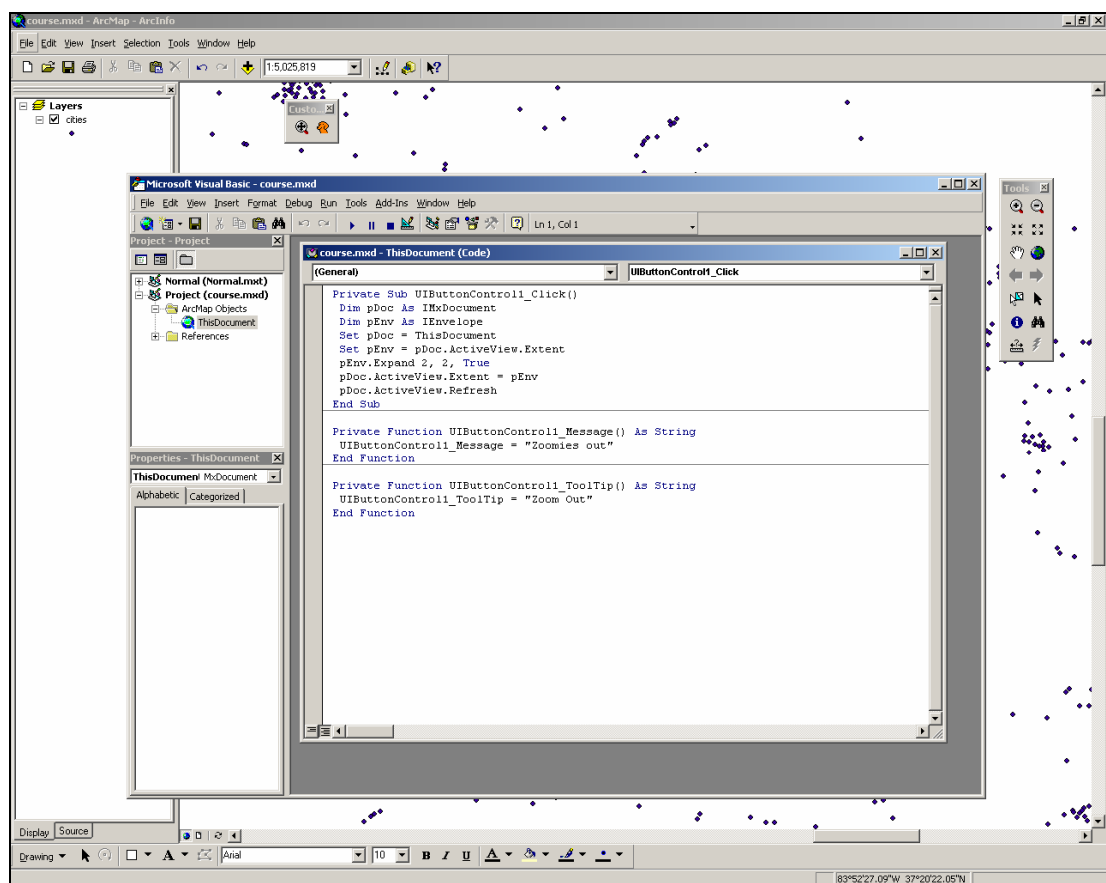


# Programming in ArcGIS using ArcObjects and AML

ArcObjects is the development environment of the Desktop ArcGIS applications ArcMap, ArcCatalog and ArcScene. It is used to customise and extend ArcGIS using the embedded Visual Basic for Applications (VBA).

Arc Macro Language (AML) is the programming language of Workstation ArcInfo. It is used to communicate with the Arc Environment and can be used to build programs of frequently used ArcInfo commands, or menu-driven applications.

Prior working knowledge of ArcGIS and Workstation ArcInfo is assumed throughout this Guide.



Document code: **Guide 93**  
Title: **Programming in ArcGIS using ArcObjects and AML**  
Version: **3.0**  
Date: **June 2007**  
Produced by: **University of Durham Information Technology Service**

**Copyright © 2007 University of Durham Information Technology Service**

### **Conventions:**

In this document, the following conventions are used:

- A typewriter font is used for what you see on the screen.
- A **bold typewriter font** is used to represent the actual characters you type at the keyboard.
- A *slanted typewriter font* is used for items such as filenames which you should replace with particular instances.
- A **bold font** is used to indicate named keys on the keyboard, for example, **Esc** and **Enter**, represent the keys marked Esc and Enter, respectively.
- A **bold font** is also used where a technical term or command name is used in the text.
- Where two keys are separated by a forward slash (as in **Ctrl/B**, for example), press and hold down the first key (**Ctrl**), tap the second (**B**), and then release the first key.

