FlightGear Multiplayer Extensions

Note

All these features involves multi-player fgfs. Notably, AIs and scenarios are not local, but should always be run thru a multi-player server (and should be run on an Internet server, probably not a user's PC).

Virtualized Reality Flights

Fetch data from http://flightaware.com/live/ and inject into fgfs servers the real flights. This may need a further delay, to be able to simulate trajectories.

Virtualized Reality Scenery

Fetch satelite pictures from GoogleMaps to use them as terrain in fgfs. The pictures should be kept in a cache for fgfs to use. Requests should be done depending on the fly paths of the users of fgfs, plus possibly some "random" walks. See Scenery Server.

Automatic ATC

We would need an automatic ATC, able to direct automatically departure and arrival, both of AI and user flights, and also taking into account the (forced, prioritary) virtualized reality flights.

Virtual Cost Tracking

It would be nice to have a system to track the costs of kerozen consumption, and of material destructions (both flying and non-flying), per each pilot and virtual airline or squadron.

Virtual Squadron & Virtual Airlines

There are already virtual airlines running with fgfs.

```
http://wiki.flightgear.org/index.php/Category:Virtual_airlines
```

I would like a system to automatically organize and run a virtual squadron. It could also be derived to run a virtual airline, ignoring military specific features. Some military operations are entirely similar to civilian: troop transport, cargo transport.

A squadron commander would create a squadron, specifying its base airport, and its fleet. (Several squadrons may be based on the same airport). The base airport ought to be a military airport in real life, to avoid interferences with civilian users, and victualized reality flights.

Missions can be defined, and mission famillies can be generated automatically, including plane requirements, flightplans and objectives. Mission are scheduled for a squadron. Notice that mission can involve different airports than the squadron airport (in case of "campains").

Pilots may register to a squadron, and receive their orders automatically from the next scheduled mission (optionally either in advance via email, or in real-time when they're on-line), or may choose a specific mission. They may choose to filter out the missions they want to be assigned to (type of plane, type of mission, time, etc).

Unassigned places in a scheduled mission are filled by automatic (AI) pilots.

There should be a mission control system that would track the execution of a mission (either automatic, or staffed by users), and a report is issued on the mission completion (with update of the pilot's logbook).

FlightGear Patches

It would be nice to add ecl into FlightGear, so that we can write extensions in Common Lisp, instead of nasal.

Or perhaps, the other way around, make flightgear a library accessible thru CFFI to any other CL implementation.

FlightGear Tools Developed

dump-net-fdm

Files:

- com.ogamita.fgfs.dump-net-fdm.asd
- binary-packet.lisp
- net-fdm-24.lisp
- dump-net-fdm clisp script.

Usage:

Launch fgfs with:

```
fgfs_period=20
slaveIP=localhost
fgfs --native-fdm=socket,out,${fgfs_period},${slaveIP},5510,udp  ...
```

Launch dump-net-fdm in a terminal:

```
xterm -e ./dump-net-fdm
```

Output sample:

```
--------------------------------------------------------------------------------
long     -2.130152 alti    10.05      #engi 4        Engines
lati      0.652921 agl      3.13      estat   2        0        0        0
                   climb   -0.001     rpm      0.00    0.00    0.00    0.00
alpha    -0.002    stall    0.        flow     0.04    0.00    0.00    0.00
speed     0.1                        fpres    0.00    0.00    0.00    0.00
wind      5.068                      egt     59.19    0.00    0.00    0.00
time  1302294800.0                   cht      0.00    0.00    0.00    0.00
warp       28980.0                    mposi    0.00    0.00    0.00    0.00
visi  16093.4                        tit      0.00    0.00    0.00    0.00
  eleva   0.000    etrim   0.000      oilt     0.00    0.00    0.00    0.00
  lflap   1.000    rflap   1.000      oilp     0.00    0.00    0.00    0.00
  laile   0.000    raile   0.000                    Fuel Tanks
  ruder   0.050    spoil   0.000      #tanks 4
  nosew   0.000    brake   0.000      fuelq  699.40  654.76  186.01   44.64
#wheels 3
wow       1.00    1.00    1.00
gpos      1.00    1.00    1.00
gsteer   -0.00    0.00    0.00
gcompr    0.14    0.40    0.40
--------------------------------------------------------------------------------
```

Some parameters are not printed, and only the modules of some vectors are printed (speed, wind).

Observations:

- this covers only "Flight Dynamic Model" data about the simulated plane. Nothing about "AI" planes, or multiplayers, or embedded systems (radar, navigation, etc).
- F-14b doesn't display the status of the two engines separately (even if we can fail one or the other, and one engine can stop before the other when there remains almost no fuel).
- F-14b fuel reservoirs are different and, more numerous. The quantities are given in gallons! Given: FWD, AFT, one (L or R?) Beam Box, one (L or R?)Sump. Missing: externals, wings.
- Nothing about weapons, passengers, cargo, mass.
- time is not updated in packets (not used anymore in multiplayer packets, IIRC).
- what is warp? I've seen 0 and the above value.
- what is wow? ==> Weight On Wheels, when the plane is landed.
- we seem to get from fgfs 2.0.0 packets with invalid floating point bit patterns (all 0).

dump-multiplayer

Files:

- dump-multiplayer.lisp
- multiplayer.lisp

See `mptraffic.dump` and `mptraffic.log` for an example of output.

Modules

COM.OGAMITA.FGFS.APT

Source: apt.lisp

Parses Airports/apt.dat.gz and retrieve information about airports.

COM.OGAMITA.FGFS.GEOMETRY

Source: geometry.lisp

This package exports geometry functions. Literally, Earth measuring ones, coordinates, distances, etc.

COM.OGAMITA.FGFS.BINARY-PACKET

Source: binary-packet.lisp

This package exports a macro to define structures with typed fields and corresponding binary packets, and serialization and deserialization methods.

COM.OGAMITA.FGFS.UDP

Source: udp.lisp

Provides a simple UDP API.

COM.CYBERTIGGYR.XDR

Source: xdr.lisp, xdr-tests.lisp

This package exports a XDR serialiser/deserializer. (Adapted from cybertiggyr.com).

props

Source: props.lisp

Implement a props client to fgfs.

Unfortunately, fetching all the props is very slow.

radar-map

Source: radar-map.lisp

glut program displaying the multiplayer traffic.

Not implemented yet.

Resources

http://mapserver.flightgear.org/git/gitweb.pl?p=terragear-cs

Random Notes

Matthias Fröhlich added HLA/RTI support last year in these commits:
http://gitorious.org/fg/flightgear/commit/70dd6279a742030271b5b0927501f59bc9aecb98
http://gitorious.org/fg/simgear/commit/44ff23b227dcc1f3efbd10a4df4d8b723165c11c

http://developer.berlios.de/projects/openrti/

Simgear also already has some rti abstraction library that should help to implement hla federates.

Flightgear git already has an alternative multiplayer implementation in place that uses hla. But that is only thought as a proof of concept. The next step is probably to provide a seperate hla federate that runs the ai traffic and feeds that into an rti federation. All I did here started using the above hla implementation. So this one already works for this kind of stuff.