

# Ships Components

- 1) MBD NAMES FOR SHIP SIREN
- 2) SHIP DATA FILE  
HAS INFO ABOUT MESH FILE
- 3) SHIP MESH FILE
- 4)

#1 checkout Med.cfg

#2 DOS cd date\mesh

- copy existing mesh to new name of mesh  
ex: copy MULTI.IFF SIREN.IFF

- cd \dsrc\ships

- ex: copy wesp.xmf siren.xmf <sup>8 letters</sup>

- edit siren.xmf

- change comments + "Base" to "siren"

- Save + exit

edit md5.bat (checkout)

- make icon. c:\wst\util\missedit\cfg\siren.ico

- util\missedit edit med.cfg change ship stuff

mesh name or shipfile name = 8 characters    Medname = NO SPACES

## Wing Commander Armageddon Series Overview

### Series: Simulator Training Missions

#### Series Objectives / Description

There will be 5-10 Generic Missions. Mission Names S1-Sx

There will be 5-10 specific ship missions. Mission Names SS1-SSx

There will be 5-10 Multi-Player Scenarios for the players to choose from. Mission Names SM1-SMx.

There will be 5-10 Gauntlet-type missions. Mission Names SG1-SGx.

The objective of the simulator missions is to become intimately familiar with most flight, combat, and game elements seen in the regular missions throughout the "real" game, but with no enemy threat or plot-driven pressure.

The naming convention is as follows:

- The **Generic missions** will teach the player how to operate the ship in stages of complexity. The first will teach enough to get through the first three or four missions without needing to learn anything new, the second will test him on his ability to do basics. The pattern continues from there, presenting progressively complex information and testing the player's skills to that point. And the final stages of training will be optional elements of the cockpit (e.g. power control).
- **Specific Ship missions** will train the player on the advantages and disadvantages of each particular ship as it becomes available.
- **Multi-Player scenarios** will put the player against varying degrees of enemy ships.
- **Gauntlet missions** will put the player against varying degrees of enemy ships. Only Kilrathi Gauntlet modes will be available at first, but around Series E the alien ships will become available.

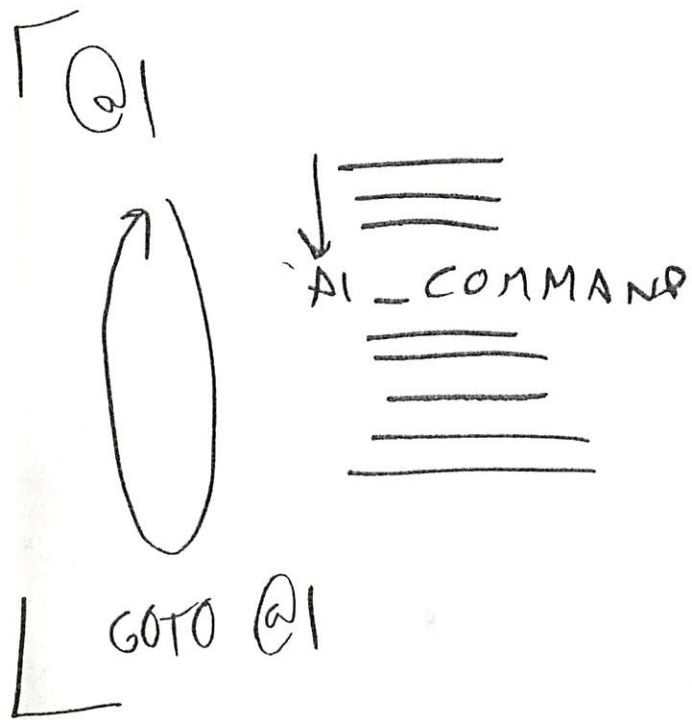
All of the training commands will be delivered by a "trainer", who will provide voice-overs. They will deliver more than adequate amounts of information.

**The actual narration will vary depending on the peripherals the player has installed** (joystick, keyboard, rudder pedals, etc.). See the mission-specific descriptions for details.

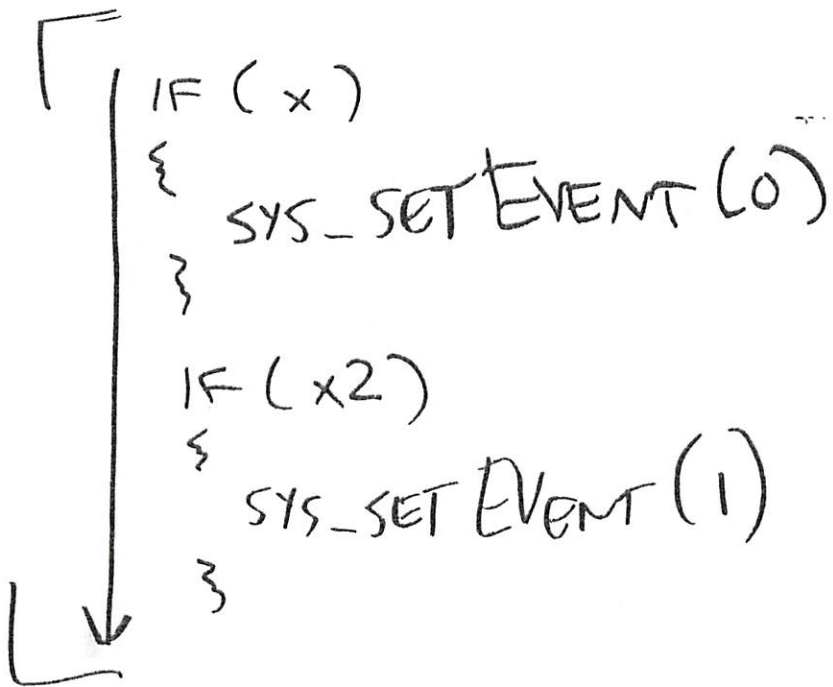
**If the player ever wants to repeat one of these missions, he can.**

We will probably give the player a small trinket, such as a sub-campaign ribbon (Wing I style), for completing all of the simulator missions.

# PROGRAM



# EVENT HANDLER



# EVENTO



# EVENTI



PROGRAM

→ Q1

AI\_WAIT\_SECONDS (10)

→ GOTO Q1

EVENT HANDLER

{ IF (AI\_GET\_DISTANCE (PLAYER) < 100)

{ SET\_EVENT (0)

{ IF (AI\_GET\_DISTANCE (PLAYER) < 500)

{ SET\_EVENT (1)

}

EVENT 0

→

AI\_ATTACK (PLAYER)

L

EVENT 1

{

PLAY\_SFX (comm)

L

## Cain, Billy

---

**From:** Sommers, Andy  
**Sent:** Tuesday, May 27, 1997 10:57 AM  
**To:** Hughes, Jason; Barnes, Cinco  
**Cc:** ML WCP DESIGN  
**Subject:** RE: Thoughts on Object Program Economy...

SYS\_Message has been added for you (through the message plaque system) to print the value of any flag or inform you that you are executing a block of code WITHOUT fataling.

To test the randomness of "random", you should be able to use the following added lines in your program below :

-----  
**From:** Barnes, Cinco  
**Sent:** Saturday, May 24, 1997 10:32 AM  
**To:** Sommers, Andy; Hughes, Jason  
**Cc:** ML WCP DESIGN  
**Subject:** Thoughts on Object Program Economy...

We've been informed that limiting the number of unique object programs is a good thing for frame rate.

I'm trying to take this to its logical extreme.

The idea is to have one program that does a random roll for target importance that **each object using the program** will get a \*somewhat\* different value for. ( Rather, that each object will get its own \*random\* number. )

However, I can't seem to generate the results that I need.

