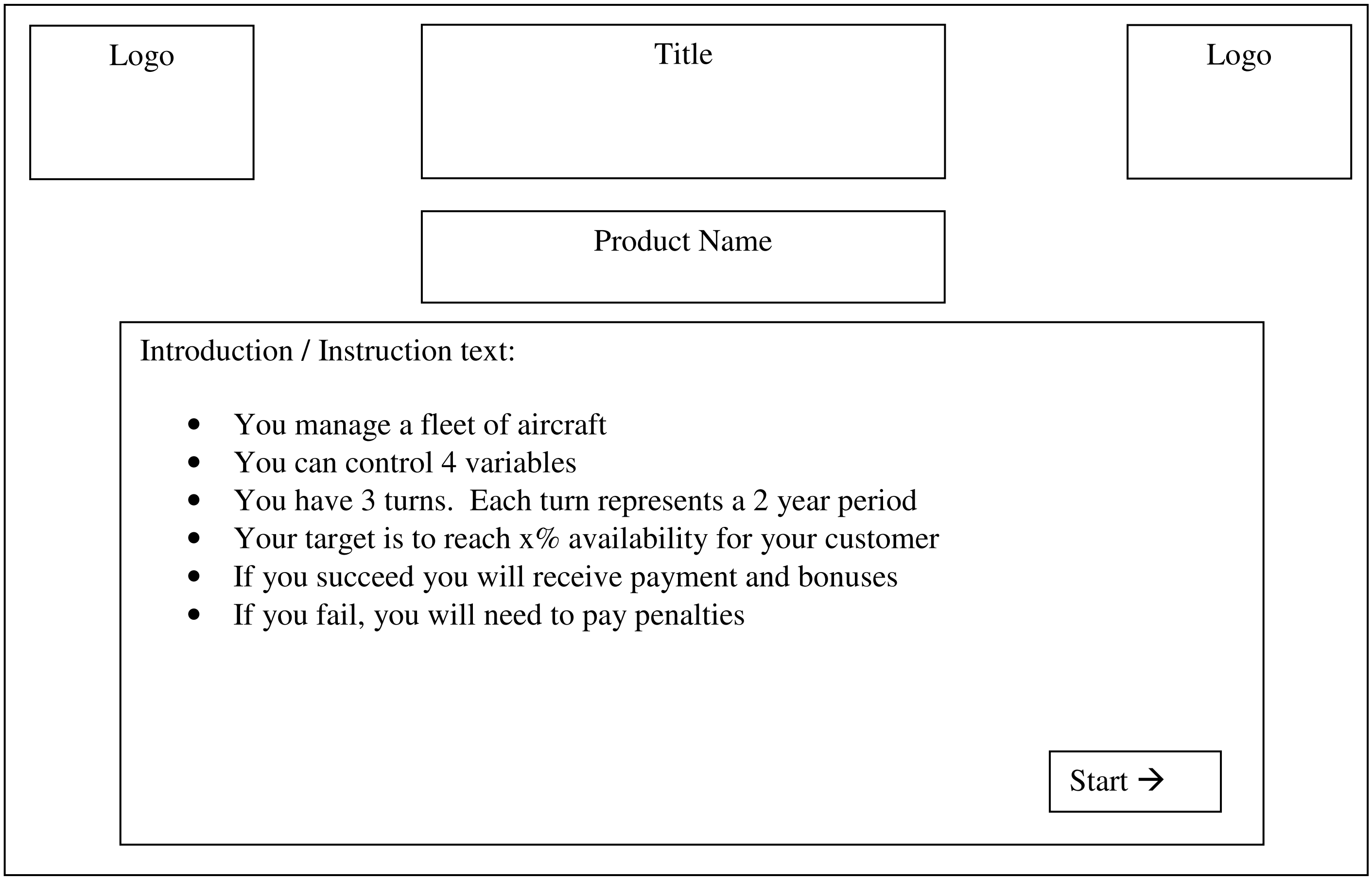
1) Intro Screen



Logo

Title

Logo

Product Name

Introduction / Instruction text:

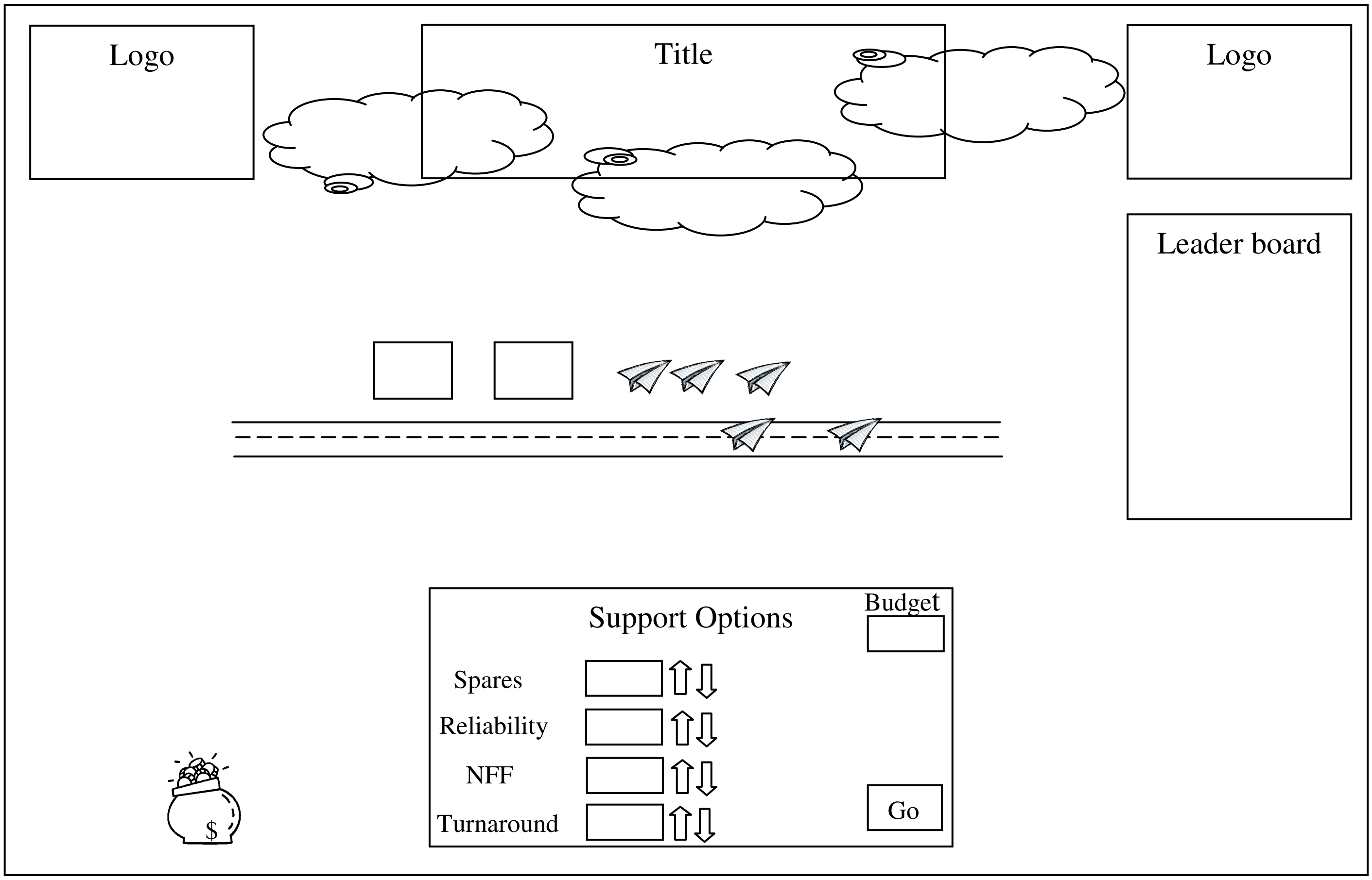
- You manage a fleet of aircraft
- You can control 4 variables
- You have 3 turns. Each turn represents a 2 year period
- Your target is to reach x% availability for your customer
- If you succeed you will receive payment and bonuses
- If you fail, you will need to pay penalties

