Subject: Re: IDL routines dependencies map maker
Posted by DavidPS on Mon, 12 Mar 2012 23:48:40 GMT
View Forum Message <> Reply to Message

> In general, 100% accuracy in static analysis like this is going to be
> impossible for a dynamic language like IDL. But, a lot of useful
> information can still be derived from the code.

Really useful if the map.dot file is opened with gephi. You can change the size of the circles, and get some statistics straight away.

> It is impossible to handle the general case for objects. A routine can
> call a method on an object without knowing, until runtime, what class
> and method they are actually calling.

Ok, so then I completely forget about objects ;-)

> Also, CALL_PROCEDURE, CALL_FUNCTION, CALL_METHOD, and EXECUTE also make
> the general case impossible to deal with. I would just ignore these cases.

I'm not sure in the CALL_ ones, but I think my scripts will find the function being called from EXECUTE. Not the procedures because I've set to look at the starting of the line.

> Another problem case is determining the difference between function
> calls and array indexing (IDL has a difficult time with this too, which
> is why the "compile_opt strictarr" option is so useful).

In the way it works, it find first the definition of the functions, and then search if they are called. So, yes, it could be a "false" edge if there's a variable called with the same name than the function. I was not aware of that "compile_opt"... I still don't know how I would use it.. but at least I've always used () just for functions.

> IDLdoc has a :Uses: field which could be used to pass hints to the
> dependency finding algorithm in the difficult cases, e.g., when using
> EXECUTE you could just specify what routine you are calling in the
> :Uses: field.

Then everything would be a lot easier!! ;-)

David