Please help me by taking a look at the following med code:

```
-----  
void M_EnemyFighter ()  
{  
  
// Enemy Fighter startup is standard.  
// However, the ships will all be bound to 0 so that the player can  
// have them "on his ass" the whole time!  
  
    AI_WaitUntilActive();  
  
    SF_SetObjectFlag(OF_vulnerable, 0 );  
    SF_SetObjectFlag(OF_alignment, ALIGN_ALIEN );  
  
    SF_BindToActionSphere(0);  
  
    AI_ClearAttackList();  
  
// To generate the attack list, each object that runs this program  
// should get its own value for target importance.  
// However, "Random" can generate zero, so we add ( 1 ) to avoid  
// crashing.
```

```
int temp1 = SYS_Random (100) + 1;
int temp2 = SYS_Random (100) + 1;
SYS_Message (temp1);
SYS_Message (temp2);
AI_AddToAttackList( Alpha2 , temp1);
AI_AddToAttackList( Player , temp2);
```

```
while ( 1 )
{
    AI_Attack();
}
}
```

The major problem for me is that the random values for target importance appear to roll \*once\* total.

Let's say three active ships have the above program. After they roll, they ALL go attack the same object.

If you're frowning, shrugging, with palms to the sky saying "What the fuck did you think this would do, Cinco?"

I would then like to ask that there be some way of getting individual objects to roll individual random values when hitting a program like this. I don't know if this would defeat the purpose, or if it would even be possible.... but I'd like to know.

Thanks!

- Cinco

## DIFF TYPES of C

- EACH COMPONENT
- TRAPS / PITFALLS
- DEBUGGING TECHNIQUES

TYPES

OPERATORS

EXPRESSIONS

STATEMENTS

CONTROL

FUNCTIONS

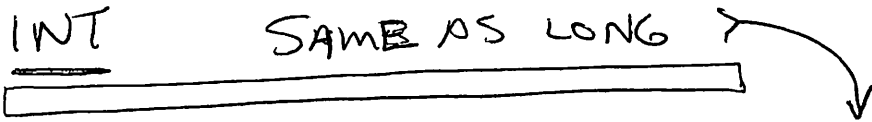
STYLE

TRICKS

DEBUGGING

# TYPES - DATA REPRESENTATION

INT      SAME AS LONG



LONG    Value from -2147483648 to 2,147,483,647  
DON'T USE LONGS

VOID

MOD/WCP USGS ONLY INT, LONG + VOID

void FUNCTION001()

{

  INT Temp;

}

void is a type with no value    NOT ZERO - empty type.

[INT    BOOLEAN = {0, 1}]    MED DOES NOT SUPPORT !!



**OPERATORS** - characters to allow ~~type~~ manipulations on types

**OPERAND** - SOMETHING THAT IS OPERATED ON.  
(TYPE OR VALUE OR RESULT OF SERIES OF OPERATIONS)

3 Types

UNARY works on one operand

BINARY works on 2 operands

RELATIONAL }  
Comparison

- NOT "!" NOT

```
int temp;
temp = !temp;
Reverses State
IF 0 => NON-ZERO (1)
IF 1 => ZERO
```

- "-" negation

switches sign of value  
IF negative => pos  
IF pos => negative

```
int temp = 7;
temp = -temp;
temp == 7
```

- "=" assignment

```
int temp;
temp = (-temp);
temp = 7;
```

can't work by themselves

- "+" addition

```
int temp;
```

```
temp = 8 + 7;
```

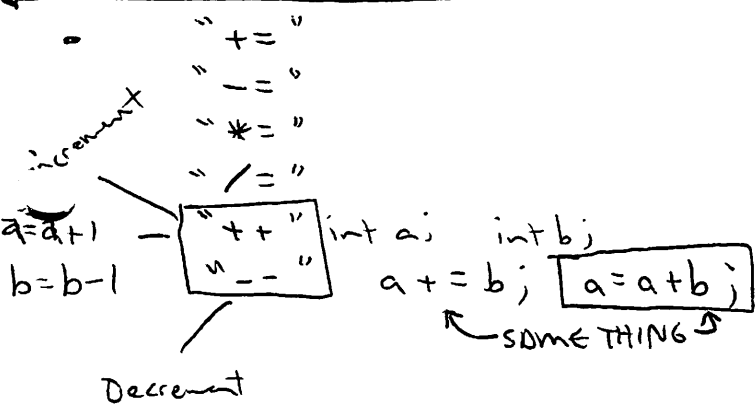
- "-"

- "\*"

- "/"

go from left to right

```
int a;
int b;
int c;
c = a + b;
a - b;
a * b;
a / b; - PVIDE BY ZERO error
```



# BINARY OPERATOR

"&"

&	TRUE	FALSE
TRUE		
FALSE		

# RELATIONAL

[&&] "Logical AND"

BOOLEAN  
OPERATION

0 = FALSE

> -1 < 1 = TRUE

&&	True	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

money

RETURNS 0

```
int a;  
a = 0;  
int b;  
b = 0;
```

```
c = a && b; → c = 0
```

```
if (a && b)
```

RELATIONAL

"Logical OR"

		true	;	false
true		<u>true</u>		<u>true</u>
false		<u>true</u>		false

\$\$

Money or ID gets into club

BINARY RETURN 0 or 1

ID

if (FRONT KILLED || Vaquero killed)

"==" "equality" (1) returns true if both operands are equal  
returns (0) if they are not equal to each other

```
int a;
a = (7 == 7)  a = 1
```

```
if (shuttle docked == 1)
{
}
}

```

[!=] "INEQUALITY" or "not equals"

returns 0 if both operands are equal  
1 if both operands are not equal

```
a = (7 != 8);
```

# RELATIONS

- [ > ] Greater than
- [ >= ] Greater than or equal to
- [ < ] less than
- [ <= ] less than or equal to

return a true (1) or false (0)

```

a = 7 > 8
a = 7 >= 8
a = 7 < 8
a = 7 <= 8;

```

0
0
1
1

# EXPRESSIONS

A collection of operators and (or) operands that can be evaluated

```

(7 > 8)      (!a)      (a == 3)      (a + b)
(a)          (5)

```

**COMPLEX EXPRESSION**: AN EXPRESSION COMPOSED OF MORE THAN ONE EXPRESSION.

```

int a;      a = 7;
int b;
b = (a > 5) && (a <= 10);

```

b == 1

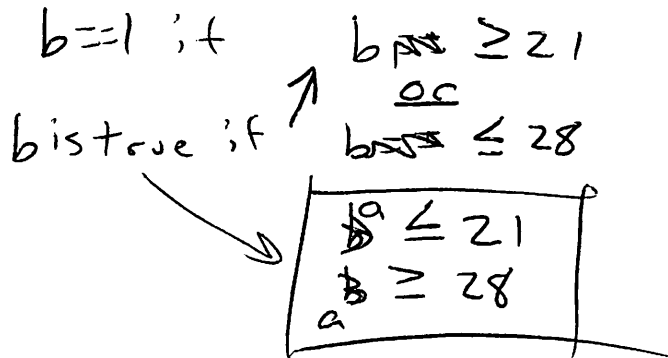
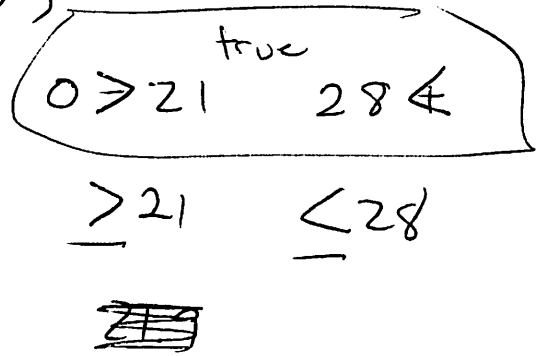
**STATEMENT**: A LINE OF CODE (THAT CAN BE EXECUTED) WITH ONE OR MORE EXPRESSIONS

YOU MUST HAVE A SEMI COLON AT THE END OF ANY STATEMENT LINE.

int a;

b = (a < 22) || (a > 27);

b true  
b false



b is false if a = 22, 23, 24, 25, 26, 27,

int a = 7;  
int b = -3;

a = (b \* 7) / (a - a);

a = 21 / 0;

