional types, as needed. Each message type described below is implemented as a request message from the client to the server and an associated response message from the server to the client. The following message types are currently defined:

- Logon: A message used to establish a connection between the client and the server. Although it is primarily intended to implement security in the client interface, the logon message could be used in NGAO to initiate communications between the AO Controls function and the RTC.
- Logoff: A message used to terminate a connection and all communications between the client and the server. Upon receipt of this message, the server will cancel all pending requests and delete all temporary internal data structures used to support the connection. For NGAO, this could be used when putting the AO system into a standby state.
- Symbol lookup: The symbol lookup message is used to request meta-data for any of the symbols used by the server. Assuming that all the symbols used are well defined and known to both sides of the interface, this message type may not be required for the RTC interface.
- Data read: A message used to read data from the server. The data transfer for a data read can be configured to be on-demand, periodic, or event based:
    - On-demand: The client sends a request message and the server replies with a response message.
    - Periodic: The client sends a request message which the server uses to configure the periodic read. The server then sends response messages with data at the requested periodic rate.
    - Event based: The client sends a request message which the server uses to configure an event based read. The event is represented by a change in a server parameter. Upon sensing the change, the server sends a response message with the requested data. The event based trigger could be used to permit the RTC to communicate a particular kind of real-time event to the AO Controls host function.
- Cancel: A message used to cancel a previous periodic or event based read request.
- Data write: A message used to write parameter data from the client to the server.
- Server status: A message used to get the current status of the server. It can also be used as a heartbeat mechanism to indicate to the server that the client is still alive.

5. Message bodies

This section describes the message bodies defined above in sufficient enough detail that the interface described in this document can easily be specified with real data types and field lengths and used as the basis for an interface design and implementation. For each message type, the success of the request is returned in the status field of the response header along with a timestamp.

5.1. Logon

The logon request and response messages are shown in Figure 1 below. The request has four parameters in addition to the header, while the response consists of only the response message header. The request parameters are defined as follows:

- Logon Option: Used to determine whether the server will apply timeout control to this connection. Under timeout control, if no requests are received during some default timeout period, the connection is terminated by the server.
- Version Number: This field can be used to ensure synchronization between the version ID of the client and sever interface SW.
- User: The user name (not required for NGAO).
- Password: The password (not required for NGAO).

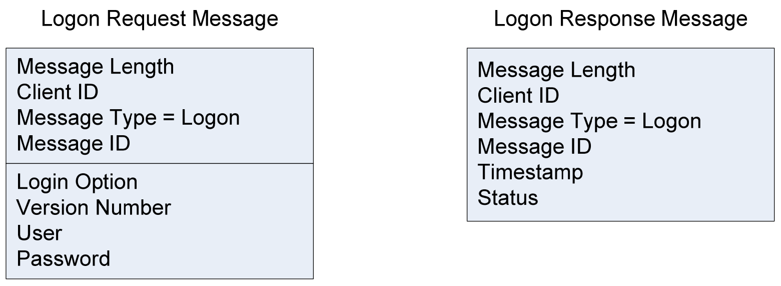


Figure 1: The Logon request and response message bodies.

5.2. Logoff

The logoff request and response messages are shown in Figure 2 below. The request and the reply consist solely of the message headers.

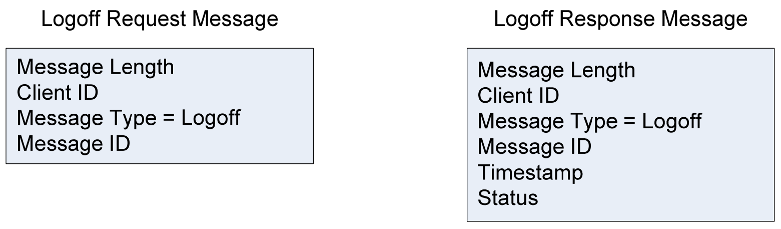


Figure 2: The Logoff request and response message bodies.

