webAKT Tutorial 2
Creating and editing a simple knowledge base
Part 2: The tutorial itself

Accessing webAKT and creating a new knowledge base

Using a web browser, go to http://dev.webakt.org/.....

A large blank window appears with the main menubar in the top left-hand corner, as in Figure 1.

Figure 1. *The main menubar*

We will use the following menus:

File menu: This is concerned with creating a new knowledge base, and opening and saving an existing knowledge base

KB menu: Contains commands for working with a specific knowledge base, such as listing out the statements, sources etc.

Diagram: For viewing and working with diagrams reflecting networks of causal relationships.

Note for AKT5 users:
In AKT5, the commands for working with knowledge base files were in the same menu (the KB menu) as the commands for working with the currently selected knowledge base.

In webAKT, these 2 operations are separated into 2 menus: File and KB. This is in accordance with many applications: for example, a word processor uses the File menu for opening and saving documents, and the Edit menu for commands relating to editing the current document.

Creating a new knowledge base

For the moment, the only command we need from the File menu is New, to create a new knowledge base.

* Select **New** from the **File** menu

Creates a new, empty knowledge base file. You will be asked for the name of the knowledge base, but note: this is separate from the name of the file you eventually save the knowledge base in. (Hence, you can have several versions of the same knowledge base saved in file, and it's up to you to choose suitable names, perhaps numbering or dating them.)

This command displays a panel allowing you to put in various metadata relating to the knowledge base, e.g. author, location, etc. None of this is obligatory. This panel can also be accessed using the **KB** > **Metadata** command.

IMPORTANT NOTE

In AKT5, any change you made was added to the currently-loaded knowledge base when you click the Save button in a dialogue window – but it was up to you to save the knowledge base to file periodically to protect against loss of data in the event of e.g. a power cut.

In webaKT, every change you make and record using the Update window in any panel causes the whole knowledge base to be written to your browser's Local Storage (think of it like a big cookie) on your computer's hard disk. So you do not need to save periodically.

Moreover, when you return to webAKT the next day, this save copy is automatically loaded, so you do not need to worry if you forgot to save it to file the day before.

Entering metadata

* Select **Metadata** from the **KB** menu (if the Metadata panel is not already open).

This opens up a panel with the following text fields:

Project name
This is the working name used within a group for this project. Unlike AKT5, this has no formal correspndence with the saved file(s). It is editable, but it is up to you to ensure consistency with other versions of the knowledge base that have been or will be saved to file.

File name (read-only)
This is the name of the file that the knowledge base has been loaded from. It is read-only, because the file name is only relevant when you save to or load from a file.

Title of the knowledge base
A descriptive title.

Description
A more detailed description of the knowledge base (reasons, context etc).

Author(s) of knowledge base
Free text listing of the people who have contributed to creating and maintaining the knowledge base (subject to data protection restrictions).

The following are self-explanatory:
Acknowledgements
Associated documentation
Study area
Methods
Timing

* Enter text into the various edit fields. Note that none of this is obligatory.

* Click the **Update** button to update the knowledge base
Note that, throughout webAKT, no additions to the current knowledge base are stored unless you click the Update button. See Note 2.

Note 1: In AKT5, knowledge base metadata is entered into a memo, albeit a special one which, unlike all other memos, has various fields in it. In webAKT, the metadata component is treated as a top-level element, like the collection of statements or sources.

Note 2: In almost all computer programs or applications that allow you to save to a document to or load it from a file, the file name is the unique identifier for a particulat document, or a particular version of one. It is up to the person saving the file to give it a meaningful name. The same is true for webAKT.

Entering sources

In webAKT Tutorial 1, we looked at statements before sources, since statements are the primary form of information – that's why we create a knowledge base in the first place. However, when you are creating a knowledge base, it is good practice to enter sources before statements. This makes it easier to attach a source to each statement, and reduces the risk of having some unattributed statements. Unlike AKT5, it is not mandatory to attach a source to a statement, which makes it a good idea to develop the habit of setting up your sources first.