DIVIDE BY 0 error

NO, ~~it~~ WILL COMPILE ← ~~it~~

What is a? = wont compile

OK why because you're dividing by zero

OK WCP = Divide by zero error in function 001 ✓✓

# CONTROL

How is your program going to execute?  
PROGRAM FLOW

LABEL/GOTO

IF/ELSE

WHILE

LABEL/GOTO

LABEL IS A ~~COLLECTION~~ <sup>STRING</sup> OF CHARACTERS  $\leq 4$   
FOLLOWED BY A COLON [ALPHA NUMERIC WITH  
ALPHA FIRST]

LOOP: ✓    1 LOO: X    1 CATT: X

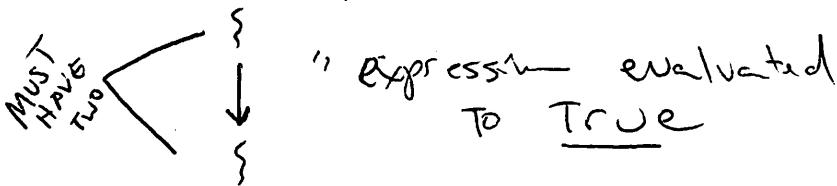
GOTO [LABELNAME];

Block of CODE

{ ← START OF CODE BLOCK  
" " ;  
" " ;  
" " ;  
} ← END of CODE BLOCK

IF/else

IF (expression)



if (expression)

{

// TRUE

}

else

{

// expression evaluated to FALSE

}

```

if (expr 1)
{
    // 1 but dont care about 2
    if (expr 2)
    {
        // Both are true
    }
    else
    {
        // is 1 not 2
    }
}
else
{
    // NOT 1 but dont care about 2
    if (expr 2)
    {
        // 2 but not 1
    }
    else
    {
        // neither are true
    }
}

```

while

loop construct

while (expr1) ← is true

{

  // code is executed

  // as long as expr1 is true

}

BREAK "Get me out of this while loop"



$W(h)$   
3

3

# STYLE

## TABS AT 4

- 1) USE COMMENTS
- 2) AT MOST ONE ~~STATEMENT~~ PER LINE  
2.1 Author name/initials // bjc
- 3) NO COMMENTS ON SAME LINE AS CODE
- 4) PLACE COMMENTS (IF ANY) BEFORE STATEMENT OR CODE BLOCK...
- 5) INDENT COMMENT TO MATCH INDENTATION OF STATEMENT OR CODE BLOCK
- 6) BRACES SHOULD BE ALIGNED w/ IF / ELSE / WHILE / FUNCTION

## \*\* 7) USE THE FOLLOWING NAMING CONVENTION

- G\_\*  $\Rightarrow$  function in MCP
  - g\_\*  $\Rightarrow$  variable (flag) in MCP
  - M\_\*  $\Rightarrow$  USER FUNCTION IN MISSION
  - m\_\*  $\Rightarrow$  User variable in mission
  - all others  $\Rightarrow$  local variables
- } GLOBAL  
} mission global

8)



# TRAPS + Pitfalls

1) "=" vs "=="      make sure you're using the correct operator

1.a) "&" vs "&&"      "      "

1.b) "!" vs "!!"      "      "

2) misleading comments → make sure your comments make sense...

3) number (int) overflow      -2<sup>31</sup> to 2<sup>31</sup>

4) uninitialized variables

5) all variables should have unique names...  
+ Functions

6) Overcomplicated statements (Gödel's theorem)

7) Infinite loops: make sure your conditions are valid for all cases

8) Verify IF / while conditions...

9) Operator Precedence

# DEBUGGING

## TWO TYPES OF BUGS

### SYNTACTICAL

```
if ( a = *b )  
if ;  
temp = 0;
```

### SEMANTICAL

```
int avg ( int a, int b )  
{  
    Return (a+b)/3 ;  
}
```



1) Write Perfect code to start with.

2) STEPS TO FINDING BUGS

A) Identify Bug symptoms ←

B) ISOLATE THE LOCATION

C) STEP THROUGH THE CODE, LINE by LINE.

1. Verify line is correct

a. if line is not a simple statement; break down the line

2. Try printing out values of variables

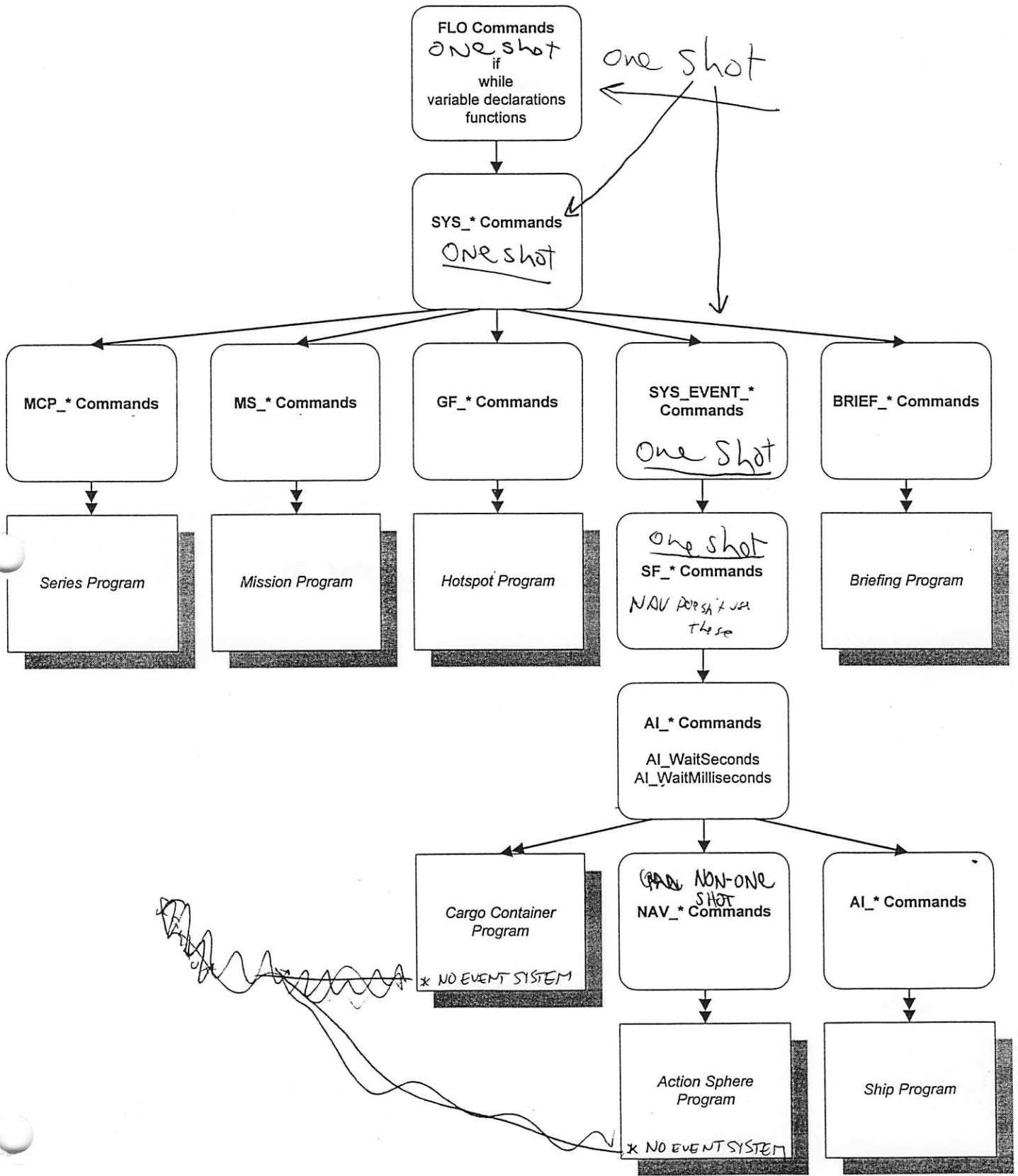
3. Try using ~~SYS FATAL~~ for "impossible" cases  
(BREAKPOINTS, IMPOSSIBLE CASES)