## Contents

### Part 1

<b>1</b>	<b>Introduction</b> .....	<b>1</b>
1.1	Guide overview .....	1
1.2	ArcObjects introduction .....	1
<b>2</b>	<b>Customising the user interface</b> .....	<b>1</b>
2.1	Alter an existing toolbar .....	2
2.2	Add a new custom toolbar .....	2
<b>3</b>	<b>ArcObjects Help</b> .....	<b>3</b>
<b>4</b>	<b>Writing VBA code</b> .....	<b>4</b>
4.1	Tool customisation .....	5
4.1.1	Add a tool tip .....	5
4.1.2	Add a tool description .....	5
<b>5</b>	<b>Using ArcObjects Developer Kit samples</b> .....	<b>6</b>
<b>6</b>	<b>Using scripts from ArcScripts</b> .....	<b>6</b>
<b>7</b>	<b>Using ArcObjects in ArcMap</b> .....	<b>7</b>
7.1	Using VBA code in the Field Calculator .....	7
7.2	Using VBA code in the Raster Calculator .....	8
<b>8</b>	<b>Where next?</b> .....	<b>9</b>

### Part 2

<b>9</b>	<b>AML Introduction</b> .....	<b>10</b>
<b>10</b>	<b>AML elements</b> .....	<b>10</b>
10.1	Directives .....	10
10.2	Variables .....	11
10.3	Functions .....	11
<b>11</b>	<b>Creating and running an AML program</b> .....	<b>12</b>
11.1	Quoted strings .....	13
<b>12</b>	<b>Decision making</b> .....	<b>13</b>
<b>13</b>	<b>Using loops</b> .....	<b>14</b>
<b>14</b>	<b>Accessing files</b> .....	<b>14</b>
<b>15</b>	<b>Creating menus</b> .....	<b>14</b>
<b>16</b>	<b>Creating AMLs automatically</b> .....	<b>15</b>
<b>17</b>	<b>De-bugging AML programs</b> .....	<b>15</b>
17.1	Testing .....	15
17.2	Suppressing message when running AML .....	16

<b>18</b>	<b>Modular programming</b> .....	<b>16</b>
<b>19</b>	<b>AML help</b> .....	<b>16</b>
<b>20</b>	<b>Open Development Environment (ODE)</b> .....	<b>16</b>
<b>21</b>	<b>An example AML application</b> .....	<b>17</b>
21.1	Step 1 .....	17
21.2	Step 2 .....	17
21.3	Step 3 .....	17
21.4	Step 4 .....	18
<b>22</b>	<b>Creating AMLs from ArcToolbox</b> .....	<b>18</b>
<b>23</b>	<b>Further information</b> .....	<b>18</b>
23.1	Books and manuals .....	18
23.2	Other sources of information .....	19

# Part 1

## 1 Introduction

### 1.1 Guide overview

Part 1 of this Guide covers using ArcObjects in Desktop ArcGIS (ArcMap, ArcCatalog and ArcScene). Part 2 covers using AML in Workstation ArcInfo.

### 1.2 ArcObjects introduction

ArcObjects is the development environment introduced with ArcGIS 8. It provides customisation for ArcMap, ArcCatalog and ArcScene and is based on Visual Basic for Applications (VBA) which is embedded in ArcGIS.

This Guide provides an introduction to ArcObjects, but does not attempt to teach VBA programming. A list of books, manuals and other resources is included at the end of this Guide for those wanting to learn more.

Why use ArcObjects?

- It provides access to functionality not available through the ArcMap, ArcCatalog or ArcScene interfaces.
- Allows customisation of the interface for end users.
- Provides the ability to add functionality written by a 3<sup>rd</sup> party (e.g. code from ESRI's ArcScripts).

There are three levels of customisation provided in ArcGIS:

- Customise the user interface (does not use ArcObjects directly).
- Write VBA code.
- Use an external development environment to create a stand-alone COM component (using for example Visual Basic, Visual C++ or Visual J++). These can be used to incorporate ArcObjects into non-GIS applications, but they still need an ArcGIS licence to run.

The first two levels will be introduced here. For information on the third level see the books and manuals section at the end of this Guide, in particular *Exploring ArcObjects (Vols. I & II)*.

## 2 Customising the user interface

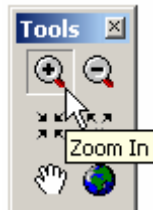
Customising the user interface in ArcMap, ArcCatalog or ArcScene requires no programming. The following examples show how to alter an existing toolbar and add a new custom toolbar.

Before beginning this section start ArcMap, select **File | Save**, create a new folder called **arcobjects** on your **J:\** drive (use the **Create New Folder** icon in the **Save As** dialog box) and save your ArcMap document as **J:\arcobjects\course.mxd**. This will be used to save all customisations from this course.

## 2.1 Alter an existing toolbar

Open ArcMap and select the menu **Tools | Customize**. ArcMap's state will change so that existing toolbars can be altered.