In the present context, we have just one source, the paper by Bandy et al, so we will enter that. Typically, most of your statements will come from local farmers, so you will have multiple sources.

* Select **Sources** from the **KB** menu.
This opens up the **sources** panel. Initially, the listbox listing the sources will be empty, but will display each source as you add it.

* Click on the **New** button.
This opens up the **source_details** panel.

* Click on the **Type** drop-down menu (currently displaying Person), and select **Reference.**
Note that the form fields change from ones appropriate for an interview to those appropriate for a published reference.

* Enter an **ID** for this reference.
This can be anything you want, limited to alphanumeric characters and the underscore character. It is good practice to make this related to the author(s) name(s) in the reference, so that you can recognise the source if you just see its ID (in a list of sources). In this case, **bandy** would be a suitable ID. In the unlikely event that you enter an ID that is already in use for a source, then webAKT will require you to choose another one.

> ***Important note for people migrating from AKT5:***
> *AKT5 automatically composed an ID for a source from a name, year and possible suffix (and a location if a Person (interview) source. This made for quite a long ID, with only the name really contributing to making the ID unique. The present approach is intended to simplify the process of providing an ID that is short yet distinguishable, and meaningful to the knowledge base creator.*

* Enter the relevant information for the other fields. In this case, it is:
name: Bandy,D.E., Garrity,D.P. and Sanchez,P.A.
title: The worldwide problem of slash-and-burn agriculture
publication: Agroforestry Today
volume: 3
number: 5
pages: 2-6

* Click on the **Update** button.
You will see the ID of the source listed in the **sources** listbox.

Warning! *You must press the* ***Update*** *button before closing this panel, otherwise the statement you have entered (or any other change you have made) will be lost, and you will have to type it again!*

Entering statements – the text-based method

We are now in a position to enter the statements we have obtained. In this tutorial, we will enter the statements that we have extracted from the Bandy *et al* reference.

The original AKT5 provided two methods for entering statements:

- the **text-based method**, in which statements are expressed directly in AKT's formal grammar, and
- the **diagramming method**, for causal statements, by drawing a link between two nodes on a diagram, where each node represents the cause or effect term for the statement.

webAKT supplements these with a third method – the **statement template method**. This allows statements to be entered using a template form, which simplifies the task by reducing the need for the user to memorise the various syntactic forms that a statement can take.

This section will take you through the text-based method. This is AKT's native method, since statements are expressed directly in AKT's formal syntax, and is the most generic, since any valid AKT formal statement can be expressed. The template method and the diagramming method are left till the end of this tutorial, so as to not interrupt the flow of learning about the other steps involved in creating a knowledge base. But if you do want to jump ahead and explore them, feel free, since they are simply alternative ways of entering statements.

1. Select **Statements** from the **KB** menu.
This opens up the **statements** panel. Initially, the listbox listing the statements will be empty, but will display each statement as you add it.

2. Click on the **New** button.
This opens up the **statement_details** panel.

3. Enter the formal version of the statement into the **Formal Language** box.

```
action(burning,site) causes1way att_value(pests,numbers,decrease)
```

Note the vertical bar on the right-hand side of this box. Initially, as you type, it is red. When the statement or partial statement is syntactically correct according to the AKT grammar, it changes to green. Note that, when you enter a conditional statement (with the first part of the statement followed by the word IF), it turns to green when you have entered the part before the word IF, then changes to red again until you have entered the complete statement. If it remains red when you have finished entering the statement, then you have made a mistake: check carefully, and correct it.

5. Click on the **Add source** button.

6. In the s**ources** panel that opens up, select the reference source you have previously added.
It should now have been added in the **Source** listbox in the **statement_details** panel.

7. When you are satisfied, click on the **Update** button.
This will update the statements stored in the loaded knowledge base. Look at the s**tatements** panel, and you will see that the list of statements in the listbox has been automatically updated.