D) Attempt fix + Recompile / Run

E. Get A Second Opinion

# Wing Commander : Prophecy

## Mission System Program Hierarchy

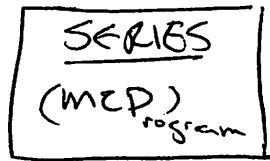


Uses FLO  
 SYS  
 AI-WS  
 AI-WM

# MED CLASS

- SERIES
- MISSION
- Room
- ▣ SECTOR

WING PROPHECY TREE



- ONLY ONE MED @ ONCE
- HOLDS ALL GLOBAL FLAGS + FUNCTIONS



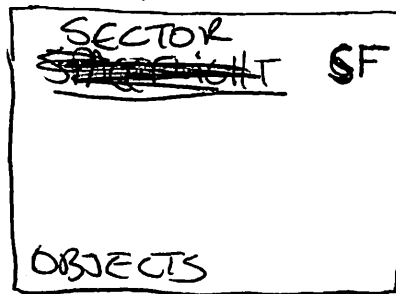
- only one mission prog per mission
- Holds all mission flags + functions

LIST



- more than one room allowed
- Holds hotspots for active room
- Hotspots have programs that use MISSION + GLOBAL FLAGS

GAMEFLOW



- "INFINITE # OBJECTS"
- OBJECTS HAVE programs that use MISSION + global flags.

## SERIES

- SERIES PROGRAM
- LIST OF GLOBAL FLAGS  
NO LIMIT ON NAME SIZE
- LIST OF GLOBAL FUNCTIONS  
~~NOT~~ LIMITED TO 31 CHARACTERS
- LIST OF MISSIONS  
(BUILT ONLY)

### SERIES PROGRAM:

- MCP\* COMMANDS:

```
MCP_RunMission (MISSION_GI);
```

- SYS\_\* COMMANDS

```
int Sys_Clock();
```

Purpose: Returns current ~~in seconds~~ <sup>stack</sup> time from start of game in seconds

USAGE:

```
[var] = SYS_CLOCK();
```

```
int clocktime = SYS_CLOCK();
```

```
if (SYS_CLOCK() = whatever)
```

```
{
```

```
}
```

```
void SYS_FATAL Fatal (int error_code);
```

Purpose: Crashes game with specified error code (for debugging)

usage:

```
SYS_fatal(100);
```

```
void SYS_PlaySFX(int wavfile);
```

Purpose: Play sound effect denoted by wavfile  
USAGE:

```
SYS_PlaySFX(37);
```

```
void SYS_PlayMusic
```

 same as above with music

```
int SYS_Random (int max);
```

Purpose: Returns Random number from 0 to (Max-1)  
USAGE:

```
int n = SYS_Random (37);
```

returns  $n \Rightarrow 0 \dots 36$

GUI Program has a timer associated with it

```
void SYS_TimerSet (int sec);
```

Purpose: Set "Program" timer (timer counts down)

```
SYS_TimerSet (3);
```

 "Timer is set to 3 seconds

```
int TimerGet ();
```

Purpose: Reads "Program" timer (timer counts down)

```
SYS_TimerSet (3);
```

```
int timer_val = SYS_TimerGet ();
```

```
if (SYS_TimerGet () < 10)
```



```
void M_FighterMain ()
```

```
{
```

```
  SYS_SetTimer(60);  
  while (SYS_TIMER Get (0))
```

```
{
```

```
  AI wait milliseconds(250);
```

```
}
```

```
  SF_Activate Self();
```

```
int SYS_TimerZero ();
```

Purpose ; Returns True if Timer expired or = zero

USAGE :

(1) - 10/11/77

(2) - 10/12/77

(3) - 10/13/77

(4) - 10/14/77

ROY G BIV

(5) - 10/15/77

(6) - 10/16/77

(7) - 10/17/77

10/18/77

# Med Class Day 2

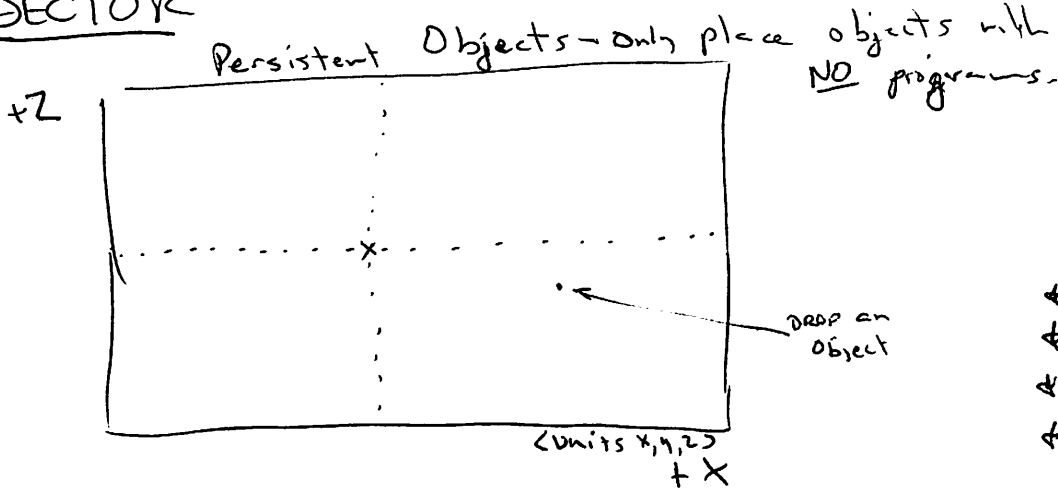
SERIES

MISSION

ROOM

SECTOR

## SECTOR



- \*  SHIP
- \*  AST Belt Start
- \*  AST BELT WIDTH
- \*  CARBO
- DEAD OBJECT
- DEBRIS FIELD
- EJECTED PILOT
- HEAVENLY BODY
- JUMP POINT
- \*  LIGHT SOURCE
- MINEFIELD
- \*  NAUPDINT
- TURRET

ALL OBJECTS START AT +Z

## A LIGHTSOURCE (3 per ACTION SPHERE)

ObjType DO NOT MODIFY {OT-lightsource}

Intensity: { r, g, b } (0-1)

Orientation { y, p, r } (0-360)

Radius { clicks }

Rotation Rate { y/second, p/sec, r/sec }

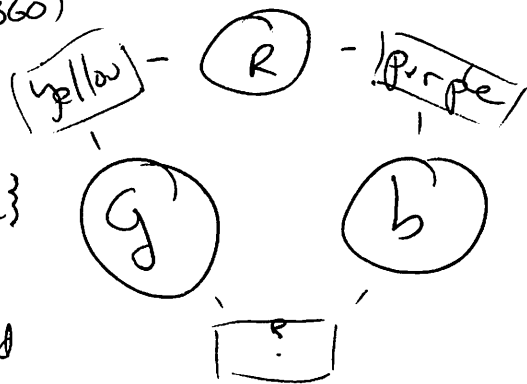
degrees/second

Start NavPoint { ~~n~~ n } 0...9

TypeLight {LI-Ambient, LI-parallel, LI-point, LI-spt}

yaw 180 = face backwards

NOT supported yet

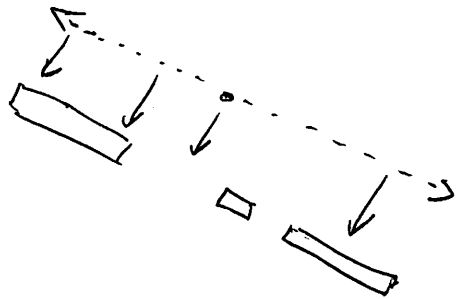


LT\_ambient - intensity  
Start New Point

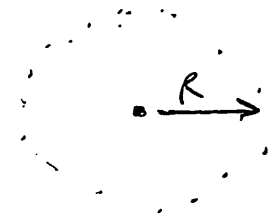
- LIGHTS ALL SIDES OF OBJECT AND IS OVERALL LIGHT VALUE IN ACTION SPHERE
- 0.2 ... 0.4 are good values (Proposed)