Click on the zoom-in tool on the floating toolbar and drag it to the main menu:

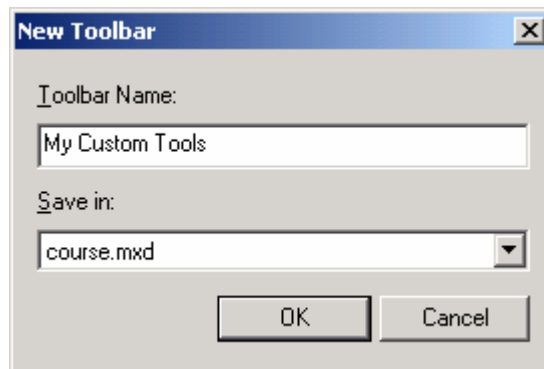


The tool will stay there and can be used as normal. Before finishing, drag it back to its original position and click **Close** on the Customize dialog box.

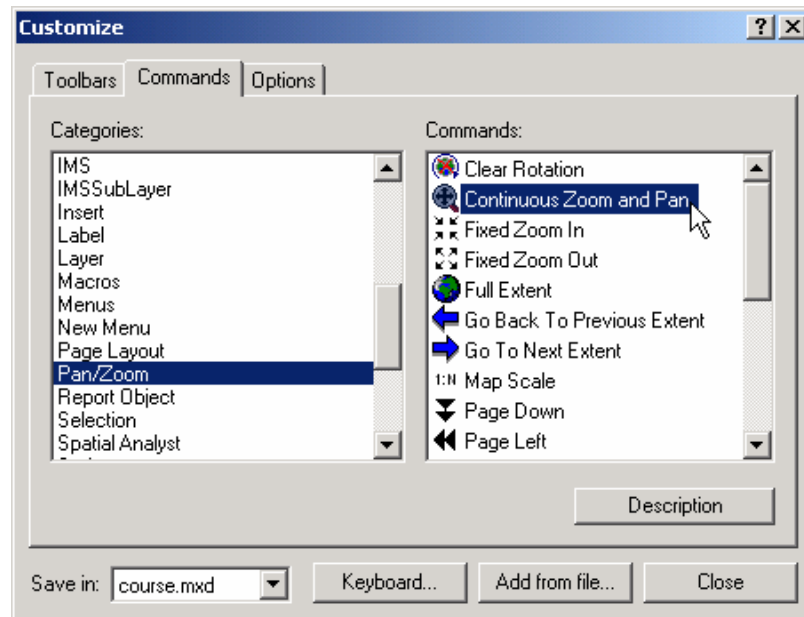
Selecting **Tools | Customize** in ArcCatalog or ArcScene has the same effect.

## 2.2 Add a new custom toolbar

- 1 To create a custom toolbar in ArcMap choose **Tools | Customize** from the menu bar, make sure the **Toolbars** tab is selected, and click on **New**.
- 2 Enter a name for the new toolbar, and set **Save in:** to **course.mxd**:



- 3 Click **OK**, then choose the **Commands** tab in the Customize dialog box.
- 4 Select the **Pan/Zoom** category on the left of the dialog box.
- 5 Drag the **Continuous Zoom and Pan** command icon from the right-hand pane onto your custom toolbar:



6 Click **Close** when finished.

Copy **T:\its\gis\cities.shp** and **T:\its\gis\hawaii\_cities.shp** to the folder **J:\arcobjects\** using ArcCatalog. Add **cities.shp** to ArcMap and try the new Continuous Zoom and Pan tool. Hint: use a combination of the left and right mouse button to zoom in/out and pan around the map.

To save the changes to the interface in course.mxd select **File | Save**. To use these changes in a future session course.mxd must be opened in ArcMap. Permanent changes to the interface are stored by ArcMap in the normal template Normal.mxt. This can not be altered by a user on the Networked PC Service.

On an office machine the changes to the user interface can be saved in Normal.mxt, and are stored under c:\arcgis\arcexe83\bin\template\ and will apply every time ArcMap is launched.

### 3 ArcObjects Help

Detailed help on ArcObjects is available in the **ArcObjects Developer Help**. To access this, select **Start | Programs | GIS | ArcGIS | ArcObjects Developer Help**. This is the main source of information for ArcObjects, and includes topics ranging from Getting Started to Object Model Diagrams. Some of the samples used in this Guide come from the **ArcObjects Developer Kit** which is one of the main topics described in the Developer Help.

There is an equivalent online resource on the internet called **ArcObjects Online** that includes updates made available since the software was released. It is at: [www.esri.com/arcobjectsonline](http://www.esri.com/arcobjectsonline)

## 4 Writing VBA code

Visual Basic for Applications is embedded in ArcGIS, and can be used to further customise it's applications. This means that an external editor is not needed, and you do not have to close down the application. Note that source code is visible to the user - to encrypt code an external development environment must be used.

The following examples shows how to add VB code to zoom out from a map.

- 1 Choose **Tools | Customize**.
- 2 Make sure the **Commands** tab is highlighted.
- 3 Set the **Save in:** box at the bottom left of the dialog box to the current project - **course.mxd**.
- 4 Select the **UIControls** category from the left-hand pane, and click on the **New UIControl** button.
- 5 Make sure **UIButtonControl** is selected and click **Create and Edit**.

