is being severely degraded. A set of commands facilitates transparent switching of servers when a server's load gets too high.

The OTHERSERVERS command in conjunction with the LOAD command can be used to find a less loaded server. Then, the client can perform this series of steps: (1) Connect to the less loaded server. (2) Issue a REQUESTDUMP to the less loaded server, giving the host of the client's current server. (3) Issue a GIVEDUMP to the current (more loaded) server, giving the less loaded server's host and the port number it returned from the REQUESTDUMP call. (4) Scrap the connection to the more loaded server and use the less loaded server. The REQUESTDUMP/GIVEDUMP calls cause the more loaded server to give the less loaded server a *brain dump* which contains all the state *relating to this particular client* the more loaded server has, enabling transparent switching of servers. The format of the dump itself is not described here, but contains such things as which forms the client has checked out and any cached information the first server has.

On Table 0

This table contains a list of all defined Phaedo server commands.

The numbers under *Possible Errors* refer to the Defined Error Table above. Note that some errors, specifically, the syntax errors (500, 501, 502), the "server overloaded" errors (399, 499), and those errors whose first digits are 6 (and sometimes 7), are possible in response to almost any command and are not always listed.

When an argument or return value is described as a form header, form description, or form contents, refer to *Form Headers, Descriptions, and Contents Over the Data Stream* on page 7.

Table 0: Defined Phaedo Commands

Command *Authorization Required*	*Arguments*	*Normal Return Values*	*Description*	*Possible Errors*
OPEN	*v*	none	Must be the first command the client sends the server (but see LOAD). *V* is a string which is used to differentiate between versions of the protocol. The current protocol version string is "2.11".	399 709 [the connection closes if error 709 is produced]
LOAD	none	*load*	Returns the current load on this server. This command may be used in lieu of OPEN to open a server connection; in this case, however, the server closes the connection once *load* is sent. *Load* is an integer between 0 and 100, with 100 being maximum load.	none
OTHERSERVERS	none	*server-host port#* *server-host port#* ...	Returns a list of other Phaedo servers available, for when this server is overloaded.	none
REQUESTDUMP	*server-host*	*port#*	Requests that this Phaedo server wait for a brain dump from a server on the specified host, and returns the port number which the specified server should connect with.	560
GIVEDUMP	*server-host port#*	none	Tells this Phaedo server to give a brain dump to the specified server. The dumping server (this server) closes its connection via an implicit call to QUIT when the dump is successfully completed. See *Server Load and Brain Dumps* on page 8.	560
AUTHENTICATE	*ticket*	none	Authenticates the client to the server using Kerberos. *Ticket* is a Kerberos ticket, normally encoded using the binary quoting mechanism.	502
UNAUTHENTICATE	none	none	Remove the client's authentication.	none
DESCRIPTION description	*name version#*	The form description	Returns the form description for the form class named *name* of version *version#*. Versions are integers, starting generally at 0.	510 511 521 530
DESCRIPVERSIONS description	*name*	*version#* *version#* ...	Returns the list of available version numbers of the fom description for the form class named *name*.	510 530
GET read	*formclass id*	001 (linecount) field-value pair field-value pair ...	Returns the data contained in the *formclass* form named *id*.	510 513 521 530 691

Table 0: Defined Phaedo Commands

Command *Authorization Required*	*Arguments*	*Normal Return Values*	*Description*	*Possible Errors*
GETTABLE read	*formclass field-name [field-name...]*	001 (linecount) value1 [value2...] value1 [value2...]	Returns tab-separated values of all requested fields in forms of this class	530 691 514 513
NEW new	*formclass idprefix*	*id* The form header The form contents	Returns a new form of class *formclass*. Also creates a new ID string for the form which is guaranteed unique among forms named *name*, and returns it in *id*. *Id* will have the supplied prefix, *idprefix*. Automatically checks out the new form.	510 521 532
NEWNAMED new	*formclass id*	The form header The form contents	Returns a new form of class *formclass* and ID *id*. Automatically checks out the new form.	510 514 521 532
CHKOUT checkout	*formclass id*	none	Checks out the *formclass* form named *id.* A form must be checked out before a call to PUT. Checked-out forms cannot be checked out by anyone else.	510 513 521 531 541
CHKIN	*formclass id*	none	Checks in the *formclass* form named *id.*	510 513 440
ALLIN	none	none	Checks in all the forms which the client has currently checked out.	none
LSCLASSES list	*pattern*	*formclass* *formclass* ...	Returns the list of form classes the client could examine whose names match the regexp *pattern*. This can depend on the client's current authentication.	520
LSFORMS list	*formclass pattern*	*id* *id* ...	Returns the list of *formclass* forms whose names match the regexp *pattern.*	510 520
LSFORMHEADERS list	*formclass pattern*	A form header A form header ...	Returns a list of the headers of *formclass* forms the client could read whose names match the regexp *pattern*. This can depend on the client's current authentication.	510 520 535
PUT write	*formclass id node* The form contents	none	Puts the *formclass* form named *id* into the database. The form is sent along to the node number *node.* The values of the form's fields are also given.	440 510 513 521 531 550 551
DELETE delete	*formclass id*	none	Deletes the *formclass* form named *id* from the database.	440 510 513 521 534
MOVE move	*formclass id node*	none	Moves the *formclass* form named *id* to the node number *node.*	440 510 513 521 533 550
CHOWNER move	*formclass id owner*	none	Changes the *formclass* form named *id*'s owner to *owner.*	440 510 513 521 533 552