5.3. Symbol Lookup

The symbol lookup request and response messages are shown in Figure 3 below. The request body contains the number of symbols to be looked up by the server, followed by a list of the symbol names. The server response consists of the number of symbols found followed by the symbol names and the meta-data for each symbol.

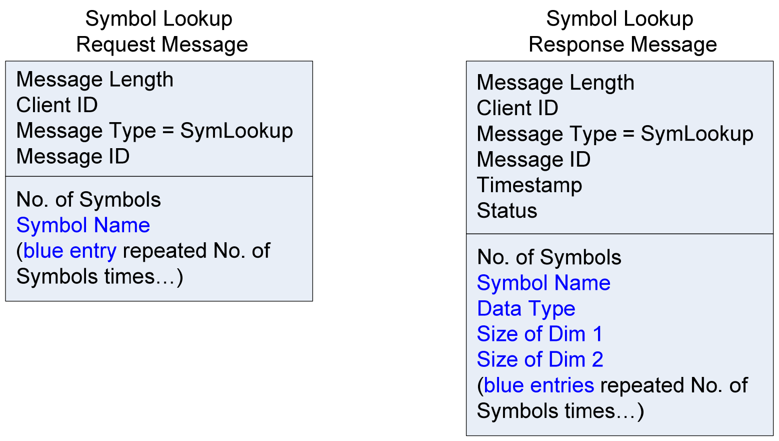


Figure 3: The Symbol Lookup request and response message bodies.

5.4. Data read

The data read request and response messages are shown below in Figure 4. The read request has two responses: an initial status response to acknowledge the read request, and a read response which contains the data. The two response message types are required to support periodic and event-based reads. In those cases, a single read request sets up the read and is acknowledged by a status response message. Subsequent to that, every time the trigger or periodic conditions are met, a read response message is sent by the server. The Message ID field is used by the client to correlate the response messages with their originating request message.

The read request message consists of a trigger name, trigger type, and rate, which are used to set up event-based or periodic reads, followed by the number of symbols requested and a list of their names. The read status response returns information on the trigger followed by the number of symbols to be returned by the read along with their names and meta-data. The meta-data includes an offset field which is the offset in bytes of the data from the beginning of the read response message(s) that will follow. The read response message returns the number of symbols in the data, followed by the data status (read status) and the data for each symbol.

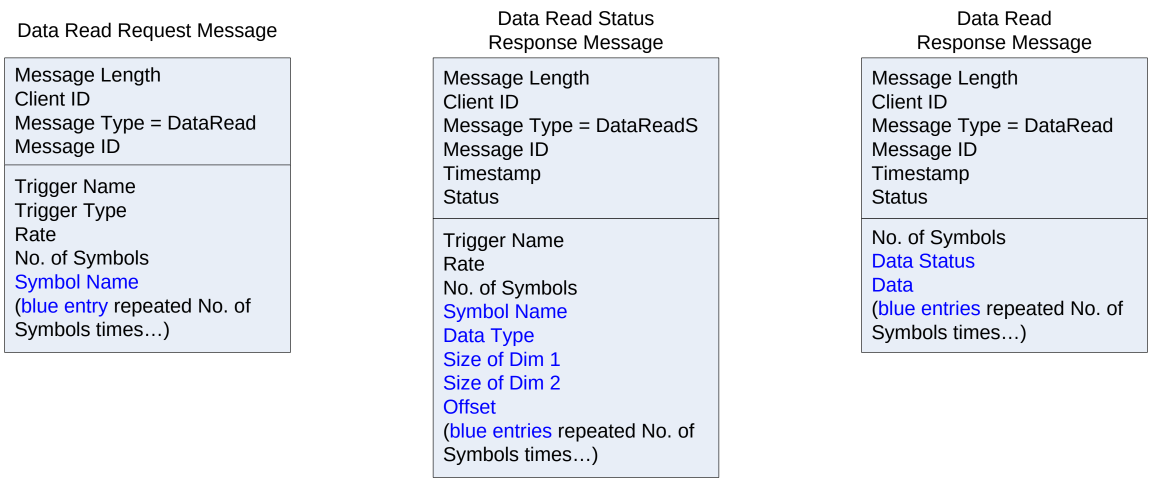


Figure 4: The Data Read request and response message bodies.