The Visual Basic Editor will open.

- 6 Enter the following code in the Code window between **Private Sub** and **End Sub** (Indent each line one space):

```
Dim pDoc As IMxDocument
Dim pEnv As IEnvelope
Set pDoc = ThisDocument
Set pEnv = pDoc.ActiveView.Extent
pEnv.Expand 2, 2, True
pDoc.ActiveView.Extent = pEnv
pDoc.ActiveView.Refresh
```

- 7 Hint: When designing code the ArcObjects Component Help is invaluable. To access it click a method and press the F1 key. Click **Refresh** in the code just entered and press the F1 key. Help on this specific method will be displayed.
- 8 Close the Help when finished reading.

Note: For help with Visual Basic code (rather than specific ArcObjects code) select **Help | Microsoft Visual Basic Help** from the menu.

- 9 Close the Visual Basic Editor (**File | Close and Return to ArcMap**)
- 10 Choose **Tools | Customize**.
- 11 Make sure the **Commands** tab is selected and drag the **Project.UIButtonControl1** icon to your custom toolbar.
- 12 To change the default icon, right-click the new icon on your custom toolbar, select **Change Button Image** and choose an appropriate icon.

- 13 Click **Close** on the **Customize** dialog box when finished, and click on the new tool. The map should zoom out.

#### 4.1 Tool customisation

Additional tool customisation includes adding tool tips and providing a description for the tool.

##### 4.1.1 Add a tool tip

To display a tool tip (i.e. the label displayed when the mouse cursor is held over the tool icon), use the Visual Basic Editor to add a ToolTip procedure.

- 1 Choose **Tools | Macros | Visual Basic Editor**.
- 2 Select **ToolTip** from the **Procedure combo box**:



- 3 Enter the following code between **Private Function** and **End Function**:

```
UIButtonControl1.ToolTip= "Zoom Out"
```

- 4 Close the Visual Basic Editor (**File | Close and Return to ArcMap**)
- 5 Hover the mouse over your new tool – a label will appear over the tool with the words **Zoom Out**.

##### 4.1.2 Add a tool description

To add a tool description (i.e. the text displayed to the left of the status bar when the mouse cursor is held over the tool icon), use the Visual Basic Editor to add a ToolTip procedure.

- 6 Choose **Tools | Macros | Visual Basic Editor**.
- 7 Select **Message** from the **Procedure combo box**:



- 8 Enter the following code between **Private Function** and **End Function**:

```
UIButtonControl1.Message= "Zoom Out a set factor from the display"
```

- 9 Close the Visual Basic Editor (**File | Close and Return to ArcMap**)
- Hold the mouse over your new tool – the descriptive message will appear at the left of the status bar (i.e. at the bottom of the ArcMap window).

## 5 Using ArcObjects Developer Kit samples

There are many ArcObjects samples included with ArcGIS. To view these open the ArcObjects Developer Help and select Samples from the Contents menu. The samples are categorised and are either short tips, or more complex tools. Look at a few of the sample descriptions and notes in the Help.

Some of the samples require a .dll file to be registered which cannot be done on the Networked PC Service as the permissions required to do this are only available to administrators. The following example has therefore been chosen as it does not require .dll file registration.

In the ArcObjects Developer Help navigate to **Samples | ArcMap** and select 'Export Active View to JPEG'. Read the notes for this and follow the instructions to run the sample.

The tool samples are located in I:\licence\arcgis\arcexe83\ArcObjects Developer Kit\Samples\

## 6 Using scripts from ArcScripts

ArcScripts is a large collection of scripts for ArcGIS and other ESRI software held at [arcscripts.esri.com](http://arcscripts.esri.com). These have been written by developers and ESRI staff and are made available for general use (with appropriate licence conditions). Arcscripts is not the only online repository of ArcGIS scripts, but it provides easy access to many useful ones.

The example here uses a script to add X and Y coordinate positions of points to a shapefile's attribute table.

- 1 Go to **[www.esri.com/arcscripts](http://www.esri.com/arcscripts)**, enter **bizet** in **Keywords** and click **Search**.
- 2 A list of scripts submitted to ArcScripts by Grégory Bizet will be displayed. The one we need is titled **Add X and Y coordinates of features to Attribute Table**.
- 3 Click on the title for this script, read the information for the script and click **Download**
- 4 Click the **Accept** button to agree to the licence conditions and save the file in the folder **J:\arcobjects\**

The script is provided in a zipped Word document:

- 5 Unzip and open the document then select and **Copy** all of the contents.

In ArcMap, remove **cities.shp** and add **J:\arcobjects\hawaii\_cities\_shp** to the map. **hawaii\_cities.shp** contains only a few points, so the script to add X, Y coordinates will not take long to run. Make sure **Hawaii\_cities.shp** is highlighted in the Table of Contents.

