

Instructions for using XML in ATC-lab^{Advanced}

The University of Queensland

Selina Fothergill, Shayne Loft and Andrew Neal

July 2008

This document presents a step-by-step guide on how to create a script using the Extensible Markup Language (XML). The purpose is to illustrate the programming control afforded by ATC-lab^{Advanced}. It contains a selection of trials and features from published experiments that have been conducted to date.

Some important information to start with:

1. Each XML script has general sections; which include Configuration, Map and Aircraft, Rating Scales and Experiment Presentation. The details included in these sections can differ between experiments. In the example XML script we have included several different instantiations of these details to demonstrate the programming control components of ATC-lab^{Advanced}. Thus, the example xml is not an experiment we have conducted per se but rather a selection of different experiments.
2. We recommend that users initially base their experimental XML script on the template provided. It is much easier to modify the specifications from the template, rather than create a new one. Once comfortable using the scripting language, users may wish to create their own unique scripts.
3. Scripts being developed and pilot tested in the XML language can be viewed in several programs. We recommend Scintilla Text Editor (SciTE). This can be downloaded at the following address: <http://scintilla.sourceforge.net/SciTEDownload.html> In SciTE the user can specify the line numbers beside each text line. By selecting 'line numbers' from the 'view' menu button. This will assist the user to quickly locate the line and character where errors have occurred. Alternatively, scripts can be opened in Notepad or Wordpad, which is available in the Microsoft Windows 'Start' menu.
4. The scripts must end with the extension .XML. To start running the script, drop the XML file into the pact.exe application. The first scenario should begin. To move on to the next scenario before it is scripted to finish, press ['ctrl', 'alt' and 'N'] together. To quit the program, press ['ctrl', 'alt' and 'Q'] together (this action will generate a data log file). For more information regarding installation of ATC-lab^{Advanced} please Installation Intructions.pdf.
5. If there are errors in a script, when the XML file is dropped into the pact application the ATC program will not start. An error message will appear, telling the user where the error is located in the XML script. For example, if the phrase 'atc:map' is not written at the bottom of a map's specifications, the error might read something like:

*RunTime error: SAX parsing exception [C:\Desktop\XML script] (line=312, char=5)
Expected end of tag 'atc:map'*

To locate errors in scripting, it is beneficial to have the line numbers beside each text line (refer to point 3 above).

6. In combination with the math spreadsheets, users can determine aircraft starting coordinates from online drawing programs. One such example is AutoCAD, which can provide (x,y) coordinates for points and map dimensions. For the conflict resolution experiment, the maps and routes were drawn in AutoCAD. The coordinates for the vertices of sectors and points along a route were provided by AUTOCAD. These (x,y) coordinates were then re-typed into the XML script for the development of scenario maps. To obtain the starting coordinates for aircraft required on these routes, circles (to represent aircraft track symbols) were drawn onto the maps in AutoCAD. The AUTOCAD program also lists the coordinates of these circles. These coordinates were also typed into the XML script for aircraft starting positions. Some earlier versions of AutoCAD are available for free download over the web (e.g., the 2008 version at <http://www.brothersoft.com/autocad-download-51230.html>).

A. CONFIGURATION

Aircraft Performance Data

Here the user specifies aircraft performance data. Aircraft types are listed at the start of the XML script at 'aircraft performance data' (line 16 as viewed in the Scintilla Text Editor). Aircraft will climb, descend and cruise at their average performance rates stated in this section of the xml. The only exception is when a level requirement solution is issued (refer to training manual). The XML script refers to the text file called acperf.txt. This file contains aircraft performance profiles for a variety of aircraft types. As discussed below, the aircraft type is described when specifying the details of each individual aircraft.