Average RGB  $R+G+B/3 = 0.2 \text{ to } 0.4$

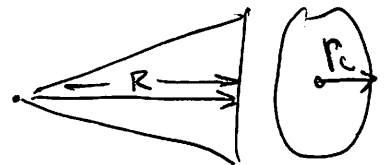
LT\_parallel - Intensity  
Start New Point  
Orientation  
Rotation Rate



LT\_point - Intensity  
Start New Point  
~~Orientation~~  
Radius



LT\_Spot - Intensity  
Start New  
Orientation  
Radius  
Rotation Rate  
Start New Point



# Med class

## Asteroid Belt Start + Asteroid Belt Control Points

### PROPERTIES

- ObjType - DO NOT MODIFY
- Briefing Data
- Field Info
  - number (Field)  $\Rightarrow$  which field does this belong to (0...3)
  - number Items  $\Rightarrow$  number of asteroids in field (1...2000)
  - random seed  $\Rightarrow$  seed to initialize random number generator
  - gps  $\Rightarrow$  max yaw/sec for an asteroid
  - pps  $\Rightarrow$  " pitch/sec " " "
  - rps " roll/sec " " "
  - radius  $\Rightarrow$  radius of crazy straw "cylinder" size/goodness

NOTE - ONE AST BELT START PER AST FIELD

- AT LEAST ONE CONTROL POINT NEEDED
- UP TO NINE ADDITIONAL CONTROL POINTS

### Ast Belt Control Points

#### Properties

- ObjType - DO NOT MODIFY
- CONTROL POINT
  - Field number  $\Rightarrow$  which field does this control point belong to?
  - cp-index  $\Rightarrow$  " Control point am I in the field.

DEBRIS FIELD  
MINEFIELD

# NAV POINTS (SAME AS ACTIONS SPHERE)

"CONTAINER FOR OBJECTS"  
 "0 is persistent actionsphere"

## Properties

~~Obj\_Type~~ Do\_Not\_Modify

Briefing Data

DescNav {String}

FLO-main "FILL THIS IN"

### Nav Point

nav\_index {1...9}

hiddenflag {true or false}

radius

propoff ~~range~~

- nav point number in Nav Point List
- whether or not Nav Point shows up in briefing
- in clicks NavPointRadius
- Distance from Center along entry vector..

## NUMBER

~~ALL~~ all NON-HIDDEN NAV POINTS FIRST

void NAV\_ActivateSelf();

purpose: To have an actionsphere 'turn on' and start running the programs for ~~steps~~ objects attached to this Nav.

void NAV\_DeActivateSelf(); - turn off and Pause programs

~~int~~ NAV\_WithinSphere(~~ObjId o~~) (\_ObjId o);

purpose: returns true if ~~objId o~~ o is within the action sphere defined by radius

~~void~~ NAV\_SetDescription(~~SP\_String s~~);

purpose: changes Nav Description in NavMap and NAV IDU to 's'

void NAV\_HideNav();  
 purpose: Hides Nav in ~~NavMap~~ NavMap

void NAV\_ShowNav;  
 purpose Shows nav in ~~NavMap~~ NavMap

void NAV\_SetPlayerNav(int new\_nav);

purpose: sets the player's next nav to new\_nav

# SAMPLE NAV PROGRAM

```
void m_nav1(1)
```

```
{  
  int loop = 1;
```

```
  while (loop)
```

```
  {  
    if (Nav_With_Sphere(Plaza))
```

```
    {  
      NAV_Activate_Self();
```

```
    }
```

```
    else
```

```
    {
```

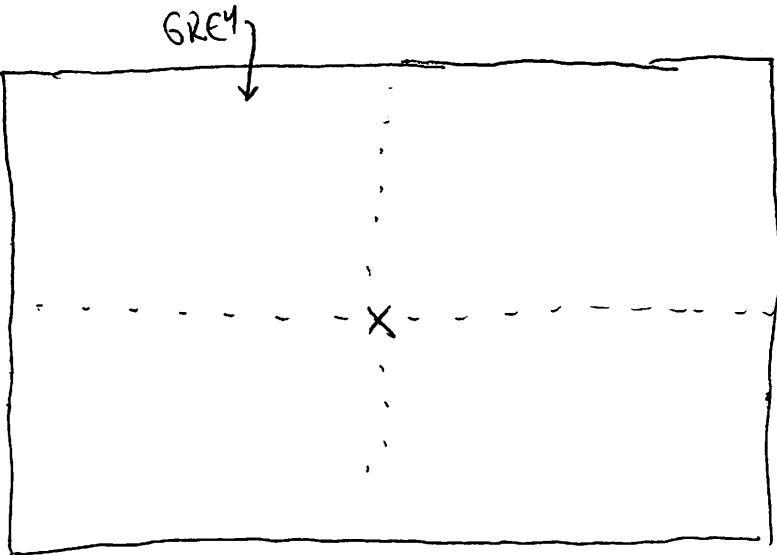
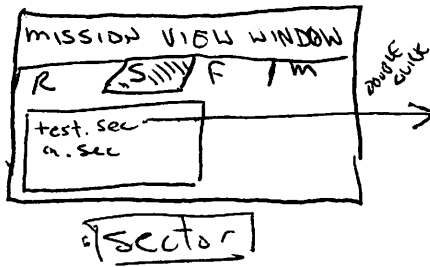
```
      NAV_Deactivate_Self();
```

```
    }
```

```
    AI_WAIT_SECONDS(1);
```

```
  }
```

# MISSION - Control GAMEFLOW + SPACEFLIGHT



Sector window allows you to place Mission specific objects into a space flight sector.

## MISC

### Properties

- ObjType - DO NOT MODIFY
- FLO\_MAIN "FILL THIS IN"
- start\_seed (RANDOMSEED)
- sample\_list (SEX\_LIST)

## FUNCTION

commands (mission specific [MS\_\*])

void MS\_RunGameflow (int <sup>start room number</sup> start room number)

Starts the user in the Gameflow room specified by start room number

Start room number starts at 0 + corresponds to the number the room was added to the room list

NOTE: THIS MAY NOT CORRESPOND TO THE ORDER THE ROOMS ARE LISTED.

usage: MS\_RunGameflow(0); //starts user in 1<sup>st</sup> room



void MS\_RunSpaceflight (int n); (= 'is ignored

- starts the user in the FIRST (and ONLY) sector listed

usage MS\_RunSpaceflight (0);

void MS\_Run <sup>ICIS</sup> ~~ICIS~~ ();  
Runs the ICIS for this mission

usage MS\_Run ICIS ();

void MS\_Run Briefing ();

Runs the Briefing for the mission

void MS\_Exit ();

exits mission + goes back to series

```
void M_MissionMain ()
```

```
{  
    // see if Vagabond available  
    if (!g - VagabondTalkedTo)
```

```
{  
    MS_RunGameflow (1);
```

```
};  
else
```

```
{  
    MS_Run_Gameflow (0);
```

// see if player viewed  
Briefing

```
{  
    if (g - Briefing Viewed)
```

```
{  
        MS_RunICIS ();
```

```
};  
else MS_RunBriefing ();
```

```
};  
// Go into Flight  
MS_RunSpaceFlight ();
```

```
}
```

# Cargo Containers

OBJECTS THAT CAN BE IDENTIFIED BY THE PLAYER'S TARGETING SYSTEM

## PROPERTIES:

- Obj\_Type DO NOT modify
  - Appear Cargo {Shape}
  - BriefingData
  - DescCargo
    - identifiable
    - ident.fied
    - contents
- |            |
|------------|
| true/false |
| true/false |
| SE_STRING  |
- can be identified Y/N  
false = UNKNOWN  
what shows up in target VDU
- \* FLO\_Death
    - death program that's run when the object dies
    - only one-shot commands are allowed.
    - name of a function expected
    - if "NULL"
  - FLO\_Main
    - Cargo container's main program
    - (See sheet for types of MISSION Program commands)
  - orientation
    - 90° in Degrees
  - \* Spawn\_number
    - { 1 or above }
  - Start Nav Point
    - { 0 to # of Navs - 1 }