An alternative to typing code straight into the VB Editor (as in the previous example) is to use a macro, as shown here:

- 1 In ArcMap choose **Tools | Macros | Macros**.
- 2 Enter a name **AddXY** for the macro and click **Create**.
- 3 This opens the Visual Basic Editor where a new module is created for this macro.
- 4 Delete the text already automatically added to the VB Editor and paste in the code you copied from the Word document.
- 5 Make sure you already have the point file hawaii\_cities.shp open in ArcMap with the title highlighted in the Table of Contents.
- 6 From the VB Editor menu select **Run | Run Sub/UserForm**.
- 7 This adds new X and Y fields to the attribute table of hawaii\_cities.shp and populates these with coordinate positions of the points. Open the attribute table and check that the X and Y fields have been added correctly at the end of the existing fields. This macro could be modified to include a Z value too if appropriate.

This script could also be attached to an icon, but there is no need to for this example.

Close the Visual Basic Editor before going on to the next section.

## 7 Using ArcObjects in ArcMap

VBA code can be used in some ArcMap dialog boxes, as demonstrated in the following two examples.

### 7.1 Using VBA code in the Field Calculator

The example here uses code to add X and Y coordinates to a file, as in the previous example, but using the Field Calculator and not the Visual Basic Editor. It requires more user input.

- 1 Open the attribute table for **hawaii\_cities.shp**.
- 2 Click the **Options** button and select **Add Field**.
- 3 Set **Name:** to **X**, **Type:** to **Float**, **Precision** to **6** and **Scale** to **2**.
- 4 Repeat steps 2 and 3, replacing **X** with **Y**.
- 5 Right-click the **X** field title and select **Calculate Values..**
- 6 Click **Yes** on the message that appears.
- 7 Tick the **Advanced** button in the **Field Calculator** dialog box.
- 8 Type the following code into the **Pre-Logic VBA Script Code** box:

```
Dim dbfPoint As Double
Dim pPoint as Ipoint
Set pPoint = [Shape]
dbfPoint = pPoint.X
```

- 9 Type **dblPoint** in the lowest box (**X =**)
- 10 Click **OK**.
- 11 The X field will be filled automatically with the X coordinate for each point.
- 12 Repeat steps 5 to 10, replacing X with Y.

## 7.2 Using VBA code in the Raster Calculator

This example uses VBA code in the Raster Calculator to calculate the slope of an elevation grid.

- 1 Remove **hawaii\_cities.shp** from the map.
- 2 Add the elevation Grid **I:\licence\arcgis\ArcTutor\Spatial\elelevation** to ArcMap.
- 3 Turn on Spatial Analyst under **Tools | Extensions**.
- 4 Choose the menu **View | Toolbars | Spatial Analyst** to display this toolbar.

To calculate a slope grid for **elevation** we can choose the menu **Spatial Analyst | Surface Analysis | Slope**. Alternatively, we can use ArcObjects methods in the Raster Calculator. To find out how to do this, open the Object Model overview poster for the **Spatial Analyst Extension**:

- 1 Open **Start | Programs | GIS | ArcGIS | ArcObjects Developer Help**.
- 2 Select **Object Model Diagrams** from the main pane.
- 3 Select **Spatial Analyst Extension** from the list of diagrams and choose to open the file.
- 4 Zoom into part of the diagram so that its text is readable.
- 5 In Adobe Acrobat choose **Edit | Find** and search for **slope**.
- 6 Zoom to the first instance. This displays the syntax for the slope method, and the relationship of this to the rest of the Spatial Analyst Object Model. This information can also be found in the ArcObjects Developer Help – enter **slope** in the **Index** tab to find the Slope method syntax.

Now use this syntax to calculate the slope:

- 7 In ArcMap select **Spatial Analyst | Raster Calculator**.
- 8 Based on the syntax **Slope (geoDataset, slopeType [,zFactor] )** enter **Slope ([elevation], 1)** and click **Evaluate**.
- 9 A new grid of slope will be created and displayed.

## **8 Where next?**

Look at the further information section at the end of this Guide for ideas on how to develop your ArcObjects knowledge.

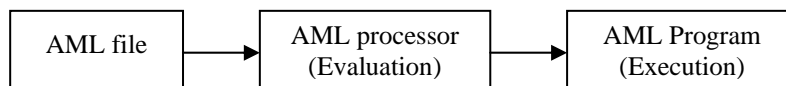
# Part 2

## 9 AML Introduction

AML, or Arc Macro Language, is the language that allows communication with the Workstation ArcInfo environment. It is a powerful, but slow, language that includes all the normal ArcInfo commands. It can be used in all Workstation modules (ArcPlot, ArcEdit, INFO, Tables, GRID etc).

It is used to build command macros (a sequence of ArcInfo commands) that are particularly useful for frequently used ArcInfo commands. It is also used to develop menu-driven applications for end users.

AML is an interpretative language where each line is interpreted by the AML processor before being executed by the relevant ARC program:



The AML processor evaluates variable substitution, logical branching and looping before moving on to the execution stage.

The next part of this Guide covers many of the aspects of AML, followed by a worked example incorporating the functionality introduced.