Instructions

Here the user specifies the content and presentation style (but not the timing of presentation, which is specified in the Experiment Presentation section) of task instructions. The content and presentation style of three instructions are specified on lines 42 through 96. Each instruction must be labelled (e.g., 'act:inx:work2'). The user has control over the size and position of the text box, the text, and the size/colour of the font. These and other features are illustrated in the example below;

```
<atc:instruction atc:idx="work2">
<atc:text>
<![CDATA[<qt>
<center>
<table width="60%" bgcolor="darkCyan" border="0" cellspacing="0"
cellpadding="50">
<tr><td align="left"><font face="verdana" size="1">
<p align="center"><font size="+4">HIGH WORKLOAD</font></p>
<p align="center"><font size="+1">When making your decisions for the NEXT
FOUR pairs of aircraft please assume that you are under conditions of HIGH
WORKLOAD.</font></p>
<p align="center"><font size="+1">By high workload we mean that you are
very busy and under excessive time pressure. You are struggling to find the
time to accomplish your tasks. </font></p>
<p align="center"><font color="red"><b>Press the [SPACE] key to
continue<b></font></p>
</font></td></tr>
</table>
</center>
```

```
</qt>]]>
</atc:text>
<atc:keyEvent>Space</atc:keyEvent>
</atc:instruction>
```

MAP DEFINITIONS

Experimental Parameters

In this section, the user specifies the update rate of the simulation in ms (line 112). In the example shown, the update rate of the simulation is set at one update per 5 seconds. This means the aircraft will update position (and transition state) every 5 seconds.

Below this is the scenario tester function. In the example on line 113 it is set at 1. However, it can be set faster which allows the user to view (at a faster speed) the air traffic scenarios that are being developed.

On lines 115-121 the specific colours used to denote aircraft transitional states are set (note that the positions in sectors where aircraft automatically begin climbing or descending are specified later in the xml). The different aircraft states are described below (also see the Training Manual. pdf)

- Non_colour refers to the color of aircraft when not under participants jurisdiction (but approaching the sector)
- Annonced_colour is the color of aircraft when they are 15 minutes from the sector boundary
- Proposed_colour is the color of aircraft that are read to be accepted (they can also be set to blink)
- Accepted_colour is the color of aircraft in the sector (under jurisdiction) of the participant.
- Overout_colour is the color of aircraft when soon to exit the sector
- Handoff_colour is the color of aircraft alter being handed off and accepted by the controller in the next sector
- Nomore_colour refers to the colour of aircraft after formally leaving the sector.

The aircraft can be also be set to be always black by simply wrapping parenthetical symbols <!-- text here --> around lines 115-121 (see Removing Unwanted Tex section)

Map Region

The user can specify the total region of the airspace in x and y coordinates. For example on line 130 the region' for map 1 is defined as

```
<atc:region atc:y_dim='194.25' atc:y='-50' atc:x_dim='259' atc:x='-30' />
```

Note that if you change the size of a sector, you need to keep the y to x ratio at .75 (194.25 / 259 = .75) for XX screens and a ratio of 0.6 for wide screens

Map Waypoints

The user can specify the (x,y) coordinates for various waypoints on a map. These are written at 'atc:location' beneath the map dimensions. Waypoints are also named here at 'atc:idx'. For map 1 in the example script, these are written on lines 132-162. .

```
<atc:location atc:y='177.36635' atc:x='-20.8663' atc:visible='off' atc:idx='A_3' />
<atc:location atc:y='84.5' atc:x='39' atc:visible='off' atc:idx='BLACK' />
<atc:location atc:y='50' atc:x='115' atc:visible='off' atc:idx='C_3' />
```

Users may also select whether waypoint names are visible or not to the participant. This is done by setting the 'atc:visible' function to either 'on' or 'off'. For map 1 in the example script, the waypoint 'BLACK' on line 133 is set to visible ('on') and will be displayed to participants.

```
<atc:location atc:y='84.5' atc:x='39' atc:visible='on' atc:idx='BLACK' />
```

Map Route Structures

The user can create routes by specifying which waypoints are connected to each other. These routes need to be named at 'atc:idx'.

The waypoints that connect to form routes are written at 'atc:location'. The first route on the first map (map 1) in the example script is on lines 164-170.

```
<atc:route atc:idx='R1_3'>
<atc:pointref atc:location='A_3' />
<atc:pointref atc:location='BLACK' />
<atc:pointref atc:location='J_3' />
<atc:pointref atc:location='O_3' />
<atc:pointref atc:location='P_3' />
</atc:route>
```

Defining Sectors

The user can specify the vertices of a sector on a map. For example, to create a rectangular sector, the four vertices need to be defined at 'atc:vertex'. Sector West on map 1 in the example script has (x,y) coordinates for the four vertices. These are on lines 236-240.

```
<atc:sector atc:idx='SECTORWEST3'>
<atc:vertex atc:y='0' atc:x='0' />
<atc:vertex atc:y='-10' atc:x='-39' />
<atc:vertex atc:y='96' atc:x='-39' />
<atc:vertex atc:y='96' atc:x='0' />
</atc:sector>
```