Warning! You must press the **Update** button before closing this panel, otherwise the statement you have entered (or any other change you have made) will be lost, and you will have to type it again!

6. Repeat steps 2 to 5 for the other 15 statements in Table 2 in Part 1 of this tutorial.
Note1: If you are viewing this tutorial on a computer, you can simply copy-and-paste the formal version of the statement from the above table into the **Formal language** box in the **statement_detail**s form. Remember to press the **Update** button each time!
Note 2: The source you entered for the previous statement is still shown in the **Sources** listbox. This is based on the assumption that mostly the investigator creating a knowledge base will be entering multiple statements from the same source. If the source has changed, then simply select it in the listbox, and click on the **Remove source** button.

7. Save your knowledge base
Click in the **File** menu, and select the **Save as** command.
Note 1: For browser security reasons, a simple **Save** command is not available in web-based applications, so you have to use the **Save as** command. Click the **Save** button in the dialogue window when you have checked the directory and provided a suitable file name.
Note 2: The name of the file you choose to save the knowledge base in is the important name. The name you have given to the knowledge base when you first created it is an informal label. So, as with many other applications (like Word or Excel), you can save the same document, or some version of it, in various files. It is up to you to use a sensible naming convention and keep track of what is what. This makes it easy to maintain some sort of versioning system as you develop a particular knowledge base.

Editing and entering formal terms

Formal terms are the words you provide in statements which are specific to your context. These include the names of objects (e.g. maize), parts of object (e.g. leaf), processes (e.g. erosion), actions (e.g. harvesting), attributes (e.g. height), and values of attributes (e.g. high or 60_cm).

Your knowledge base maintains a glossary, or dictionary, of the formal terms you use in your statements. There are two main reasons for this. The first is so that you can provide extra information about a term, such as a list of synonyms, or a memo describing/defining it in more detail. The second is to help ensure that consistency – using the same term for the same concept throughout the knowledge base.

Terms are added to the formal terms glossary in two ways. One you have used already: when you enter a new statement (using any of the 3 methods mentioned above), webAKT extracts the various terms in the statement, analyses their type from their grammatical role in the statement, and adds them to the glossary behind the scenes. In this case the software knows nothing about the formal term apart from what it can infer from the statement (this includes its grammatical role, e.g. object, attribute), though you can elaborate the term afterwards. The second method is when you, the user, enter the term explicitly in the glossary. We will explore both methods in the following two sections.

Editing formal terms provided by statements

Re-load one of the knowledge bases you have created previously.

1. Open up the **KB** menu, then select the **Formal terms** command.
   Note that the formal terms listbox lists the formal terms extracted from the statements you entered.
   .
2. You can filter the terms for a particular type (e.g. object, attribute...) using the drop-down menu above the listbox. Try filtering by object.

3. Select a formal term from the list in the listbox.

4. Click on the **Edit** button.

5. Edit the formal term entry.
   Note that you cannot change its name or its type, but you can add a definition, synonym(s), and a memo. There is also space for displaying one or more images illustrating the term (particularly appropriate for objects), but that is not operational yet.

6. Click on the **Update** button.
   As usual, you must do this before closing the panel, otherwise our editting will be lost.

Adding a new formal term

You may wish to add formal terms before you use them in statements. A reason (in fact, a very good reason) for doing this is to establish a glossary of terms and their synonyms in advance, to help establish a common vocabulary in advance, as a way of standardising the terms used in statements before you actually enter the statements.

1. Select the **Formal terms** command from the KB menu.

2. The **Formal terms** panel should open up.

3. Click on the **New** button.

4. Provide a **Name** and select a **Type** for the new formal term.

5. Complete the rest of the form, as before, providing a definition, synonym(s) and possibly an explanatory memo, as appropriate.

6. Click on the **Update** button.
   As usual, you must do this before closing the panel, otherwise your editing will be lost.