## 10 AML elements

There are three separate elements of AML that are used to create an AML program — directives, variables and functions:

- Directives — perform some action or determines flow of control
- Variables — perform text substitution
- Functions — perform more complex text substitution

Feature	Symbol	Example
Directive	&	&workspace
Variable	%	%count%
Function	[ ]	[response]

These three elements are described in more detail below.

### 10.1 Directives

Commands beginning with an ampersand (&) are AML directives. For example, **&type**, which displays text on the screen:

Arc: **&type Hello there**

#### Other directives:

**&workspace** sets the workspace.

**&system** temporarily exits ArcInfo to the system prompt.

**&terminal** sets the terminal type for menu's.

## 10.2 Variables

There are three types of variable — local, global and program.

A local variable is only known to the AML program in which it was created whereas global variables (prefixed with a period (.)) can be accessed by all AML programs. Program variables are created by ArcInfo and are read-only.

Variables in AML are set using the **&setvar** directive:

Arc: **&setvar count := 1** {local variable}

Arc: **&setvar .populate := 100** {global variable}

Variable references are identified by % signs:

Arc: **&setvar finalcount = %count%**

**&setvar** can be abbreviated to **&sv** or **&s**.

Individual variable values can be displayed using the directive **&type**, or all variable values can be listed using **&listvar**:

Arc: **&type %count%**

Arc: **&listvar**

A variable can be deleted using **&delvar**:

Arc: **&delvar variable1**

**Note:** Assignment operators can be written in several different ways:

Arc: **&setvar count := 1**

Arc: **&setvar count = 1**

Arc: **&setvar count 1**

## 10.3 Functions

Functions perform text substitution in a slightly more complex way than variables. They are enclosed by square brackets e.g. **[response]**. AML first evaluates the function in brackets, and returns the value of the function.

For example, the function **[response]** displays a prompt and expects an entry:

Arc: **&sv col = [response 'Enter a colour' red]**

The user will be prompted to enter a colour — if none is entered the variable **col** will default to red.

There are several '**get**' functions that display a list of choices in a menu. For example, **[getfile]** or **[getcover]**. These functions require the **&terminal** directive to be issued so menu graphics can be used:

```
Arc: &terminal 9999
```

```
Arc: &sv cover = [getcover]
```

This displays a list of coverage's from the current workspace in a menu system from which a coverage is selected using the mouse.

#### Other functions:

**[getchoice]** asks the user to choose from a menu of choices: e.g.

```
&sv col = [getchoice red green – 'Select a colour']
```

**[show]** displays the current value of a parameter: e.g.

```
&type [show &workspace]
```

**[exists]** returns whether a particular object exists: e.g.

```
&type [exists landcover –clean]
```

**.true** or **.false** will be returned depending upon whether or not the coverage **landcover** has been cleaned.

## 11 Creating and running an AML program

AML programs are text files that can be created with any text editor such as Pico. The file extension used is **aml**. AML's are generally portable across platforms.

For example, the AML program **simple.aml** could contain the following AML commands:

```
&setvar message := This is a simple program
```

```
&type %message%
```

```
&return
```

The AML will run until it encounters the **&return** directive when control returns to the previous input e.g the terminal.

To run the program use the **&run** directive:

```
Arc: &run simple.aml
```

The **&run** directive can be executed from:

- The command line
- An executable line in another AML program
- An AML menu selection

Comments can be inserted into an AML by prefixing them with **/\***:

```
/* simple.aml  
/* simple AML demonstrator  
/* created 8 March
```

As AML is an interpretative language it is advisable to enter comment statements at the end of the AML to speed program execution.

Long lines can be continued onto more than one line using the line continuation character **~**. For example:

```
&type This line is very very long and has to run over two ~  
lines
```

Blank lines in AML are ignored and can be used to break up and improve readability of the program.

### 11.1 Quoted strings

If text strings are used that include an AML reserved character e.g. **%** or **;** single quotes must be used in the statement:

```
&type 'The % of change'
```

If there is an apostrophe use single quotes and double apostrophe:

```
&type 'There''s no change'
```

## 12 Decision making

AML directives that control decision-making are known as 'flow of control' directives. They are used to control a programs order of execution.

For example, such directives can be used to check the value entered by a user:

```
&setvar choice = [response 'Enter a number less than 10']  
&if %choice% > 9 &then  
  &type Number entered is not less than 10  
&else  
  &type Number entered was %choice%
```

The **&do &end** directive can be used to include more than one statement in the conditional **&if &then** statement:

```
&if %cover% = roads &then  
  &do  
    build %cover% arcs  
  &type Line topology has been created for %cover%  
&end
```

Indenting commands on each line makes it easier to read the conditional statements.

## 13 Using loops

Loops are used when repeating an action.