\* NOT HOOKED UP yet

# MED

SF\_\* commands: "Generic" Spotlight commands that can be used with almost any type of spotlight object...

void SF\_SetWorldPosition (long x, long y, long z);  
allows you to move an object (teleport) to the specified coordinates

ex: // move obj to <-10,000, 2000, 0>  
SF\_SetWorldPosition (-10,000, 2000, 0);

void SF\_ActivateSelf (int display\_jump);  
: turns the object on - makes the object visible; targetable; collidable; etc

parameter display\_jump if & do not display jump sequence; if non & = display

void ~~Activate~~ SF\_ActivateObject (-ObjId obj, int display\_jump);  
"Turns object Obj on."

ex: SF\_ActivateObject (MarineShuttle, 1);

void SF\_DeactivateObject (-ObjId obj, int display\_jump);

~~is~~ opposite of

```
void SF_DeactivateSelf (s int display_jump);  
turns object 'off'
```

ex:

```
AI_GotoObs Range (JumpPoint, 1000);  
SF_DeactivateSelf(1);
```

```
void SF_DestroySelf();
```

deactivate + Display explosion + Run Death Program

```
void SF_DestroyObject (-ObjId obj);
```

Deactivate object ↗ + Display explosion + Run Death Program

```
int SF_GetDistance (-ObjId obj);
```

returns distance to obj, in meters

ex:

```
int dist = SFSF_GetDistance (Player)
```

```
int SF_Exit();
```

exits spaceflight

ex:

```
if (SF_GetDistance (Player) < 100)
```

```
{
```

```
    SF_Exit();
```

```
}
```



```
void SF_SetObjectFlag (EObjectFlags F, int v);
```

```
EObjectFlags  
{  
    OF_alignment  
    OF_cloaked  
    OF_leeches  
    OF_vulnerable  
}
```

ALIGN\_ALIEN

ALIGN\_NEUTRAL

ALIGN\_CONFED

```
ex SF_SetObjectFlag (OF_alignment, ALIGN_ALIEN)  
SF_ " (OF_cloaked, true)  
SF_ " (OF_vulnerable, 0) : invulnerable
```

int SF\_GetObjectFlags (EObjectFlags f);

ex:

int align = SF\_GetObjectFlags (OF\_align);  
OF\_vulnerable);

void SF\_BindToActionSphere (int new\_sphere);  
switches the action sphere this object belongs to new\_sphere.

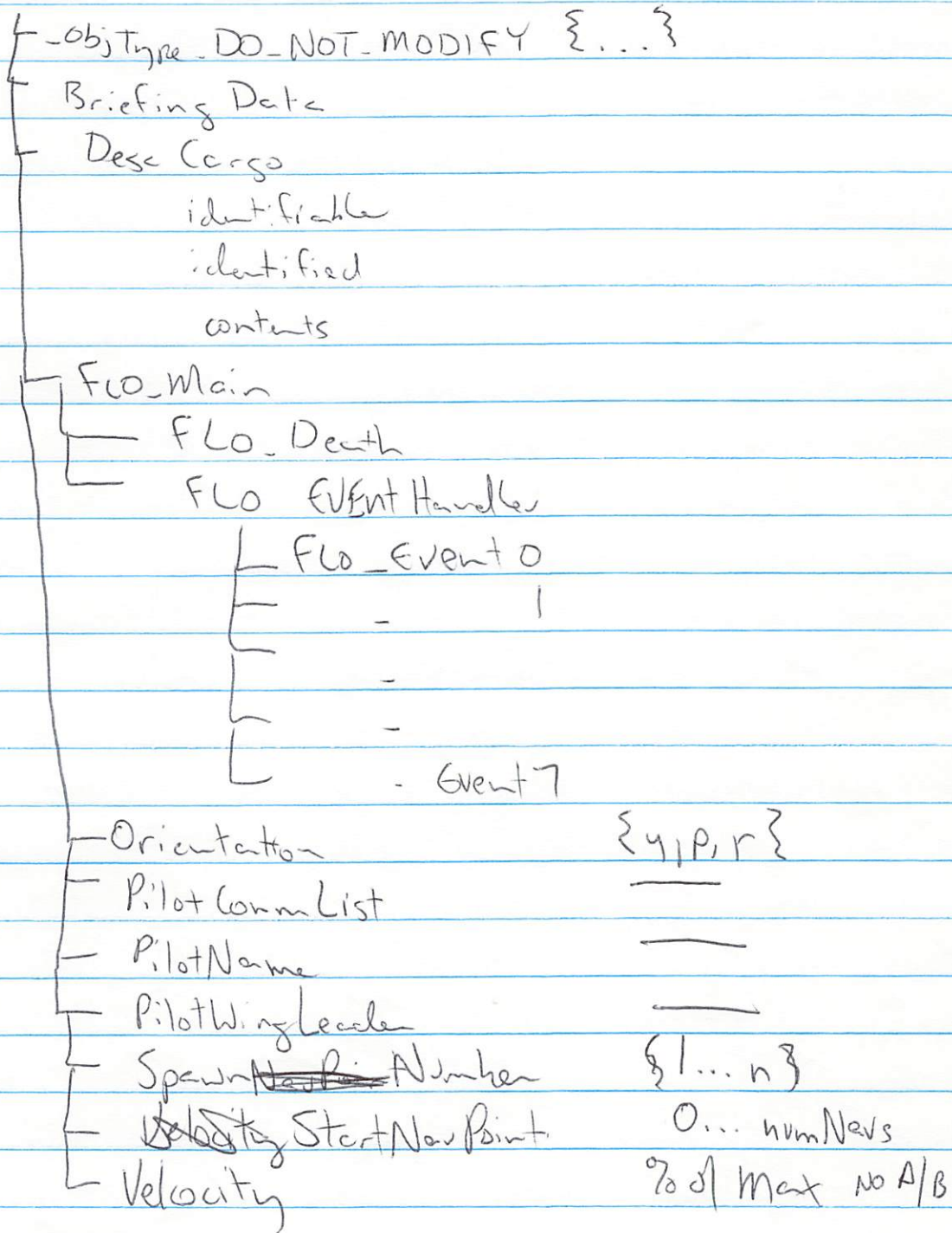
void SF\_AutopilotEnable (); NOT NEEDED

void SF\_PlayerSwitchToCam (int cam, \_ObjId stunning\_target);  
SF\_PlayerSTC (Fr)

void SF\_SaveObjectState ();  
Flag object to be saved from mission to mission

void SF\_SetShipDamage (int main, int child\_min, int child\_max);  
affect ship w/ main % damage  
apply ~~children~~ child\_min % to child\_max %  
damage to each ~~to~~ child object

# SHIP





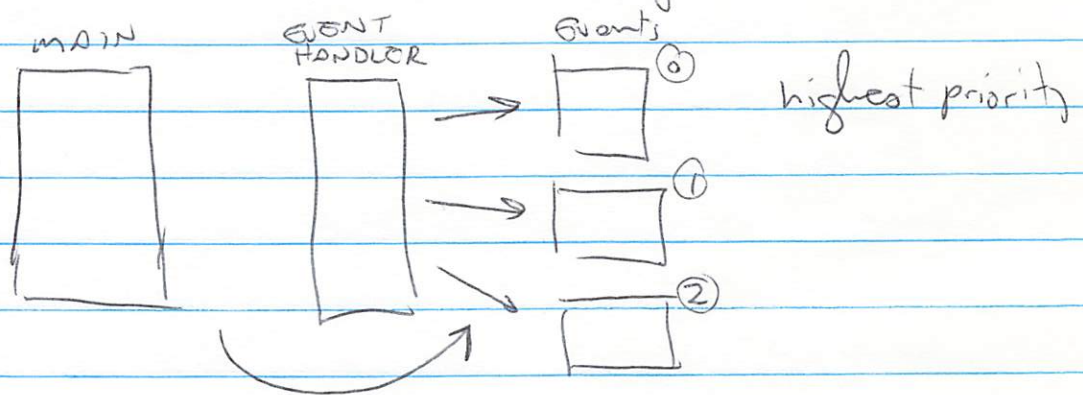
# EVENT SYSTEM

The event system allows temporary halting of the currently running command to be interrupted by a new command based on some 'event'

## EVENT HANDLER (one shot)

: Program that gets run every "object update" (if enabled)

: EH is responsible for setting events as 'true' or 'false'



NOT ONE SHOT

Events: Programs that can interrupt Main or EVENTS of Lower Priority

Priority: 0 is highest priority. An ordering scheme for the events.