7. Check that the formal term now appears in the listbox in the **formal_terms** panel

Creating and editing object hierarchies

Hierarchies in AKT provide a rich way of grouping things in AKT. The "things" might be objects, as presented in this section, or topics, presented later: the same principles apply to both. Using hierarchies allows the user of a knowledge base to explore statements that are related in some way – by the objects they contain or the topics they relate to.

A hierarchy provides a way for grouping related terms. For example, the Atwima knowledge base contains the following object hierarchies: *crop, fertility indicator, land types, soil, trees and weeds*. The first 4 of these are "flat" - the hierarchy only has one level, being a simple list of objects within that grouping. The last 2 are nested: for example, the *weeds* hierarchy contains an entry for *mwura bone* ("a general name for a bad weed"), which in turn contains the names of various weed species. The hierarchy enables hierarchical reasoning when exploring the knowledge base: for example, searching for statements referencing the object *mwura bone* retrieves all statements referring to its sub-objects; and searching for statements referencing the object *weeds* retrieves all statements referring to any of the weeds named in the hierarchy, at any level.

One object (i.e. object-type formal term) can appear in more than one hierarchy, but can appear only once in a particular hierarchy.

AkT5 provides two ways for displaying a particular hierarchy. One is using a form to show the items above and below a particular item. This gives a blinkered view of the hierarchy, since one can focus on just one item at a time. The other is as a diagram, with the hierarchy arranged as an inverted tree, with the hierarchy name at the top of the tree. This shows the whole hierarchy, but is rather spread out and thus unsuited for big hierarchies, and, being static, can only be used for displaying the hierarchy: any changes to it must be made using the form-type interface.

webAKT, in contrast, provides a single mechanism for displaying and editing hierarchies, as you have seen in Tutorial 1. It is based on a collapsible tree, similar to the interface that operating systems provide for browsing file directories. Instead of the tree being arranged top-down, it is arranged sideways, with items at a particular level being indented to the right underneath the item that contains them. Each level can be collapsed or expanded with a simple click on its containing item.

In addition to using this design for displaying object and topic hierarchies, webAKT has extended this approach to allow an item to be selected, for editing its details, and even to allow changes (additions or removals) to the hierarchy. In other words, you can edit the hierarchy itself using the same interface.

In this section, we will create a single hierarchy, with just one level below the top, and two objects in it. That should be enough to familiarise you with the technique. The same statements you entered earlier contain a reference to the object crop. We will construct a hierarchy, called Crops, with crop as the top-level object, and two subobjects: maize and wheat.

> *Typographical convention:*
> *In order to visually distinguish between formal term objects and the hierarchies they can belong to, I have adopted the following convention:*
> *. formal terms are always in lower case, e.g. maize, wheat, crop*

. hierarchies are always fully capitalised, e.g. CROPS
This is merely a convention adopted in this tutorial, but you may fund that it'sa useful convention to adopt to avoid confusion.

1. Select the **Formal terms** command in the **KB** menu, and create formal terms for **crop**, **maize** and **wheat**.

2. … using the methods you used in the previous section. **crop** may exist already if you are following on from entering the sample statements.

3. Select the **Object hierarchies** command in the **KB** menu.

4. You see a panel with no hierarchies.

5. Click on the **New** button.

6. You see an empty **Hierarchy details** panel.
7. Enter **CROPS** in the Hierarchy name field.

8. Enter **crop** in the **Root name node** field.

9. Note that, unlike AKT5, the name of the hierarchy is not the same as the name of the top-level object. (It could be the same, but that would be confusing as they are different concepts.)

10. Click the green tick.

11. webAKT creates an empty hierarchy, with **crop** as the top-level object.

12. Select the **crop** entry in the **formal terms** listbox in the **Formal terms** panel, by clicking on it.

13. Click the **Append** button.

14. webAKT now expects you to select an object-type formal term from the list in the **Formal terms** panel.

15. Click on **maize**.

16. Maize should be added to the **Crops** hierarchy under the **crop** entry. If you don't see it, click on the **crop** entry to allow it to show its subobjects.