Start →

General
=====

- All text labels throughout app read in from XML [CDATA] file
- All text throughout app read in from XML [CDATA] file

2) Round 1



Onload
======

- Read starting parameters:
  - S = spares
  - N = NFF
  - R = Reliability
  - T = Turnaround
  - B = Budget
  - SI = Spares Increment
  - NI = NFF Increment
  - RI = Reliability Increment
  - TI = Turnaround Increment
  - SL = Spares Lower Range
  - SU = Spares Upper Range
  - NL = NFF Lower Range
  - NU = NFF Upper Range
  - RL = Reliability Lower Range
  - RU = Reliability Upper Range
  - TL = Turnaround Lower Range
  - TU = Turnaround Upper Range
- Load leader board from XML file
- Load lookup tables (see last page below)
- Increment GameID (saved in leader board file)

Clicking up arrow
============

OnClick event. Get the existing value and subtract the initial value, then lookup that value in the lookup table step column. Get the next "up" step figure and subtract the difference between its cost and the previous cost from the remaining budget. Add that step figure to the initial value and update the box with the new value.

Clicking down arrow
===============

OnClick event. Get the existing value and subtract the initial value, then lookup that value in the lookup table step column. Get the next "down" step figure and add the difference between its cost and the previous cost to the remaining budget. Add that step figure to the initial value and update the box with the new value.

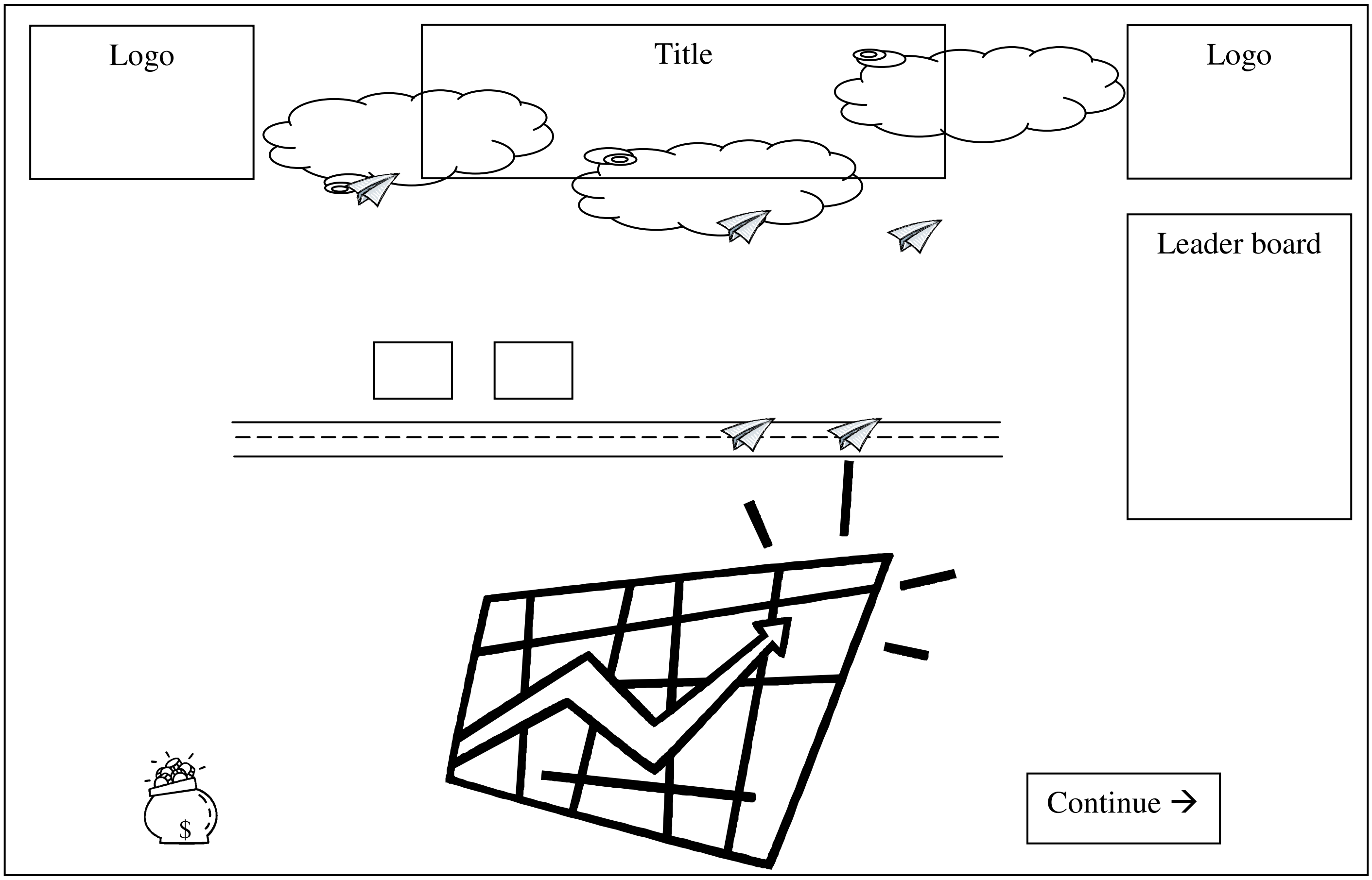
Note: values in the lookup table will need to be positive and negative.