## Event Commands

### SYS\_EVENT\_\*

void SYS\_EVENT\_EnableHandler()

turn on the EH; allow EH processing

void SYS\_EVENT\_DisableHandler()

turns off the EH;

void SYS\_EVENT\_Enable(int e);

turn on Event 'e'

void SYS\_EVENT\_Disable(int e)

turn off Event 'e'

void SYS\_EVENT\_EnableAll(); turns on all events...

DisableAll(); turns off all events...

void SYS\_EVENT\_SET(int e)

only used in EH

```

void M_shipMain()
{
    SF_ActivateSelf(1);
    while (1)
    {
        AI_Waitseconds(1)
    }
}

```

```

void M_ship EH()
{

```

```

    // check for Xena
    if g_forced to wait
    {
        SYS_EVENT_SET(0)
    }

```

```

    // send comm
    if Throttle K'ld
    {
        SYS_EVENT_SET(1)
    }

```

```

    // Play sound
    if (SF_GetDistance(Plane) < ...)
    {
        SYS_EVENT_IS SET (2)
    }

```

```

void M_EshipSound()
{
    SYS_PLAY_SFX(37);
}

```

```

void M_EshipComm()
{

```

```

    if (!m_commSent)

```

```

        "Play comm"(32)
        SYS_EVENT_DISABLE(x);

```

```

void M_FshipSuicide()
{

```

```

    SF_DestroySelf();
}

```



□ What Does the ship do if not interrupted  
[main program]

□ What outside circumstances can influence  
the ship?  
[Event Handler]

□ What do the ships do for each circumstance?  
[Events]

□ What is the Priority of these events

M\_Shipkiller\_EMerinesLanded

When player is near, play sound

IF (g\_ThraxhathKilled) is set,  
~~commit suicide~~ send comm

IF (g\_Introduced to Janna)  
commit suicide

# FUNCTIONS:

## SYSTEM FUNCTIONS:

Pre-defined

Stored "OUTSIDE" PROGRAM

## USER FUNCTIONS:

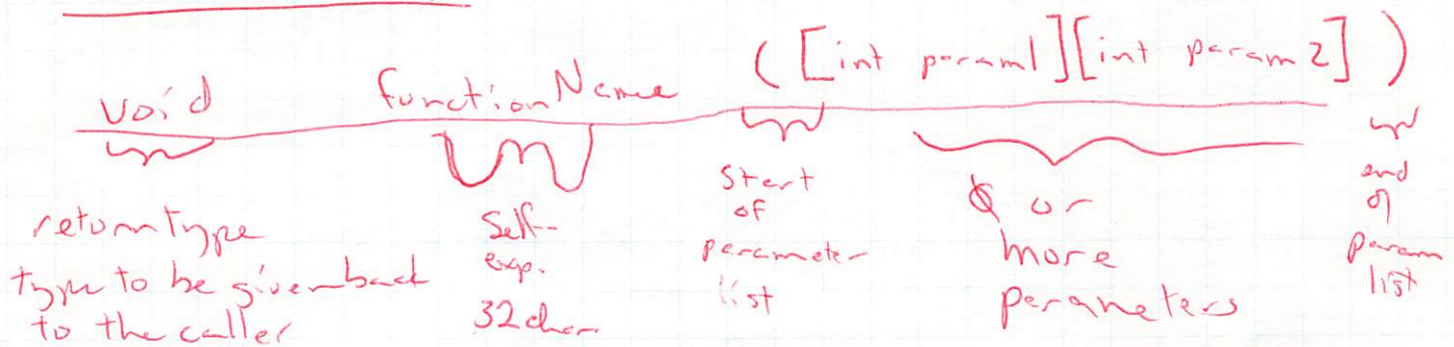
WITHIN source code (me)

Ex:

void Print (-Obid 0);

---

## FUNCTION PARTS:



## WHEN CREATING USER FUNCTIONS:

- 1) ASK yourself does this function already exist
- 2) So-it doesn't:
  - a) Does it return a ~~value~~ anything  
IF so, int  
if not, void
  - b) What's the function name?
  - c) Does it need parameters?

Function header

```
void G_getObj (_ObjId obj, int radius)
```

```
{  
  while (1)  
  {  
    A1_GotoObj, RelFrame (obj, r, 0, r)  
    " " " " " " " "  
    " " " " " " " "  
    " " " " " " " "  
    " " " " " " " "  
  }  
}
```

---

```
int total (int n)  
{  
  int result = 0;  
  result = result + 1
```

```
}
```



Function that returns the smaller of two values

MIN " " " LARGER " " "  
OVERSLO " " " " " BOTH VALUES

```
int Min(int valone, int valtwo)
{
int num = valtwo
  if valone > valtwo
num num = valone
  Return
```

---

```
int min(int valone, int valtwo)
  int num = valtwo
  if valone < valtwo
  ?
  num = valone
  Return num;
```

---

```
int Max(int valone, int valtwo)
  int num = valtwo
  if valone > valtwo
  ?
  num = valone
  Return num;
```

```

int Ave (int valone, int valtwo)
{
    int num=0;
    num = valone + valtwo
    num = num / 2
    return num;
}

```

8 parameters MAXIMUM.....

8 local variables MAXIMUM.....

---

```

int Min3 (int a, int b, int c)
{
int
    int lowest = 0;
    lowest = (min2(a, b));
    lowest = (min2(lowest, c));
    return lowest;
}

```

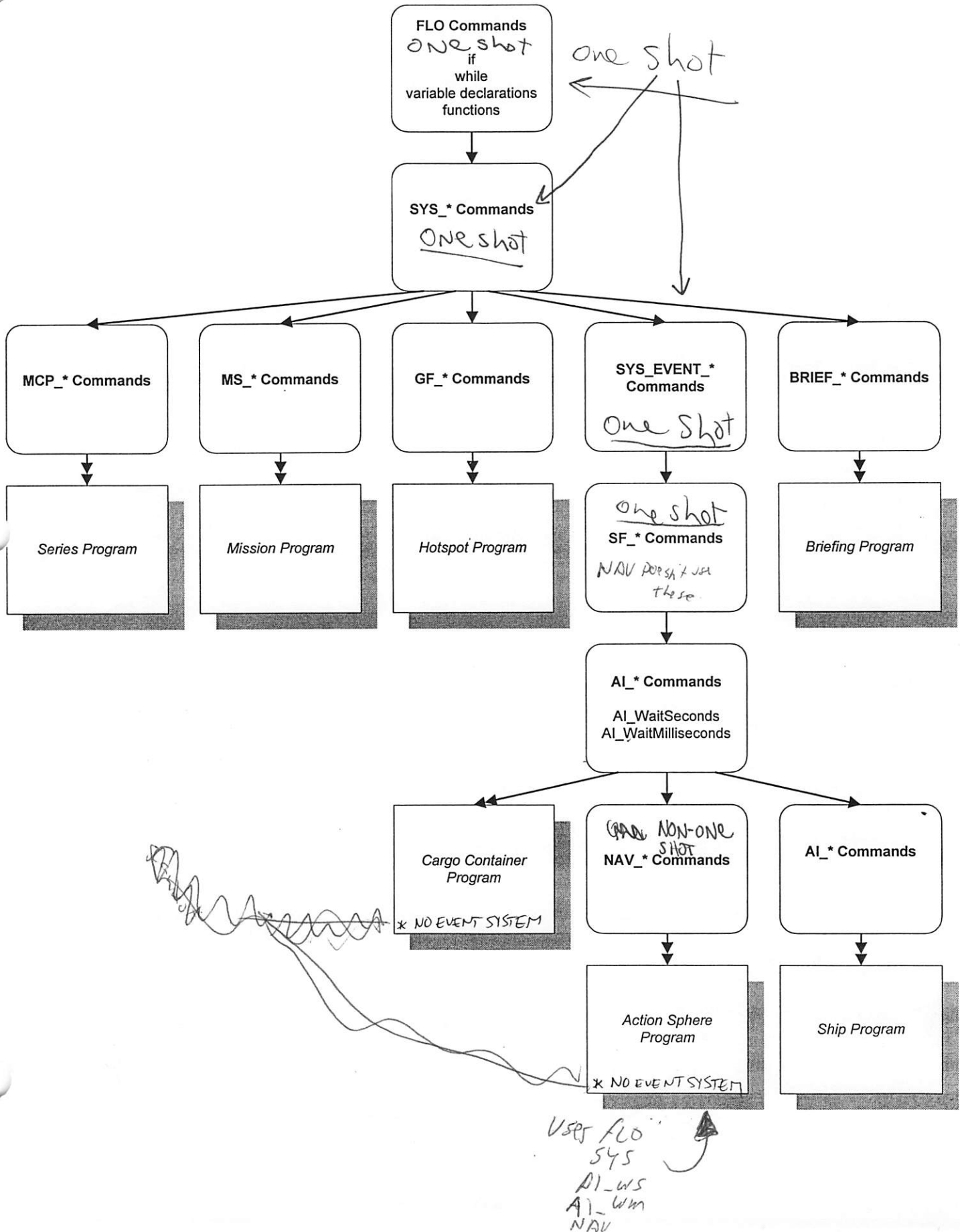


```
int Max3 (int a, int b, int c)
{
    int highest = 0;
    highest = (max2(a, b));
    highest = (max2(highest, c));
    return highest
}
```

```
int Aves (int a, int b, int c)
{
    Return (a+b+c)/3;
}
```