17. Add **maize** in the same way.

You have now created a simple object hierarchy! Click on crop, then repeat, and note how the subobjects are displayed, then hidden.

Creating and editing topics

In Tutorial 1 you came across the AKT concept of a “topic” - basically, a method for grouping statements into a set that have something in common. The “something in common” is defined by an expression which contains the formal terms that a statement needs to contain in order for that statement to be part of the set. This expression is a Boolean expression, i.e. it consists of formal terms joined by the Boolean operators “and’, “or” and “not”.

For example, a topic, say “crops”, defined by the expression “maize or wheat” will find all the statements that contain either the term “maize” or the term “wheat” somewhere in the statement.

You might think that the way to create a topic is to select Topics in the KB menu: that’s what we have done for the other components of the knowledge base. That will indeed be the path to follow in due course. Currently, however, a topic is created by using the Boolean search procedure (which you saw in Tutorial 1), and then making a topic out of the search expression.

1. Select the **Boolean search** command in the **KB** menu.

2. Formulate a Boolean search expression, as you did in Tutorial 1.
   For example, if you want to create a topic, called let us say **crops**, grouping all statements that refer to either wheat or maize, then:
   … Click on **wheat** in the listbox, and click the **Select** button;
   … Click the or button;
   … Click on **maize** in the listbox, and click the **Select** button.
   You should see **maize or wheat** in the Boolean search string box.

3. Click on the **Create** topic button.
   A topic-details panel will appear, with the search expression entered in it.

4. Enter name for the topic in the **Topic** field: say, **crops**.

5. Enter any explanatory text in the **Description** box, if you want to.

6. Click on the **Update KB** button.

7. Click the **Topics** command under the **KB** menu...
   … and you should see the topic **crops** listed in the **Topics** listbox.

And that’s it: you have created a topic.

Creating and editing topic hierarchies

The section on Creating and Editing Object Hierarchies set out the basic ideas for creating and editing hierarchies in webAKT, in the context of object-type formal terms. AKT5 also caters for hierarchies of topics, and that has been carried through to webAKT. In fact, the display of, and method for working with, topic hierarchies is identical to that used for object hierarchies.

The following exercise creates a topic hierarchy called "agriculture". This will have two topics: the "cereal" topic created in the previous section, and a new topic, called "livestock". This is a somewhat artificial exercise, since the knowledge base does not contain statements relevant to this topic, but that simply means we cannot retrieve actually statements using the topic hierarchy: that does not affect the value of this exercise in showing you how to construct a topic hierarchy.

1. First, re-do the previous exercise, to create a topic called **Crops** if this is no longer available.

2. Then, use the same procedure to create a topic called **Livestock**, defined by the Boolean search expression "**cattle or sheep**".

3. Now create a topic called **Agriculture**.
   Click on Boolean search, as you did for the other two topics Crops and Livestock, but this time do not enter a Boolean search expression for this. This topic is defined by its subtopics underneath it.

4. Now, click on the command **Topic hierarchies** in the **KB** menu.
   This opens up a panel with an empty listbox: you do not have any topic hierarchies just yet.

5. Click on the **New** button.

6. Enter a name for the hierarchy in the **Hierarchy name** edit field.
   It's good practice to make this different from the name of the top-level node in the hierarchy, so call it **AGRICULTURE**, using the same convention that we used for object hierarchies.

7. Enter the topic name (**agriculture**) for the top-level node of the hierarchy **AGRICULTURE** in the edit field.

8. Click on the **green tick button**.
   The (as yet empty) hierarchy will appear.

9. Select the **agriculture** line in the list of topics in the **topics** panel's listbox.

10. Click on the **Append** button.

11. Click on **cereals** in the listbox in the **topics** panel.
    It should appear below agriculture in the **topic_hierarchy_details** hierarchy tree.