Pass to black box
===========

G=1234 (GameID – auto increment)

I=1 (Iteration)	R=0 (Reliability)
S=10 (Spares)	T=3 (Turnaround)
N=1 (NFF)	B=12.52 (budget)

2) Round 1 Results



Unload
======

- Read black box parameters:
  - I=1 (Iteration)
  - S=10 (spares)
  - N=23 (NFF)
  - R=20 (Reliability)
  - T=12 (Turnaround)
  - B=10234567.64 (Budget)
  - A=2,4,8,2,6,5 (Availability)
  - C=3,6,1,2,4,7 (Cost)
  - F=7,6,8,9 (Flying)
  - G=3,4,2,1 (Grounded)

Notes:

- A and C will pass back 24 values for plotting the graph
- Flying = the number of planes to display in the air
- Grounded = the number of planes to display on the ground
- Flying & Grounded will pass back 8 values to be displayed at each quarter

Animation
========

1) Press “Go” button
2) Support options screen disappears
3) Aircraft launch animation
4) Planes fade –in/out as each quarter is hit showing number in air and on ground
5) Graph is drawn simultaneously.
6) Total animation of 12 Seconds (2 points plotted every second)
7) Actual number is shown in addition to the corresponding number of planes
8) Graph is in 3 parts and is preserved per iteration

Go Button
========

On the 1st Iteration, the “Go” button can only be pressed if the budget is >=0.

Visual consideration
===============

The Range will probably be between 65 → 85 (75 is probable). Therefore, there would only be 2 aircraft that vary between air and ground. Possible diamond formation.

3) Round 2/3 & Round 2/3 Results

As per previous 2 screens. The values passed back from the black box are pre-populated in the Support Options boxes. The Upper & Lower limits for each parameter are reset as per the following rules:

- Spares & Reliability:
  - The existing value becomes the LOWER limit
- NFF & Turnaround:
  - The existing value becomes the UPPER limit