Counted loops:

```
&do index = 1 &to 10 &by 1
  &type %index%
&end
```

A **&do &while** loop accepts any expression that evaluates to true or false:

```
&do &while [query 'Do you want to change coverage']
  &terminal 9999
  &setvar cover = [getcover]
&end
```

A **&do &until** loop repeats until the expression evaluates to true:

```
&do &until %finish%
  &type The loop is running
  &setvar finish = [query 'Do you want to end']
&end
```

## 14 Accessing files

To open and read a file use the **open** function:

```
&setvar temp_file := [open temp.txt openstatus -read]
&setvar line2 := [read %temp_file% readstat2]
&do &while %readstat2% ne 102
  &type Line reads: %line2%
  &setvar line2 := [read %temp_file% readstat2]
&end
```

{status variable 102 = end of file}

To create a file and write to it also use the **open** function:

```
&setvar write_file := [open check.txt openstatus2 -write]
&do index = 1 &to 10 &by 1
  &setvar writestat2 := [write %write_file% text]
&end
&setvar closestat := [close %write_file%]
```

A maximum of 20 files can be open at any one time, so it is important to close files when they are no longer needed :

```
&setvar closestat := [close %write_file%]
```

To close all open files at once:

```
&setvar closestat := [close -all]
```

## 15 Creating menus

Menus can easily be created using AML:

```
1 Sample Pulldown menu
Topology
Build BUILD PARCELS POLYGON
Clean CLEAN PARCELS PARCELS2 0.1 0.1
Quit
```

The **&terminal 9999** directive must be issued before running a menu. Menus have the file extension **.menu** and are run using the **&menu** directive.

There are several menu types available, and are referenced using the following numbers:

- 1 Pulldown
- 2 Sidebar
- 3 Matrix
- 4 Key
- 5 Tablet
- 6 Digitiser
- 7 Form

## 16 Creating AMLs automatically

The **&watch** directive is used to capture all user input and program output to a watch file. This file can then be modified and automatically converted to an AML file using the directive **&cwta**.

```
&watch test.wat {start capturing input}
```

```
&watch &off {finish capturing input}
```

```
&terminal 9999
```

```
&popup test.wat {display the watch file in a pop up window}
```

To convert the watch file to an AML file use the **&conv\_watch\_to\_aml** directive (abbreviated to **&cwta**):

```
&cwta test.wat test.aml
```

```
&run test.aml
```

## 17 De-bugging AML programs

### 17.1 Testing

An AML can be tested using the **&test** directive. This passes all input through the AML processor but not to the current AML program.

## 17.2 Suppressing message when running AML

ArcInfo AML messages can be turned off so that they are not visible in the terminal window:

```
&messages &off           {hides ARC messages}
&messages &off &all      {hides ARC & INFO messages}
```

Alternatively, all ArcInfo messages can be turned off:

```
&data arcinfo > /dev/null
```

## 18 Modular programming

AML's can be split into separate routines (modules), with each carrying out it's own task. Advantages include:

- Reusability of code
- easier to test AML

The **&call** directive is used to run a routine:

```
&do &while [query 'Do you want to build or clean the coverage']
&select [response '1 = clean 2 = build' 2]
&when 1
  &call cleancover
&when 2
  &call buildcover
&end
&end
&return

/*routines
&routine cleancover
  clean %cover% outcov
&return
&routine buildcover
  build %cover% poly
&return
```

## 19 AML help

Comprehensive Help is contained in the ArcInfo ArcDoc help system. Type **Help** at the Arc: prompt to start ArcDoc.

Command-line help is also available by typing **&commands** to get a list of all directives and functions. To get the syntax for these functions or directives type **&usage** followed by the name of the function or directive:

```
Arc: &usage response
```

## 20 Open Development Environment (ODE)

Workstation ArcInfo includes Open Development Environment (ODE) which allows programming languages other than AML to be used to build applications. On Unix, any development environment that supports X-Windows such as **Motif** or **Tcl/Tk** can be used. On NT, development

environments including **ActiveX**, **Visual Basic**, **Delphi** and **Visual C++** can be used. Information on writing customised applications can be found under **Customising ArcInfo** in the ArcDoc online help system.

## 21 An example AML application

This section goes through the stages of developing a complete AML. The first part of each sub-section includes a brief description of the particular task required of AML at that stage. Use this to write the relevant AML code alone using the online help, or alternatively follow the notes which work through each step of writing the AML code.

### 21.1 Step 1

Create an AML called **exercise.aml**, and prompt the user to select a polygon coverage from the appropriate directory:

On the Networked PC service:

```
&workspace t:\its\gis
&type [show &workspace]
&terminal 9999
&setvar cover = [getcover * -poly 'Select a coverage']
```

On UNIX:

```
&workspace /usr/local/courses/gis
&type [show &workspace]
&terminal 9999
&setvar cover = [getcover * -poly 'Select a coverage']
```

### 21.2 Step 2

Go into ArcPlot, give the user a choice of colours and use this to draw the polygons of the previously chosen coverage.

```
display 9999
arcplot
&setvar col = [getchoice red blue green - 'Select a draw colour']
mapex %cover%
linecolor %col%
clear
polygons %cover%
```

### 21.3 Step 3

Set up a loop where a choice is given to zoom into the centre of a coverage, or out from the centre of the coverage. The user should have the option to exit this which then ends the ArcPlot session.

