Christopher DuHadway

Garin Strader

Derek Dixon

Connor Parrish

COMMUNICATION PROTOCOL DESIGN

-Document Format:
Protocols are described more generally in sections I-VI and more thoroughly in section VII, with a summary of important points in section VIII.

I. Operational Overview

Our design leaves most existing spreadsheet functionality in the client. The server and client will use TCP for networking. The client sends spreadsheet changes to the server for resolution. The server is responsible for resolving race conditions on received commands. The server performs legal requested changes on a master copy of the spreadsheet. The server then sends clients messages containing changed cells. The client is responsible for calculating the spreadsheet values, and the user is expected to resolve formula errors and circular dependencies.

II. Server Startup

The specifics of how the server starts up is out of scope of this document, but the server must not rely on specific input from a client in order to start up. On startup the server must have access to the persistent spreadsheets and choose a specific port to listen to for messages. It is the responsibility of the creator of the server to inform the users client side which port was chosen. After startup the server must be available to accept new connections and ensuing commands from multiple clients. The server must have a method of holding commands waiting to be processed (e.g. a buffer); it must also hold a history of edits for each cell and a global edit history.

III. Server shutdown

Implementation of the server shutdown is out of the scope of this document, though it is expected that shutdown is initiated through a command line argument (or a GUI if you are so inclined), and the server will store all spreadsheets in some form to maintain their persistence past shutdown. Command history is not expected to be persistent. The server must send a disconnect command to notify clients that it is shutting down.

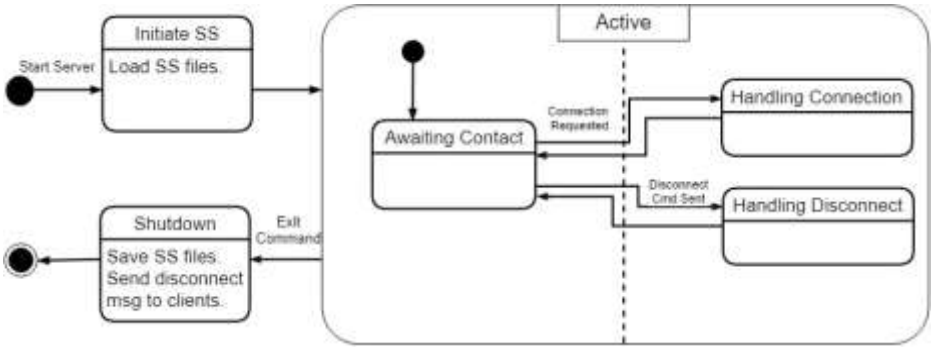


Figure 1 - Server Startup and Shutdown

IV. Initial Connection

In order for a client to connect to a spreadsheet, the client sends a spreadsheet filename to the correct port using the 'Open Existing' command. Assuming an active server, the client then receives one of two things: a message containing the entire contents of the requested spreadsheet when it exists, or an error message if the spreadsheet does not exist. The client will be disconnected if an incorrect filename is sent and must reconnect to send another filename.