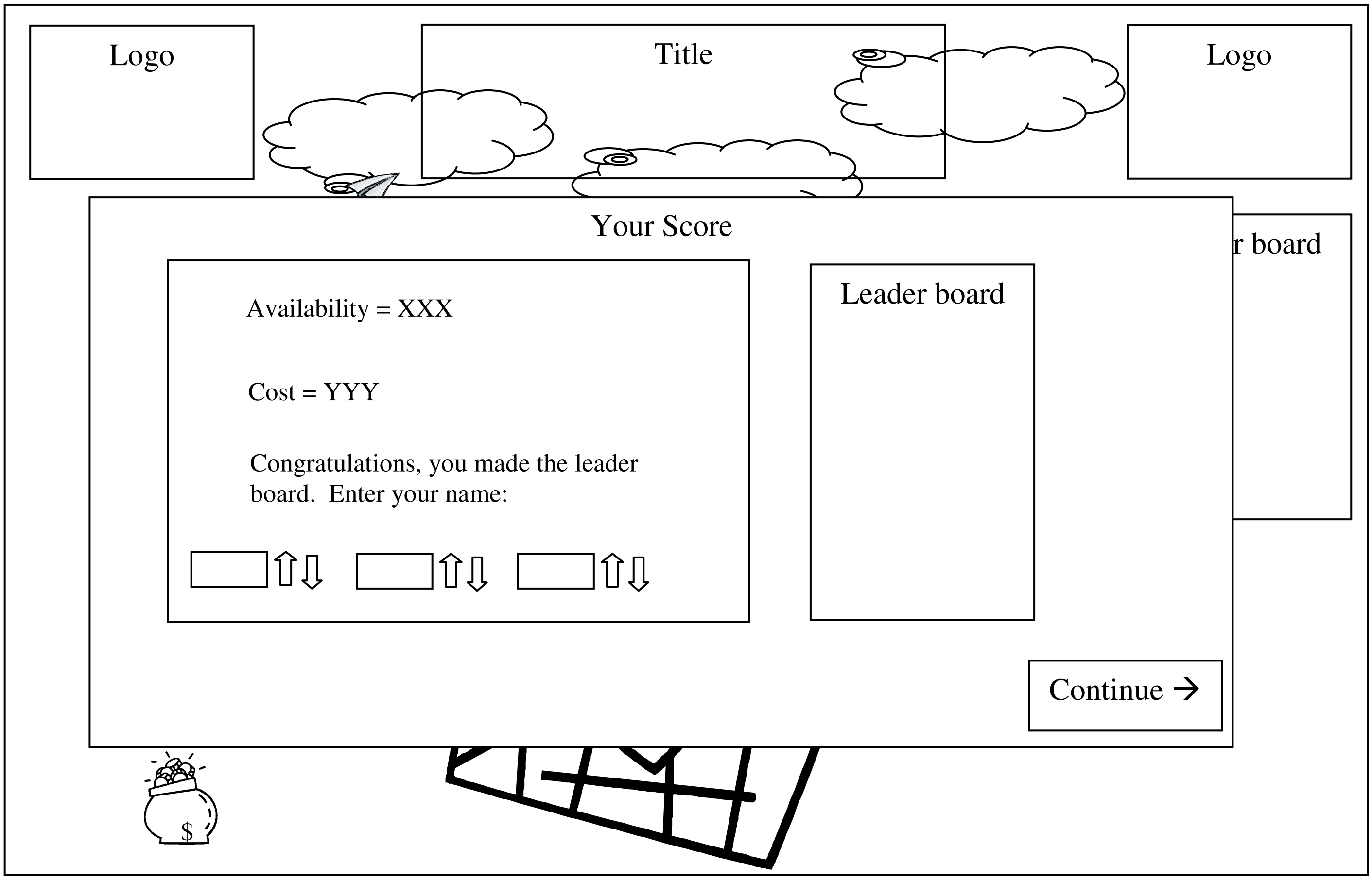
The user amends the parameters and then hits "Go" to send the modified values to the black box. The black box then returns new values for displaying back to the user (as a graph / planes / Support Options values).

Money pot
=======
The money pot shows the budget available at the beginning of each iteration. In Iteration 2 and 3, the money pot could be negative. Therefore, the visualisation needs to accommodate this.

Negative Budget
==========
It is feasible that the budget could be negative on iteration 2 & 3. If it is, then the 4 parameters are read-only and the "Go" button is still enabled.