The user can also choose to specify the dimensions for a circular (e.g., approach) sector. The radius and (x,y) coordinates need to be specified at 'atc:arc'. An example for map 1 is on lines 209-211. When defining multiple sectors, it is necessary to give them all different names (e.g., TOWER3, TOWER4 etc).

```
<atc:sector atc:idx='TOWER3'>
<atc:arc atc:r='16.5' atc:y='50' atc:x='115' />
</atc:sector>
```

Activating Sectors

Sectors which the user wishes to be active can be set by defining the vertices/coordinates of the active sector and labelling it 'active' at 'atc:status'. An example from map 1 is on lines 249-254.

```
<atc:sector atc:status='active' atc:idx='ACTIVE3'>
<atc:vertex atc:y='0' atc:x='0' />
<atc:vertex atc:y='0' atc:x='175' />
<atc:vertex atc:y='96' atc:x='175' />
<atc:vertex atc:y='96' atc:x='0' />
</atc:sector>
```

Note that `</atc:map>` needs to be written at the end of each map. For example, see line 256 where this has been placed at the end of map1 and before map2.

SKY DEFINITIONS

Here the user specifies the aircraft that are going to be presented in each trial.

Trials are given a name (e.g., sky1) at 'atc:sky atc:idx'. At the end of specifying each aircraft that will be presented on this trial, the text `</atc:sky>` needs to be scripted. For example, see sky1 which begins on line 391 and ends on line 456.

The following details are scripted for aircraft contained in each trial;

Aircraft Type

The user writes the aircraft type at 'atc:type'. For example, in the first group of aircraft called 'sky1' (line 391), the first aircraft is a DH8C. This is specified on line 393.

Aircraft Callsigns

Aircraft callsigns can be specified on the same line as the aircraft type, after 'atc:idx'. For example, in the first group of aircraft called 'sky1', the first aircraft (a DH8C) is called 'SSQ1493'. This is specified on line 393.

Starting Time

Users can specify aircraft to appear in the sector at certain times in the trial. For example, SSQ1493 is scheduled to appear in the sector at time 20 seconds (394). Aircraft with this line missing are presented at the start of the trial by default. The majority of aircraft in ATC simulations we have run to date have used this default function.

Starting Level

The starting position of an aircraft is specified at the first 'atc:altitude:' in an aircraft's parameters. For example, in the first group of aircraft, the first aircraft (SSQ1493) will start at 6000 feet (referred to as flight level 060). This is on line 395.

Starting Speed

The starting speed of an aircraft is specified at the first 'atc:velocity:' in an aircraft's parameters. For example, in the first group of aircraft, the first aircraft (SSQ1493) will start at 200 knots (represented as '20' in the aircraft's label). This is on line 396. It is important to only assign aircraft speeds that the aircraft type can perform. If this is not correct, the script will not run in the application and an error message will be generated.

Starting Coordinates (flight path)

The starting position of an aircraft is specified at the first 'atc:point atc:' in an aircraft's parameters. For example, in the first group of aircraft, the first aircraft (SSQ1493) will start at (98, 58) (line 398). Note that the x and y coordinates are written in reverse order in the script. The user can use the mathematical spreadsheets or programs like AutoCAD to determine the aircraft track symbols' coordinates.

Cleared Flight Level

The cleared flight level for an aircraft is specified after the starting coordinate line. It is written at 'atc:altitude:' in an aircraft's parameters. For example, in the first group of aircraft, the first aircraft (SSQ1493) will climb to 23000 feet (flight level 230).

Flight Path (future coordinates the aircraft will travel through)

Future coordinates that the aircraft will travel through are specified after 'atc:point atc:' lines. For example, in the first group of aircraft, the first aircraft (SSQ1493) will pass through (38, 85) (line 401) and then (-66.2533, 140.14865) (line 402). An important difference between scripting the starting coordinate and future coordinates is that there is **NO** backslash at the end of the starting coordinate line. Also, <atc:flightpath> needs to be wrapped around an aircraft's flightpath.