# Wing Commander : Prophecy

## Mission System Program Hierarchy



## Cain, Billy

---

**From:** Sommers, Andy  
**Sent:** Tuesday, June 10, 1997 8:30 AM  
**To:** ML WCP PROGRAMMERS  
**Cc:** ML WCP DESIGN  
**Subject:** Game Update 1.00.08A

### DESIGNERS :

\* cockpit=[filename] is back in use - used to load a specific cockpit

\* New commands :

```
void MS_ShowTakeoffSequence ();
```

purpose : to show the takeoff sequence  
usage : only in mission programs

```
int SF_GetVelocity ();
```

purpose : gets the velocity of the object executing the instruction  
usage : any programs that use SF\_\* commands

```
int velocity = SF_GetVelocity ();
```

```
int SF_GetVelocityObject (_ObjId o);
```

purpose : gets the velocity of the object specified as a parameter  
usage : any programs that use SF\_\* commands

```
int stunning_velocity = SF_GetVelocityObject (Stunning);
```

\* Collision now checks the object's collidable flag set with SF\_SetObjectFlag in your object's program. Default is collidable.

```
// turn off collision for this object  
SF_SetObjectFlag (OF_collidable, false);
```

\* Hopefully the crazy crashes when loading missions should be fixed. There were incorrect assumptions made by the cockpit system about the capabilities of certain ships. I addressed the following issues :

- Missile select indicator (CockpitMissileSelect) crashed if the ship had no missiles.
- Gun power indicator (CockpitGunPower) went loony if the ship had no guns.
- Gun select indicator (CockpitGunSelect) left off the parking garage if the ship had no guns.

Sigh... <Shrug>... [Groan]... "Eye-roll"... {Tsk. Tsk.}... `Lament'... (Flail)... hehehe

\* You can turn off player collisions with CTRL-C. A message will appear informing you of the current status of the flag.

\* You can turn off player invulnerability with CTRL-I. You will still be collided with, but you cannot die. A message will appear informing you of the current status of the flag.

\* OF\_collidable and OF\_vulnerable are now hooked up through SF\_SetObjectFlag.

\* The movie playing in the game has been disabled (pending the ability to stop the sound once it's started)

\* I put asteroids back in...

\* The once-disabled plaque system is back in.

# - GMP CLASS -

Write good mission programs for an object

## Semantics of Programming - NOT SYNTAX

How to script mission

For each mission

- DETERMINE # OF ACTIONSPHERES
- DETERMINE # OF OBJECTS

FOR EACH OBJECT IN MISSION (INCLUDING ACTIONSPHERE)

### STAGES

- BIRTH
- ↓
- LIFE
- ↓
- DEATH

{ WHAT HAPPENS? ;  
{ How DO I GET TO THE NEXT STAGE }

- B {
  - 1) IS THIS A MISSION CRITICAL OBJECT?  
vs. mindless fighter...
  - 2) can this object be bound to an actionsphere?
  - 3) DOES THIS OBJECT NEED TO BE BOUND TO ACTIONSPHERE "O"?
  - 4) WHAT DETERMINES HOW THIS SHIP BECOMES ACTIVE?
- L {
  - 5) DOES THE BIRTH OF THIS OBJECT AFFECT ANYTHING?
  - 6) WHAT DOES THIS OBJECT DO WHILE ACTIVE
- D {
  - 7) WHAT DETERMINES HOW THIS OBJECT DIES
  - 8) DOES THE DEATH <sup>OF</sup> THIS OBJECT AFFECT ANYTHING...? Becomes inactive

" ALL OBJECTS THAT HAVE PROGRAMS MUST BE ACTIVATED "

```
{  
  ACTIVATE_SELF ();  
  while (1)  
  {  
    {  
    }  
  }  
}
```

```
void AI_WaitUntilActive ();  
void SF_ActivateObject (_Object o);  
void SF_DeactivateObject (_Object o);
```

FIGHTER

```
{  
  AI_WAIT UNTIL ACTIVE ();  
  AI_ADD TO ATTACK LIST (XXX);  
  AI_ATTACK ();  
}
```

NDU

```
{  
  IF (...)  
  {  
    SF_ACTIVATE OBJECT (FIGHTER);  
  }  
}
```



#2 can be bound to Intersphere

#4) ship becomes active

a) can ship activate itself

~~B~~ i) immediate activation

ii) conditional activation

b) can any other object activate "me"

#5 Birth effects

a) do I need to activate OR DESTROY OTHER OBJECTS

b) send sounds

c) set flags

d)...

#7. Only to get rid of object

SF\_DeactivateSelf();

SF\_DeactivateObject(-obj, id 0);

USING DEACTIVATE WILL NOT RUN DEATH PROGRAM

DESTROY WILL RUN DEATH PROGRAM  
(OR ~~DESTROYING~~ through collision)

# FRAME RATE

## "EXPENSIVE COMMANDS"

### Exp. CALCULATIONS

- 1) ACTIVATION
- 2) DEACTIVATION
- 3) LIGHTING
  - i) one ambient light/sector
- 4) COMMANDS THAT INVOLVE OBJECTS AS PARAMETERS  
ex. `SF_GetDistance(name);`

### THINGS THAT HELP FRAME RATE

- 1) Lighting
  - 2) one ambient light/sector
  - b) at most - 2 directional lights
  - c) POINT LIGHTS - ONE (PER MISSION?)

### 2) FEWER OBJECTS

- 3) OBJECTS PERFORMING THE MINIMAL AMT OF "EXPENSIVE CALC"

### TID BITS OF INFO:

- 1) WORLD IS BEING UPDATED @ ~~60~~ X  
a second + then the world is being rendered as fast as possible

- 2) AVG # of polys / ship 130-180  
high detail Fighter  
Cap ships ~ 800-1900

- 4) OBJECTS FINISHING THEIR INSTRUCTIONS AS SOON AS POSSIBLE - TO MOVE ON TO THE NEXT OBJECT.

~~THE MORE ARE~~  
SHORTEN UPDATE LOOPS.

- 5) OBJECTS FINISHING THEIR PROGRAMS AS SOON AS POSSIBLE