Handshaking
========
The flash application calls the black box via a socket server. The Flash application then listens for when data is pushed from the socket (in the form of UTF-8 data) or writes a UTF-8 string to the socket. The black box then generates an XML string containing the data and passes it back through the socket server.

4) Score Screen



Availability = XXX

Cost = YYY

Congratulations, you made the leader board. Enter your name:

Onload
======

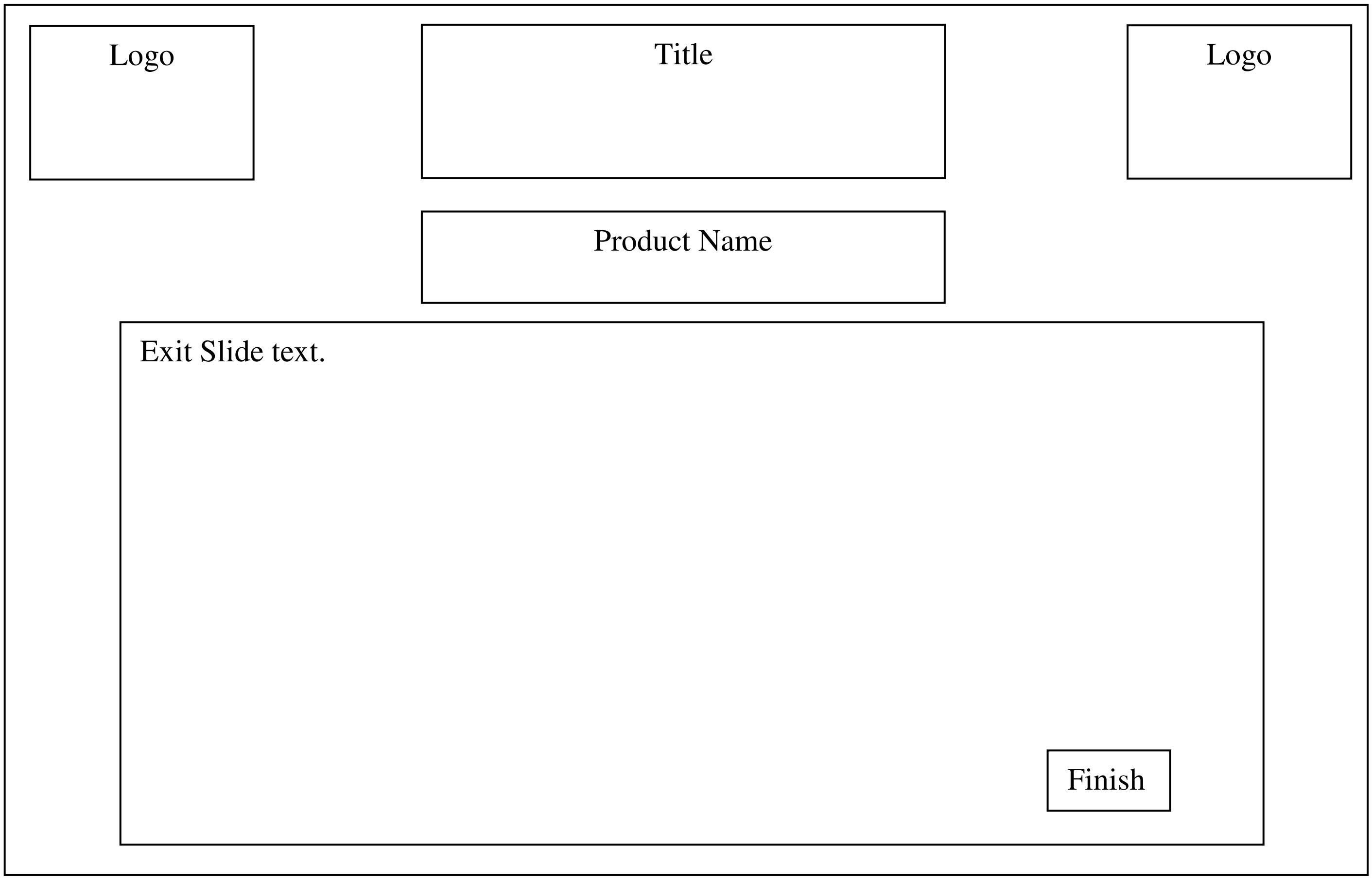
On Iteration 3, the black box will also pass back the following parameter:

- V=123456

V = the final Value (i.e. their score) that will be used to determine whether they go onto the leader board.