Finally, each aircraft needs the text </atc:aircraft> at the end of its parameters. An example of one aircraft (line 393-404) is below:

```
<atc:aircraft atc:type='DH8C' atc:idx='SSQ1493'>
<atc:start>20000</atc:start>
<atc:altitude>06000</atc:altitude>
<atc:velocity>200</atc:velocity>
<atc:flightpath>
<atc:point atc:y='58' atc:x='98'>
<atc:altitude>23000</atc:altitude>
</atc:point>
<atc:point atc:y='85' atc:x='38' />
<atc:point atc:y='140.14865' atc:x='-66.2533' />
```

```
</atc:flightpath>  
</atc:aircraft>
```

Weather Patterns

Weather pattern shapes (e.g., ellipses) can be set by specifying the relevant points on the map. These weather patterns are listed under the trial they are required to be presented in. For example, in Sky 2, line 596:

```
<atc:area atc:type='weather' atc:idx='STORM2'>  
<atc:ellipse atc:h='50' atc:w='37.5' atc:y='44.63245' atc:a='45'  
atc:x='34.9497' />  
</atc:area>
```

RATING SCALES

Here the user can specify the content and format of the questions participants are asked throughout the experiment [the timing of the presentation of these questions is specified later in the Experiment Presentation section]. See line 690-861 for an illustrative example of two rating scales. As described below, included in this programming code is specification of manner in which individuals respond and transit the series of linked rating scales. Note that the easiest way to get used the how to code rating scales is the systematically change some of the programming features described below and then examine the resulting change in presentation.

The first section (line 692-710) describes the appearance (size, placement on screen) of the border of the first rating scale, and the context/font of box text title (i.e., Intervention Decision).

```
<atc:widget atc:class="dialog" atc:name="dlgInterveneSelection">  
<atc:property atc:name="geometry">  
<atc:rect>  
<atc:x>100</atc:x>  
<atc:y>100</atc:y>  
<atc:w>200</atc:w>  
<atc:h>100</atc:h>  
</atc:rect>  
</atc:property>  
<atc:property atc:name="caption">  
<atc:string>Intervention Decision</atc:string>  
</atc:property>  
<atc:property atc:name="font">  
<atc:font>  
<atc:family>Arial</atc:family>  
<atc:pointsize>10</atc:pointsize>  
<atc:bold>1</atc:bold>  
</atc:font>  
</atc:property>
```

The next section (lines 712-750) describes the response options for that rating scale. The question displayed in this rating scale is 'Would you intervene now or in the future to assure separation between aircraft'. As shown below, one of the displayed response options is 'Definitely' (the others are Likely, Unlikely and Definitely Not). Also specified is the manner

in which these options are chosen - in this case it is a 'push button event' (line 723), which means the participant clicks the option with the mouse.

```
<atc:layout atc:type="vbox">
<atc:stretch/>
<atc:widget atc:class="label" atc:name="lblInterveneSelection">
<atc:property atc:name="text">
<atc:string>Would you intervene now (or in the future) to assure
separation?</atc:string>
</atc:property>
</atc:widget>
<atc:space/>
<atc:widget atc:class="frame" atc:name="frameInterveneSelection">
<atc:layout atc:type="hbox">
<atc:stretch/>
<atc:widget atc:class="pushbutton" atc:name="btnIntervene1">
<atc:property atc:name="text">
<atc:string>Definitely</atc:string>
```

A second rating scale box is then defined using the same process detailed as above (lines 753-794).

The next step is to define the manner in which participants will transit through these rating scales (i.e., how the rating scales are linked – if they indeed are) (lines 799-859). Basically, this part of the XML describes the sequences of events that happen when participants click different response options on the two scripted rating scales (again the easiest way to learn this is to change some of the parameters and observe resulting presentation)

The section on lines 801-804 specifies that 'InterveneSelection' response box will be shown at the 'start' of any trial that is scripted to be presented on (see Experiment Presentation)

```
<atc:connection>
<atc:signal>started</atc:signal>
<atc:slot atc:rx="dlgInterveneSelection">show</atc:slot>
</atc:connection>
```