5.5. Cancel

The cancel request and response messages are shown in Figure 5 below. The request and the reply consist solely of the message headers. The Message ID field is used to indicate to the server which read request to cancel.

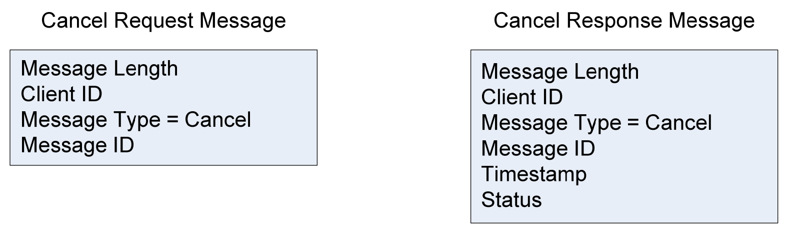


Figure 5: The Cancel request and response message bodies.

5.6. Data write

The data write request and response messages are shown below in Figure 6. The write request message consists of the number of symbols followed by a list of the symbol names and their meta-data. The meta-data is used by the server to validate that the data from the client is consistent with what it expects. The data write response message consists of the number of symbols followed by a list of their names and the write result statuses.

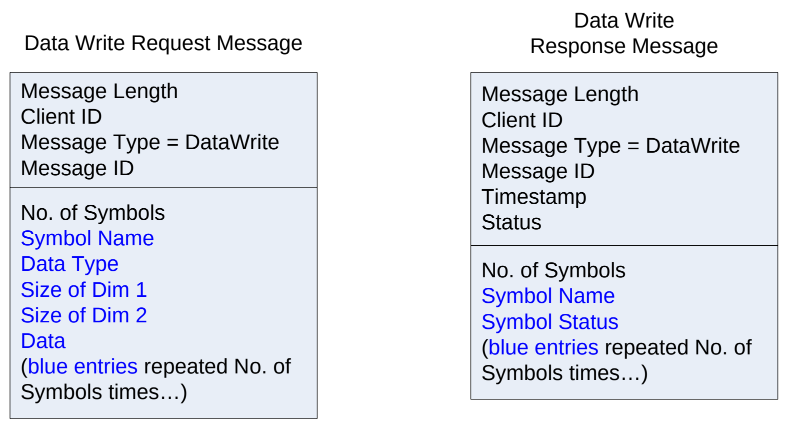


Figure 6: The Data Write request and response message bodies.

5.7. Server Status

The server status request and response messages are shown below in Figure 7. The status request message consists of only a message header. The status response message consists of a status word followed by the number of outstanding requests that are pending.

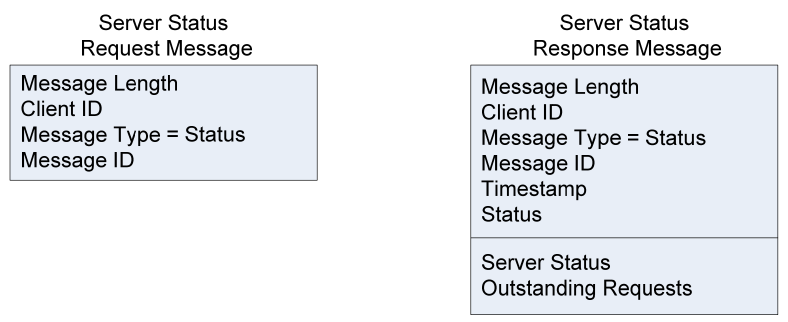


Figure 7: The Server Status request and response message bodies.