Note: The Continue button goes to an Exit Slide screen.

1) Exit Slide Screen



The Finish button restarts the game by going back to the Intro screen.

5) Lookup tables

Spares (Pack) *Initial value = 120*

Units	Step	Cost
n/a	0	0
n/a	1	0.4
n/a	…	…
n/a	100	40

Fixed price of 0.4 per unit. Cannot go lower than each iteration's starting value. Upper limit is as many as can be purchased within budget.
The value to pass to the blackbox = (*total spares packs – initial value*)

Reliability (hours) *Initial value = 15*

Units	Step	Cost
0	0	0
1	0.3	5
2	0.6	10
3	1.2	20
4	2.4	30
5	4.8	40

The value to pass to the blackbox = the number of units purchased in that iteration (0-5)

NFF (%) *Initial value = 45%*

Units	Step	Cost
0	0	0
1	-5%	2
2	-10%	4
3	-15%	8
4	-20%	10
5	-25%	15

The value to pass to the blackbox = the number of units purchased in that iteration (0-5)

Turnaround (days) *Initial value = 45*

Units	Step	Cost
0	0	0
1	-8	2
2	-16	4
3	-24	8
4	-32	16
5	-40	32

The value to pass to the blackbox = the number of units purchased in that iteration (0-5)

Notes:
Increasing Spares & reliability is a ***good*** thing and there is a cost associated to it.
Up Arrow = Subtract the cost from the budget

Down Arrow = Add the cost to the budget

Decreasing NFF and Turnaround is a *good* thing and there is a cost associated to it.

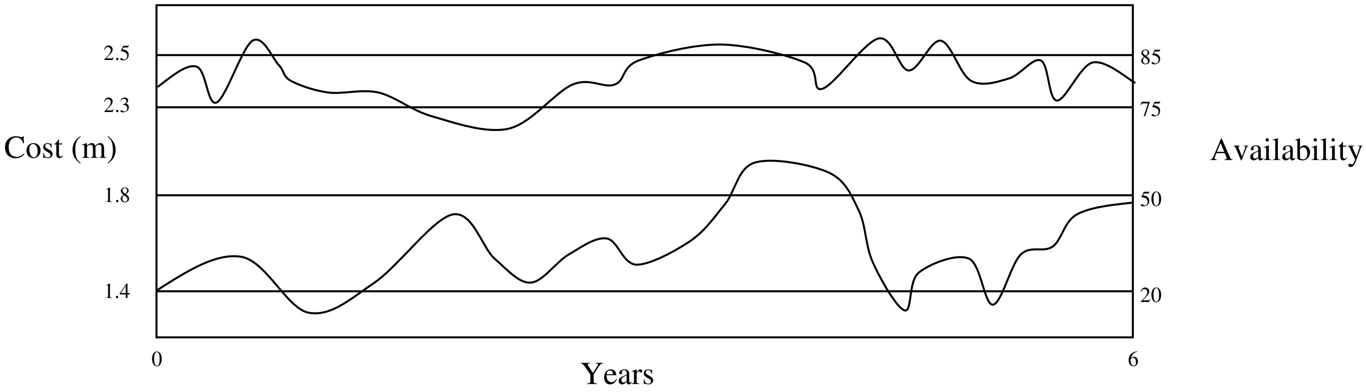
Up Arrow = Add the cost to the budget

Down Arrow = Subtract the cost from the budget

Q: Are the values in Millions or Thousands of Dollars?