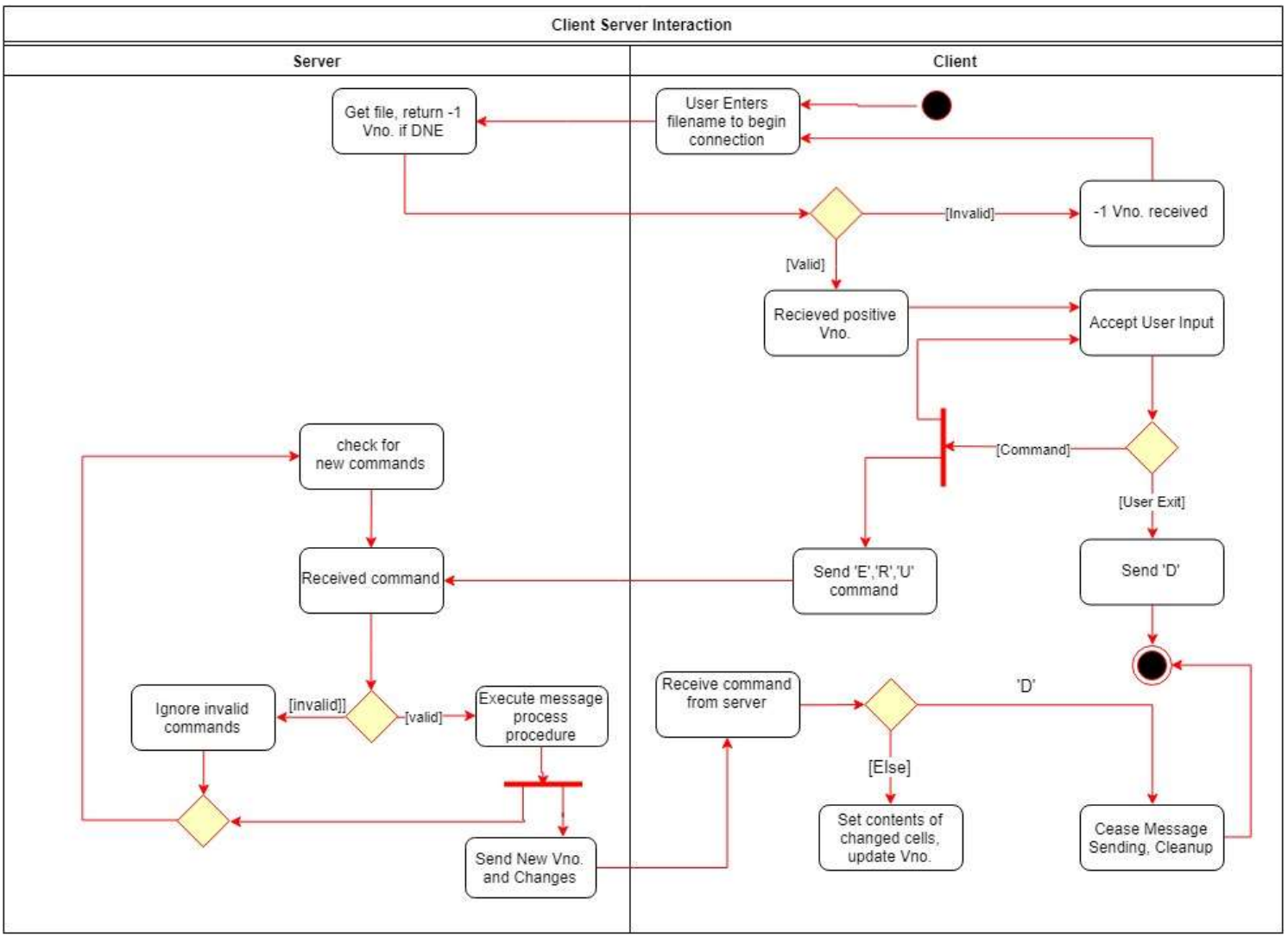
The client may also create a new spreadsheet without being disconnected. See 'Open New' in Message Specifications for more detail.

V. Command Messages

Once connected, there are four valid message types the client can send to the server. Those are edit, undo, revert, and disconnect denoted by the command types 'E', 'U', 'R' and 'D', respectively. All messages consist of a string of characters. The first character in the message denotes the command type. If the first character is not a valid command type, the message is ignored. If the command is incomplete it will remain in the server's command queue (most likely a buffer) until it is completed by the next message. Sending an open command at this stage is invalid and will also be ignored.

VI. Undo and Revert Process

'Undo' commands will undo the last edit to a spreadsheet (from the global history). 'Undo' is *not an edit* to the spreadsheet itself. 'Revert' will undo the last edit to the currently selected cell (local history). Since 'Revert' *is an edit*, it is stored and can be undone using 'Undo'. 'Revert', however will not undo another 'Revert'. As 'Undo' is not a spreadsheet edit, 'Revert' cannot undo an 'Undo' either.



*Message process procedure given in summary.

Figure 2 – Client Server Interaction

VII. Message Specifications

Client to Server messages:

This table provides the specific format and an example of messages sent to the server from the client.

Message	Format	Example
Open Existing	O\0<spreadsheet_name>\0	O\0Spreadsheet_1\0
Open New	N\0<spreadsheet_name>\0	N\0Spreadsheet_1\0
Edit Command	E\0<version>\0<cell_name>\0<contents>\0	E\0A2\042\0
Undo Command	U\0<version>\0	U\042\0
Revert Command	E\0<version>\0<cell_name>\0	E\042\0AB23\0
Disconnect Command	D\0	D\0

Parameters:

- spreadsheet_name: The name of the spreadsheet to be opened by clients.
- version: The client's current version number.
- cell_name: The format of a *cell_name* is one or more capital letters followed by one or more numbers.
- contents: The new contents of the indicated cell.