There are then four different paths (lines 806-859) depending on whether the participant chooses Definitely, Likely, Unlikely or Definitely not. For example, if participants choose definitely (btnIntervene 1) then the second response box called 'Intervene Selection' will be displayed (as defined earlier). When the participant then clicks on the Time or Distance response option the response display terminates and the next trial is presented.

```
<!-- Def Intervene Path -->

<atc:connection>
<atc:signal atc:tx="btnIntervene1">clicked</atc:signal>
<atc:slot>pause</atc:slot>
<atc:slot atc:rx="dlgWhen">position</atc:slot>
<atc:slot atc:rx="dlgWhen">show</atc:slot>
<atc:slot atc:rx="dlgInterveneSelection">hide</atc:slot>
</atc:connection>

<atc:connection>
<atc:signal atc:tx="btnTimeOK">clicked</atc:signal>
<atc:slot>terminate</atc:slot>
</atc:connection>
```

```
<atc:connection>
<atc:signal atc:tx="btnDistanceOK">clicked</atc:signal>
<atc:slot>terminate</atc:slot>
</atc:connection>
```

EXPERIMENT PRESENTATION

The user now specifies the order in which the different components of the experiment will be presented to the participant. This includes specifying which trials (groups of aircraft) will be presented with which experimental parameter settings, map definitions, instructions, and response boxes.

This section runs from line 874-915. Each phase of the experiment is given a name, such as Set 1. In turn, each trial in a phase can be given names such as `atc:idx="E000-T01"`.

Set 1 is designed to give some examples of the types of trials presented by the Fothergill and Neal (in press) conflict resolution experiment. This part of the experiment is using the default parameter settings, displaying aircraft set 'sky2' on 'map1' (line 879). In the next trial (E000-T02), the experimenter is displaying aircraft set 'sky1' on 'map2' (883). The order of trials can thus be easily changed (e.g., for counterbalancing purposes). On lines 880 and 884 the code `<atc:keyEvent>Space</atc:keyEvent>` specifies that participants can move on to the next trial by pressing the space bar.

```
<atc:phase atc:idx="set1">
<atc:trial atc:idx="E000-T01" atc:param="default" atc:map="map1"
atc:sky="sky2">
<atc:keyEvent>Space</atc:keyEvent>
</atc:trial>
<atc:trial atc:idx="E000-T02" atc:param="default" atc:map="map2"
atc:sky="sky1">
<atc:keyEvent>Space</atc:keyEvent>
</atc:trial>
</atc:phase>
```

You will see that in Set 2 we have included some examples from the types of trials presented in the Loft et al. (2007) conflict detection experiment. This part of the experiment is also using the default parameter settings. Note that if we wanted to say present set 2 of the experiment at a faster update rate, or with a different aircraft notification system, we would have simply specified an alternative Experimental Parameters setting and referred to this setting in this Experiment Presentation section. Note that in Set 2 instructions are presented (lines 892 and 902) and rating scales are presented (`atc:ui="ui001"`). Also note that trials are scripted to be presented for 120 second (`<atc:timeEvent>120</atc:timeEvent>`)

OTHER PROGRAMMING FEATURES

Pausing and Resuming Tasks

Users can pause (stop) the scenarios at any stage by pressing the 'S' button on the keyboard. This is particularly useful if the study contains instructions, verbal interview protocols or if the experimenter needs to conduct handover/takeovers with controllers in the field. Users can also resume (advance) the scenarios at any stage by pressing the 'A' button on the keyboard.

Data Logging

Participants' actions and aircraft movements will be logged into a text file in the same file folder as the script, after it has run. When the task is finished press ALT & CTRL & Q at the same time and the task will terminate. The data will include a log of ALL events and participant actions time stamped at the millisecond level. This log file will have the same title as the xml script from which it was generated from. Users will need to write code in order to extract their particular variables of interest to relevant data analysis programs (e.g., SPSS, Microsoft Excel)

Prediction Tools

Regular prediction tools (e.g., bearing and range lines) available to controllers are built into the software platform. The user can decide which (if any) of these tools they wish their participants to be aware of (and thus use). How to use these tools or turn them off is described in the training manual.

Removing Unwanted Text

If the user wishes to alter the script temporarily by removing certain information (e.g., several aircraft), which they may wish to use later, parentheses can be used to temporarily 'block out' the information from the script. This is particularly useful when the user is developing the script and wishes to check certain details on a particular scenario quickly. Alternatively, it can be used to write titles (e.g., trial presentation order) of sections into the script, which are not read by the application. To achieve this, the user needs to wrap these parenthetical symbols <!-- text here --> around the text they do not wish to use.