Note: use the ArcPlot **windows** command to zoom.

```

&do &until %finish%
  &setvar zoomtype := [response 'Enter 1 to zoom into the
  centre, 2 to zoom out']
  &if %zoomtype% = 1 &then
    &do
      windows zoomin main
      clear
      polygons %cover%
    &end
  &if %zoomtype% = 2 &then
    &do
      windows zoomout main
      clear
      polygons %cover%
    &end
  &setvar finish = [query 'Do you want to end']
&end

```

#### 21.4 Step 4

Quit ArcPlot, enter Tables, select the PAT table for the landcover coverage, display the coverage items, list the whole file and exit tables informing the user that the end of the program has been reached.

```

q
tables
select landcover.pat
items
list
q
&type 'The end of the program has been reached'
&return

```

A complete listing of exercise.aml is given in Appendix B.

## 22 Creating AMLs from ArcToolbox

When using ArcToolbox in batch mode click on the **Save as AML** icon to save the requests as an AML. This can then be saved for future use. To use an existing AML in ArcToolbox select **My Tools | Run Geoprocessing AML**.

## 23 Further information

### 23.1 Books and manuals

The following manuals are available from the IT Service Desk:

#### ArcGIS:

*ESRI Exploring ArcObjects Vols. I & II*. These books are also available in digital Adobe pdf format. On the Networked PC Service choose **Start | Programs | GIS | ArcGIS | Digital Books** and look under the ArcObjects folder. On a standalone machine these documents are in the `..\arcgis\arcexe83\Documentation\Digital Books\ArcObjects` folder.

*ArcObjects developer's guide* (available from the Main Library).

#### **Workstation ArcInfo:**

*Understanding GIS — The ArcInfo Method*. Lesson 10 Customising ArcInfo introduces AML.

Other books available in the Main Library:

*ARC Macro Language: Developing ArcInfo Menus and Macros with AML*. 1997. ESRI.

*Inside ArcInfo 8*. 2000. Jay Flynn and Teresa Pitts.

*AML User's Guide*.

*VB & VBA in a nutshell : the language*. Sebastopol, CA : O'Reilly, 1998

### **23.2 Other sources of information**

A list of WWW links to GIS resources, and information on GIS discussion lists and newsgroups can be seen on the ITS GIS web pages at:

**[www.dur.ac.uk/its/software/gis/](http://www.dur.ac.uk/its/software/gis/)**

ESRI Virtual Campus at **[campus.esri.com](http://campus.esri.com)** provides online courses for ESRI software. For example:

- Introduction to Visual Basic 6

Those written by ESRI are free for staff and students at Durham – to obtain access codes for these, contact the IT Service Desk.

If further information or advice is required please contact the IT Service Desk (**[itservicedesk@durham.ac.uk](mailto:itservicedesk@durham.ac.uk)** or tel. 41515).

## Appendix A: AML Operators

### Arithmetic

+ addition

- subtraction

\* multiplication

/ division

\*\* exponentiation

LN logarithm

### Relational operators

= or EQ {equal to}

< or LT {less than}

> or GT {greater than}

<= or LE {less than or equal to}

>= or GE {greater than or equal to}

^= or NE {not equal to}

CN contains

NC does not contain

IN is contained in specified set

LK is like the specified character expression

**NB:** operators must have a blank space on either side of them.

## Appendix B: exercise.aml

```
/*set the workspace and display to screen
&workspace /usr/local/courses/gis
&type [show &workspace]

/*set the correct terminal display
&terminal 9999

/*ask user to choose coverage
&setvar cover = [getcover * -poly 'Select a coverage']

/*enter arcplot and ask user to choose draw colour
display 9999
arcplot
&setvar col = [getchoice red blue green - 'Select a draw colour']

/*set the mapextent to the chosen coverage and draw the polygons in the chosen colour
mapex %cover%
linecolor %col%
clear
polygons %cover%

/*set up a loop to zoom around coverage
&do &until %finish%
/*give user choice of zooming into coverage, or to extent
&setvar zoomtype := [response 'Enter 1 to zoom into centre, 2 to zoom out']

/*check to see which option user choose and carry out operation
&if %zoomtype% = 1 &then
&do
windows zoomin main
clear
polygons %cover%
&end
&if %zoomtype% = 2 &then
&do
windows zoomout main
clear
polygons %cover%
&end

/*check whether user wants to end zoom
&setvar finish = [query 'Do you want to end']
&end

/*quit arcplot and enter tables
q
tables

/*select the pat for the chosen coverage (landcover)
select landcover.pat

/*display the items and list the pat
items
list

/*quit info
q
```

```
&type 'The end of the program has been reached'  
&return
```

```
/* exercise.aml  
/*program demonstrating use of AML in ArcPlot and tables  
/*created 4 march
```