12. Repeat steps 9-11 for livestock in the Topics panel.

13. Click on the Update button in the Topic hierarchies panel.

And that's it! You have now created a topic hierarchy. Check that it's now listed in the Topic hierarchies panel.

Entering statements: the template method

An alternative to entering statements in text form is to create them by filling in a simple template form. The value of this may not be immediately obvious after the previous exercise, where statements were provided on a plate, but in real life entering statements requires a good understanding of AKT's grammar. The idea behind the statements template approach is to help the user enter syntactically-correct statements.

A number of templates are available, corresponding to the various types of statement defined by the AKT grammar. In order to guide the user to the correct template, the user selects options depending on the statement type (e.g. att_value or causal and, for att_value: object, process or action). The user then clicks on each field in the form, and then either selects a formal term from a list of all known terms of that grammatical type, or enters a new formal term.

We will begin with the first statement:
`action(burning,site) causes1way att_value(pests,numbers,decrease)`
then you will enter the remaining 15 statements.

1. Create a new knowledge base: **File** menu, **New** command, giving it a new name
The reason for doing this is to avoid duplicating statements in the previous knowledge base.

2. Select **Statements** from the **KB** menu.
This opens up the **statements** panel. Initially, the listbox listing the statements will be empty, but will display each statement as you add it.

3. Click on the **New** button.
This opens up the **statement_details** panel.

4. Click on the **Template** button.

5. Click on the **Causal** radio button.
… because we are going to enter the first of the 16 statements, which is a causal statement.
Note that 4 options are presented for both the Cause and the Effect parts of a causal statement.

6. Click on the **an action** option under the **Cause** heading.

7. Enter **burning** in the **Action** field.

8. Enter **site** in the **Object1** field.

9. Click on the **an attribute** option under the **Effect** heading.
Three more options now appear: object, process and action.

10. Click on the **Object** option.

11. Enter **Pests** in the **Object** field

12. Enter **Numbers** in the **Attribute** field

13. Enter **decrease** in the **Value** field.
This completes the entering of the statement using the template.

14. Click on the **OK** button.
The statement is now entered in the **Formal language** field.

15. Add the source, as you did in the first exercise.

16. Click on the **Update** button.
… and the statement appears in the **Statements** listbox.

17. Repeat for the other 15 statements.

Comment: This exercise has involved many more steps than the first one. This is because you are being led through the various options available, rather than having to remember a fairly complex grammar. At the end of the day, each investigator can choose whatever they find easier to work with.

Entering causal statements: the diagramming method

webAKT, like AKT5 before it, provides a way of visualising causal statements. This is called the diagramming interface, and you have already encountered it in Tutorial 1. If not, please go back and follow through that tutorial, as it will make the following much clearer.

The nodes in the diagram represent the two parts of a causal statement (the cause and the effect respectively). The link ("arc") between 2 nodes roughly corresponds to the word "causes". So the statement "rainfall causes soil erosion" is represented as an arc between the two nodes "rainfall' and "soil erosion". Multiple causal statements can form a network: for example, soil erosion can in turn "cause" a reduction in crop yield.

> *This is actually a simplification. Many causal statements include a reference to the value of an attribute of an object, for the cause and/or the effect parts, but the nodes for this causal relationship in the diagram are expressed just in terms of the attribute of an object. So one causal arc in the diagram may cover several statements. This is addressed in detail in the diagramming section of the webAKT Manual.*

The webAKT diagramming interface has two roles. The first is to provide a way of visualising the causal statements that exist in the knowledge base. You have seen how to do this in the first webAKT tutorial. The second is as a way for entering causal statements, as an alternative to entering them as text in AKT's formal language. This section shows how to do that.

As an example, we will enter the statements abstracted from the first paragraph of the text:

> "....burning helps to control pests and diseases ... the higher soil temperatures that follow clearing and burning also accelerate the decomposition of organic matter in the top layers of the soil."

