

# Game Use Cases

For now, all interfaces will be implemented in MFC in a separate executable to save programming effort. A “session configuration” will be contained in a single ini-style text file and represents the entire communication between the menu application and the game application; the menu creates it, and the game reads it.

Launch syntax:

<game exe> <config file>

i.e.

client.exe sessioncfg.ini

This saves us the hell that is command line parameters and is much more extensible. Also, it separates session setup from execution in a nice generic way; later on we could always move the menu system into the game without any complicated module interaction.

## [UC1] Generic

Underlined words denote interface objects; see below for more details on each.

- Main menu appears (MFC)
  - Single player* (disabled)
  - Multiplayer*
  - Options* (just runs Settings.exe)
  - Credits*
  - Quit*
- User clicks on Multiplayer.
  - New menu appears:
    - *LAN*
      - Immediately jumps to Game List; EnumHosts called on local network and directplay reports the hosted games
    - *Internet*
      - Immediately jumps to Game List. One of two things then happens:
        - Primary Goal*
          - Game list is empty; when the client clicks on *Connect*, a dialog box asks for server IP. Polls the server for an active game, and if found is added to the Game List.
        - Secondary Goal (if there's time)*
          - Connect to a static, dedicated list server to retrieve a list of IP's with active games. “List server” is not to be confused with “game

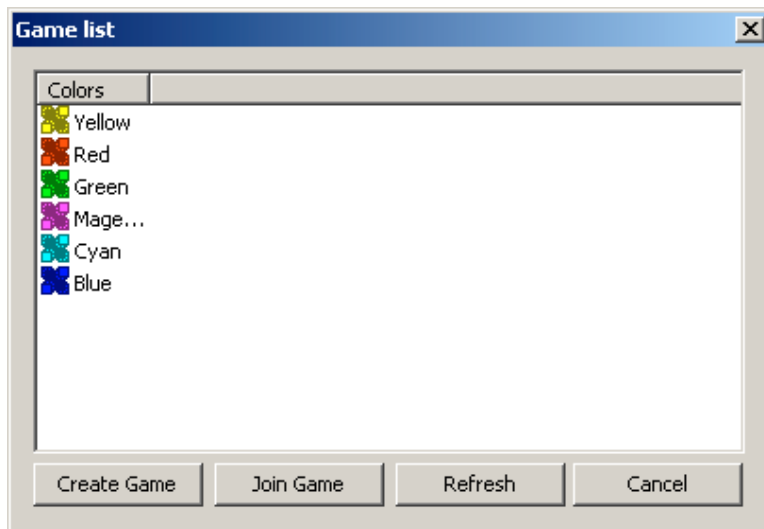
server". Game servers submit their IP's to the list server, and game clients query the list server for the list.

- User clicks *LAN*, selects a server and clicks *Connect*.
- The Creation Interface is displayed. For the client user, all interface controls are disabled except for the chat window, the ship selection controls, and the *Start* button. The game host is the only user who may set up the game type.
- Server user configures the game. Once all players have hit *Start*, the menu app creates the session configuration file and launches the game executable, and then waits. When the game executable terminates, the menu app will revert back to the game list.

NOTE: Menu app net connections are independent from game net connections. For simplicity, there will be three packet types for the menu app: the server packet, client packet, and chat packet. The server packet will contain the states of all host-editable Creation Interface controls, and similarly the client packet will contain the client-editable control states. The chat packet will be a variable length packet containing just the message (the host will implicitly know the address that sent the message). The host machine will receive and route all of these packets to the clients.

## Interfaces

### Game List



Note: umm, instead of "colors", there'll be "Name", "Ping", "# Players", "IP", etc.

### Creation Interface

The screenshot shows a window titled "Bob's game - low pings only!". The interface is organized into several sections:

- Player Settings:** A table with three columns: "Player", "Ship", and "Color". It contains eight rows. The first three rows have the player names "Twibble Klu", "Drip", and "PirateDan" in the "Player" column. The remaining five rows have empty "Player" cells. Each "Ship" and "Color" cell contains a dropdown menu.
- Game Mode Selection:** A vertical list of radio buttons on the right side, including "Hypermelee", "Soccer", "Tag", and "Sumo".
- Options:** Two checkboxes labeled "Option #1" and "Option #2".
- Time limit:** A text input field with the label "Time limit:" above it.
- Planet:** A dropdown menu with the label "Planet:" above it.
- Sample list box:** A large empty rectangular area with a dotted border and the text "Sample list box" at the top left.
- Send:** A button labeled "Send" next to a wide text input field.