Q: Do we require decimal places?

Also, we need an example graph showing the 2 sweet zones and the 2 y-axis ranges required to display the results.



Initial Parameters

Loaded when the Flash application starts

```
  <?xml version="1.0" encoding="UTF-8" ?>
<initParams>
<reliability>
  <unit>0</unit>
  <value>0</value>
  <cost>0</cost>
  <unit>1</unit>
  <value>0.3</value>
  <cost>05</cost>
  <unit>2</unit>
  <value>0.6</value>
  <cost>10</cost>
  <unit>3</unit>
  <value>1.2</value>
  <cost>20</cost>
  <unit>4</unit>
  <value>2.4</value>
  <cost>30</cost>
  <unit>5</unit>
  <value>4.8</value>
  <cost>40</cost>
  </reliability>
<nff>
  <unit>0</unit>
  <value>0</value>
  <cost>0</cost>
  <unit>1</unit>
  <value>-5</value>
  <cost>2</cost>
  <unit>2</unit>
  <value>-10</value>
  <cost>4</cost>
  <unit>3</unit>
  <value>-15</value>
  <cost>8</cost>
  <unit>4</unit>
  <value>-20</value>
  <cost>10</cost>
  <unit>5</unit>
  <value>-25</value>
  <cost>15</cost>
  </nff>
<turnaround>
  <unit>0</unit>
  <value>0</value>
  <cost>0</cost>
  <unit>1</unit>
  <value>-8</value>
  <cost>2</cost>
  <unit>2</unit>
  <value>-16</value>
  <cost>4</cost>
  <unit>3</unit>
  <value>-24</value>
  <cost>8</cost>
  <unit>4</unit>
  <value>-32</value>
  <cost>16</cost>
  <unit>5</unit>
  <value>-40</value>
  <cost>32</cost>
  </turnaround>
<spares>
  <sparesCost_ea>0.4</sparesCost_ea>
  <leadtime_mths>6</leadtime_mths>
  </spares>
<currentReliability>15</currentReliability>
  <currentNFF>45</currentNFF>
  <currentTurnaround>45</currentTurnaround>
  <currentSpares>120</currentSpares>
  <currentBudget>17</currentBudget>
</initParams>
```

Request Parameters

Sent to the Flash application at the end of each iteration

```
<?xml version="1.0" encoding="UTF-8" ?>
- <requestParams>
  <gameID>1234</gameID>
  <iteration>1</iteration>
  <reliabilityStep>2</reliabilityStep>
  <nffStep>1</nffStep>
  <turnaroundStep>3</turnaroundStep>
  <sparesBought>5</sparesBought>
  <currentBudget>2.4</currentBudget>
  </requestParams>
```

Result Parameters

Sent to the Flash application in response to the Request Parameters

```
<?xml version="1.0" encoding="UTF-8" ?>
- <resultParams>
  <gameID>1234</gameID>
  <iteration>1</iteration>
  <currentReliability>15.6</currentReliability>
  <currentNFF>40</currentNFF>
  <currentTurnaround>21</currentTurnaround>
  <currentSpares>125</currentSpares>
  <currentBudget>22.21</currentBudget>

<percentFlown>67.25,65.91,67.25,66.58,68.60,67.25,74.45,74.71,79.79,77.37,84.38,82.23,87.41,83.23,85.71,84.
00,86.57,83.14,87.43,82.29,86.57,85.71,85.71,82.29</percentFlown>
  <monthTotal>-5.82,-5.93,-5.82,-0.21,-0.03,-
0.15,0.49,0.54,0.94,0.51,1.38,1.28,1.53,1.41,1.55,1.48,1.58,1.43,1.60,1.37,1.58,1.55,1.55,1.37</monthTotal>
  <inAir>7,7,8,8,9,8,9,8</inAir>
  <onGround>3,3,2,2,1,2,1,2</onGround>
  </resultParams>
```

The final time will also contain the final Value score to determine entry to the leader board:

```
  <totalScore>1234</totalScore >
```