The following five statements might be abstracted from this paragraph:

1) Burning causes a decrease in pest numbers
2) Burning causes a decrease in disease levels
3) Clearing causes an increase in soil temperature
4) Burning causes an increase in soil temperature
5) An increase in soil temperature causes an increase in the rate of decomposition of organic matter

'Burning' can be either a process or an action. If it is used as a management tool, it is an action, otherwise it is a process. In any knowledge base it can only be used as one or the other, it cannot be entered as both action and process. If some statements also refer to burning brought about by natural causes then another term must be found, either 'fire' as an object, or 'conflagration' as a process.

Each of these statements can be represented in the diagramming interface as a pair of nodes and a link between them. In the case of the first statement the two nodes are 'burning' – an action – and 'number of pests' – an attribute (number) of an object (pests). To represent this statement, do the following:

Start a new knowledge base.
We do this to avoid entering duplicate statements into the same knowledge base you worked with so far.

1. Under the **File** menu, select the command **New**, giving the knowledge base a new name.
2. In the small popup dialog window, enter a name for the knowledge base.
3. Click the OK button.

Create a "site burning" node

1. Click the **Action** button from the **Add node** box on the left-hand side of the diagram screen.
2. Move your cursor onto the drawing area and double-click where you want the node to go. A dialog box will appear, asking for details of the action.
3. Enter the name **burning** in the **Action** field.
4. Enter the word **site** in the **Object1** field.
   Actions may refer to one or two objects (Object1 and Object2 in the dialog). These are like the subject and object of the action verb. Both are optional, but it is useful here to add in a word such as site to provide a context for the action.
5. Click the **Accept** button (to accept the two entries you have made in the form).
6. Click the **OK** button to close the dialog. Note that an Action node has now appeared in your diagram.

Create a "pests number" node

1. Click the **Attribute** button from the **Add node** box on the left-hand side of the diagram screen.
2. Move your cursor onto the drawing area and double-click where you want the node to go. A dialog box will appear, asking for details of the attribute.
3. Choose the option which says that "The node is an attribute of … an object".
4. Enter the word **pests** in the **Object** field.
5. Enter the word **numbers** in the **Attribute** field.
6. Click the **Accept** button (to accept the two entries you have made in the form).
7. Click the **OK** button to close the dialog. Note that an Attribute node has now appeared in your diagram.

Link the two nodes

1. Click the **Causes1way** button from the **Add link** box on the left-hand side of the diagram window.
2. Move the mouse pointer to the **site burning** node, press and hold the mouse button, and drag the cursor to the **pests numbe**r node.

Enter details for the causal arc

1. Double-click on the arc you have just drawn.
2. A dialog panel appears. Because one of the nodes (the target, i.e. the effect, node) is an attribute node, you can enter the value for the attribute **number**. In this case, the value is **decrease**, so enter this in the first field.

Enter the remaining 4 causal statements

1. Now enter the remaining 4 causal arcs, following the same procedures. Note that **clearing**, like burning, is an action and therefore needs an object. In this case as well, 'site' would be a valid object.
2. Look at the statements panel.
   You should see the 5 causal statements listed.

Adding images to the knowledge base, and linking them to knowledge base items

In AKT5, adding images was convoluted, and subject to severe restrictions. In addition, the only image type supported was the bitmap, now an obsolete format, at least in web terms.

webAKT provides a far simpler method for including images in the knowledge base, allows for any image format to be used. It also allows one image to be used multiple times, and allows any knowledge base item to be linked to multple images.

The key change is that an image is included in the knowedge base not as the image itself, but as a link to an image that can be anywhere on the web, or on the user's own computer. Thisinvolves the following workflow:

The knowledge-base creator:

- either finds a suitable image on the web or takes a photo and uploads it to an image-sharing site (such as flickr) or own computer;

- gets a URL for the image (by simply right-clicking over the image in a web browser);

- adds the URL into a new-image dialog panel;

- adds an identifier for the image in (for example) the edit-a-formal-term panel.

Tutorial instructions for doing this will be forthcoming, but the procedure has been implemented.