Open Existing:

Once the client and server have connected, the client sends this message to specify which spreadsheet the client wishes to edit. If the filename name does not specify an existing spreadsheet, the server will send an update with a version number of '-1' and no cell name or contents, then terminate the connection. The server expects no specific filetype. The server will attempt to load the exact message sent to the server. It is the obligation of the user initiating the server to clarify what spreadsheets are available to the clients.

Open New:

A client may send this as its first message in order to construct a new spreadsheet. As with 'Open Existing', extensions will not be checked. The filename will be the exact bytes following 'N\0' until another null terminator is reached. The client will then receive the standard 'Update'

command from the server with version number zero and no cells or contents. If there is a naming conflict with an existing file, this command will function exactly as ‘Open Existing'.

Edit Command:

This command is a spreadsheet edit sent when a client wishes to edit the contents of a cell. The contents field contains the requested change. Note, the *contents* field is not the calculated *value* of the cell. If the edit was to clear a cell, then the format will have *two null terminators* following cell name*,* meaning the *cell’s contents are empty.*

Undo Command:

This command is not a spreadsheet edit, and is sent when a client wishes to undo the previous spreadsheet edit. When an undo command is executed by the server it steps back in global history once, removing the last edit from global history and the individual cell’s history. If undo commands are called sequentially, then the server continues stepping back in history. If the server receives an undo command with an outdated version number, the server does not undo anything.

Revert Command:

This command is a spreadsheet edit sent when a client wishes to revert to a previous change in the currently selected cell. It is recorded by the server’s spreadsheet edit histories just like ‘Edit’. It may not revert another revert, or an undo.

Disconnect Command:

This command is sent by a client before they disconnect from a server. There are no fields for this command.

Server to Client messages:

Message	Format	Example
Update	U\0<version>\0<cell_name_1>\0<contents_1>\0 … \0<cell_name_n>\0<contents_n>\0	U\042\0A1\034\0A2\012\0
Disconnect	D\0	D\0

Update Message:

This message is sent to all clients every time an update to the master spreadsheet is made, and contains the new values for only changed cells. The message consists of a version number (for coordination) followed by pairs representing cells. In each pair, the cell name comes first, followed by the contents of the cell. All cell name fields follow the naming rules specified in the client Message Specifications. The contents fields contain cell *contents* and not calculated *values* of cells.

Disconnect:

This message is sent to all clients when the server shuts down. There are no fields sent with this message.

Conflict Resolution:

Three conflicts may occur when the server processes commands. Partial Commands Received refers to the case where a client's connection prevents them from being able to send their full command within a single packet. Simultaneous Commands Received refers to the case when several commands are received at once and must be processed together. Version Conflicts result when the version of an incoming command does not match the Server's current spreadsheet version.

Partial Commands Received:

Individual commands received from clients with high latency are likely to come through multiple packets sent to the server. When a command is sent to the server that appears to be missing a defined parameter from a command defined above, the message will remain in the server's storage until the rest of the message is received from the client.

Simultaneous Commands Received:

When several commands are to be processed, the server first executes a maximum of one 'Undo', ignoring all other 'Undo' commands. Next the 'Edit' and 'Revert' commands are resolved. Ignore all 'Edit' and 'Revert' commands that would change the same cell that the undo command has changed. If any remaining 'Edit' or 'Revert' commands would change the same cell as another 'Edit' or 'Revert' command, ignore the command with the oldest version number. If the version numbers are the same, execute the first 'Revert' and execute the first 'Edit' if and only if there is no 'Revert' in the same cell. Next, execute all remaining edit and revert commands.

Version Conflicts:

If an out-of-date version is received from the client, the server applies the edit on the master spreadsheet if it does not conflict with any other current commands and otherwise ignores it. The server

then sends the entire spreadsheet contents to the client, with or without their edit. When a client receives a message more than one version advanced from its current version, that client wipes its current copy of the spreadsheet. It then populates a new spreadsheet from the message.

VIII. Summary:

Process Message Procedure:

Current Version:

Execute maximum 1 'Undo'

Execute 'Revert' a maximum of once per cell if unaffected by current 'Undo'

Execute 'Edit' a maximum of once per cell only if unaffected by current 'Undo' or 'Revert'.

Outdated:

Execute outdated commands only if non-conflicting with current commands according to previous procedure.

Important to Remember:

- Client is mercilessly disconnected for improper command usage on startup
- Open commands are ignored during sheet use.
- Command history must be stored globally and locally for each cell
- 'Revert' is an edit, 'Undo' is not.
- 'Revert' only reverts 'Edit'
- 'Undo' is destructive to the history
- Server only saves version number and populated cells
- Calculations and errors are handled by the client and user