xxxInformationTracking.xml file

The **xxxInformationTracking.xml** file is an optional file that can be included in a Language Package. The file is used to reduce the number of incorrect links that may be detected when you run the CAST Universal Analyzer on your newly defined language. I.e. it allows you to refine the link search scope.

Technical background

Currently, the link creation mechanism in the Universal Analyzer is based on the link itself - for each link, we need to define the caller or callee status of each object involved (see **Parameterization of links between objects** in Defining a new language). This definition enables you to restrict the number of caller and callee objects for each link, but it does not allow you to define where to search for the called object when a link is detected.

As a result, in order to specify the search location, information needs to be added to the **xxxInformationTracking.xml** file. There are two types of information to add, if you want to restrict the search scope:

Information representing the search scope

This information indicates the "top" of the scope area in which the callee will be searched for, i.e. a metamodel type. For example, to create links only with objects in the same file, it is possible to indicate - in the search scope – the metamodel type that represents the file, i.e. **sourceFile**.

In addition, if you wanted to search for links with variables that are only defined within a class, you simply need to define the class as the search scope.

If the search scope information is not defined, by default, the search scope will be the entire analysis, i.e. the job.

Information describing a transition used to move between objects in the scope

This information takes the form of a link type (for example, the inclusion link to move between files). It is also possible to qualify this transition with the following information:

- **Transition mode**: either *direct* or *recursive* (default setting). This is used to determine whether the analyzer should follow transition links or not.
- **Priorities for creating links**: either *first* or *all* (default setting). This is used to determine which links will actually be created. For example, if you choose *first* and a link is detected in the scope, then the link will be created and no further searches will be carried out in the inclusions. If, on the other hand, you choose *all*, then the link would be created AND a search would be made in the transitions.

If this information is not specified, but a scope has been defined, then, by default, a search will only be made in the scope.

If this information is specified, but no scope has been defined, then the information is declared useless and will be ignored. This is because, as already mentioned, if no search scope has been defined, then the entire analysis (job) will be searched instead. As a result, scope information is indispensable if the transition information is to make any sense.

Further details

The two pieces of information outlined above must be provided for each calling object, i.e. the information must specify where the search for the called object is to be made when a link of such and such a type is detected in the calling object.

In addition, and still with the aim of refining the link search, CAST provides the means to specify a link search in **character strings**. This option will be global to the entire language type and will be activated by default.

Finally, there is now only one way to specify that you want to carry out a generic link search in the metamodel: create inheritance from CallerLinkable (and CalleeLinkable if you want the object to be called by generic links). Note that if CallerLinkable or CalleeLinkable are not defined, typed links will still be searched for (i.e. links defined in the LanguagePattern and Metamodel files).

Example xxxInformationTracking.xml file

The following is an example xxxInformationTracking file:

```
< InformationTracking id="PHP">
<searchLinksInString>false</searchLinksInString>
<phpClass>
<callLink>
<scope>sourceFile</scope>
<transition>
<linkType>includeLink</linkType>
<method>recursive</method>
<priority>first</priority>
</transition>
</callLink>
</phpClass>
</InformationTracking>
```

This tracking file would have the following effect:

- Character strings would be omitted from the link search
- When a call type link is detected in a phpClass, the callee object is searched for as a priority in the same file. If the callee object is not found in the same file, then the search will move into any directly included files. If the callee object is not found there, then a search is carried out in recursively included files.

Please note that you must validate your **xxxInformationTracking.xml** file using the **CAST Universal Assistant** (*UAAssistant.exe*)-- CAST end user application. Online help is provided in the application.

Deactivating the link search for sourceFile types

If you need to deactivate a link search (whether generic links, ESQL links or typed links) for sourceFile types **ONLY**, then you can now use the **xxxInformationTracking.xml** file to do so. In the past this was not possible because the sourceFile type is also present in the CAST CoreMetaModel.

By default, existing behaviour is retained, i.e.: the sourceFile type is CallerLinkable and CalleeLinkable for all link types (generics and typed links) and the search for ESQL links is activated. To deactivate a link search for sourceFile types only, you need to add a tag called ***sourceFileLinkSearchDeactivation*** and then specify what you want to do as outlined below:

For a Caller object

In the ***caller*** tag you can indicate the following:

- An **<ESQL/>** tag = this will deactivate the search for ESQL links and is equivalent to removing **<inheritedCategory name = "ESQLSearchable"/>** from the xxxMetaModel file.
- A **<genericLinks/>** tag = this will deactivate the search for generic links and is equivalent to removing **<inheritedCategory name = "CallerLinkable"/>** from the xxxMetaModel file.
- A list of links that must not be searched for as Caller. Each element is written in a **<link>** tag. For example, to deactivate the search for inherit type links, you need to include: **<link>inheritLink</link>**.

For a Callee object

In the ***callee*** tag you can indicate the following:

- A **<genericLinks/>** tag = this will signify that the file cannot be called by a generic link and is equivalent to removing **<inheritedCategory name = "CalleeLinkable"/>** from the xxxMetaModel file.
- A list of links for which the file in question cannot be a callee. Each element is written in a **<link>** tag. For example, to deactivate the search for inherit type links, you need to include: **<link>inheritLink</link>**.

Example xxxInformationTracking.xml file

The following is an example xxxInformationTracking file:

```
<InformationTracking id="PHP">
<sourceFileLinkSearchDeactivation>
<caller>
<genericLinks/>
<link>inheritLink</link>
<link>callLink</link>
</caller>
<callee>
<link>inheritLink</link>
</callee>
</sourceFileLinkSearchDeactivation>
</InformationTracking>
```

This tracking file would have the following effect:

- The search for generic links (as a caller) would not be carried out on the files.
- Inherit and Call links will not be searched for in the files
- The files cannot be called by an inherit link

Limitations

One current limitation exists for the xxxInformationTracking.xml file:

For example, it is possible that to find the Callee of a link type A you would need to search through the transitions of a link type B, which would naturally mean that the B type links would have to be already resolved. However, if, in order to resolve the B type links you had to search through the transitions of the A type links, then this would create a blockage because you would need to "do A before B AND B before A".

As a result, when this type of situation is encountered, a message is displayed in the **CAST Universal Analyzer Assistant** (when the file is verified) and also when a Universal Analyzer job that includes the information tracking file is executed. The message explains that the current configuration could lead to data incoherence. The links involved will also be listed.