owned\_grid | type ? target\_grid

grid\_name | action/behavior/permission ? grid\_name.(waypoints, move/align : face, x,y,z )

template:

hail\_grid |

-----------

grid\_name | action.mine/salvage/dock/jump ? location

grid\_name | request.dock:repair,refuel,rearm,trade ? grid\_name

grid\_name | behavior.safe/aggressive/patrol

safe is typical for unarmed craft: they will flee any kind of combat and prioritize escape

aggressive calls for a craft to attack and pursue enemies on sight

patrol puts a craft on perimeter patrol duty

- first contact, flee to position of interest, aggressive

? vector \ radius

grid\_name | permission.granted/denied ? target.coordinates(waypoints, move/align : face, x,y,z)

grid\_name | verify ? target.operation

action.dock

- request permission for interaction and/or exchange with target grid

- break and return

- will move to target if permission received

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

handshake:

-grids handle message by saving it and sending a confirmation that it was recieved

-source grid will resend at predetermined interval until confirmation is received

if( one == true)

{

Random r = new Random();

int rand = r.Next(10,70);

float n = rand\*.01;

timer.SetValueFloat("TriggerDelay", n );

return;

}

if( argument == "time")

{

Random r = new Random();

int rand = r.Next(10,70);

timer.SetValueFloat("TriggerDelay", (float)rand\*.1 );

one = true;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

example routing sequence:

From UI, multiple selections for the same task.

-at trade station, buy/sell x units of an item

-choose shipping method (self or contract)

-transaction is deducted in ledger, but not finalized until goods are signed off

-request routing data for jump gate path and station from Sector Authority

-if self, freighter selection is manual or auto

-if auto, select first freighter in fleet with ample space

-freighter is issued a string (handled ship-side) with list of orders

-Sector Authority will issue a list of jumpgates

-the last gate leads to the zone with the target station

( G\_Omni-Zulu > G\_Zulu-Delta > G\_Delta-Gamma )

-

traffic control sequence:

//ship requests dock

dock checks vacancy and queue

if open slot, send docking vectors

if no slots, send primary queue vectors

if full primary queue

-ANTI-COLLISION ROAM MODE

-can micro and move towards queue circle during clockwork trajectory countdowns

on other entering/exiting ships

//-put ship in secondary queue of 50 length (never expecting to have more than 50 grids)

//-when queue opens up, send that waiting ship to queue slot to be processed

parent->broadcast->ship->list(jump@gate > jump@gate > dock@station)

\*ship stores as string in this format, it will parse according to '>' as separator

and handle statements as a switch statement,

i.e. if jump@ then GATE | request.jump ? SHIP

if dock@ then STATION | request.dock ? SHIP

subsequent parent->broadcast statements will compound and add to the list

ship->broadcast->

station\_name | request.dock:refuel,rearm ? ship\_name

station->broadcast-> (if available bay)

ship\_name | permission.granted ? coordinates (template = Waypoints,

move/align : face, x, y, z >

move/align : face, x, y, z >

hold >

(continue received)

move/align : face, x, y, z >

move/align : face, x, y, z >

verify

(station begins connection procedure, and fulfills services according to communications between owner)