Subject: A couple of requests to help importing source libraries
Posted by mdelfede on Tue, 10 Jan 2012 23:51:02 GMT
View Forum Message <> Reply to Message

I have an huge library (OpenCascade, as usual...) which has a quite weird folder structure and build process.
I'd like to have it imported into some Upp packages,but WITHOUT touching at original source tree, because I'd like to submit patches for it and doing it on a modified source tree is cumbersome and time consuming.
So, the problem : OCC (or better, OCE, the community edition) structure is the following (in short) :

OCE
  inc containing some thousands of include files
  src containing package separated hand written source files
  drv containing packages separated auto generated src files

besides the inc folder, which contains tons of include without subfolder, the tree is arranged in a weird way; for instance, even if the library is divided in

MODULES
  |
    TOOLKITS
      |
        PACKAGES

the structure of src folder, for example is

src/PACKAGENAME/sourcefiles

My solution would be to add an 'uppoce' subfolder inside OCE main folder, with just a tree of upp packages referencing the real sources; en example .upp file :

noblitz;

options
 -I../../../../inc,
 -I../../../../src/Dynamic,
 -I../../../../drv/Dynamic;
file
 ../../../../src/Dynamic/Dynamic_AbstractVariableInstance.cxx,
 ../../../../src/Dynamic/Dynamic_BooleanParameter.cxx,
.....

Problem n.1 : upp don't allow ../../ paths in .upp files. Well, it allows them but don't compile, just preprocessing and 'show assembly' works. In Linux I can solve with symlinks of source files in package folder, but it's not portable to windows; windows has some kind of symlinks, but they're hard to use or works only on Vista+ OS.
As I'd like to ask maintainers of OCE to include an UPPOCE folder, the only viable solution would be the one I put in sample .upp file above.

Problem n.2 : I could 'translate' OCE library using a relationship of :
PACKAGE---> PACKAGE; that would bring some 2-300 upp packages...
TOOLKIT---> PACKAGE; that would bring some 56 upp packages
MODULE ---> PACKAGE, that would bring just 4-5 upp packages, but could arise in inter-dependencies problems.

The best solution would be the first, then grouping with 'super upp packages' up to module level; so :
about 300 upp packages for OCE packages
plus about 56 upp packages for OCE toolkits, grouping the former 300
plus 5-6 upp packages for modules, grouping all above.

The problem is that when opening the 'add package' dialog you'd be prompted with some 400 packages coming from OCE, mixed with upp and own ones, which is unmanageable.
The nicest solution would be to add an 'HIDE SUBPACKAGES' checkbox on upp, or add an 'hidden flag' on .upp packages and a flag "show/hide hidden packages" inside theide, which, btw, could be useful to reduce the number of presented packages.

Resuming, I'd like to have :

1) Relative paths in .upp files ("../../somefolder/somefile")
2) A way to hide on demand some packages from theide

is all that feasible ?

Max

Subject: Re: A couple of requests to help importing source libraries
Posted by dolik.rce on Wed, 11 Jan 2012 06:41:06 GMT
View Forum Message <> Reply to Message

Hi Max

Not an easy task, but probably doable. As long as there is some regular structure, you can write some scripts to create the .upp files for you.

mdelfede wrote on Wed, 11 January 2012 00:511) Relative paths in .upp files ("../../somefolder/somefile")
2) A way to hide on demand some packages from theide

Number 1 should be easy to implement and also helpful in other situations. I remember hitting that problem in the past

For number 2 at least 2 solutions exists, but none of them is really clean... One is to mark some packages main, so you could use "All main packages" or "All main packages of the first nest" to filter what you want to see. Second option is to organize the packages into two directories, hidden and visible, and specify the visible as first nest, so you can use "All packages of first nest" to filter the hidden ones out. I think the second is slightly cleaner, but still just an ugly workaround.

Best regards,
Honza

Subject: Re: A couple of requests to help importing source libraries
Posted by mirek on Wed, 11 Jan 2012 07:09:42 GMT
View Forum Message <> Reply to Message

Perhaps the solution of many of listed problem would be some sort of 'symlink' in upp?

like

nest/opencascade1.upp:

.....
include opencascade

nest/opencascade:
... here goes that terrible directory structure ...

IMO: From what I understand, the issue is that you want to create several packages for single U++ unfriendly structured tree...

Subject: Re: A couple of requests to help importing source libraries
Posted by mdelfede on Wed, 11 Jan 2012 08:31:23 GMT
View Forum Message <> Reply to Message

Hi

@Dolik : the script is already mostly ready, that's not a problem. I've already packaged OCC once, but with a script that moved viles around and pacthed them to "adapt" to Upp structure (and to clean a bit the arrangements...).
But, that one makes *very* difficult to submit patches for it, as I have to diff my patched files against a clean set packaged for Upp, and then modify the patches to fit the original tree... too time consuming, really.

Also building the libs once is not an option, as I need to have them manageable with Upp at source level to be able to debug/patch them.

@Mirek : yes, almost. I'm looking for a "general" solution that could allow, for example, pushing to Bazaar *only* the packages without the main tree (which is some 30-40 MB) and make it easy to fetch original lib sources from their site.
With my way, I could provide a small upp packages tree with almost just .upp files, insert it into lib's tree and have a set of upp packages ready to use. Simple and clean, and without having to touch a single line of external libraries.

Having upp-part INSIDE the lib's directory structure would also allow to have it included in official lib distribution, which I guess could be accepted into OCE.

On Linux I can do without ../../.. paths, but I need symlinks which are not portable to windows (and clobbers the packages folders). By now I'm doing it like this and it works.
Maybe a cleaner solution would be something like that :

```
symlinks
    src ../../../src,
    drv ../../../drv;
files
    src/somefile.cxx,
    drv/someother.cxx;
```

The other problem (the big package set displayed in package manager) could also be solved in a complicated but cleaner way having some structured package tree in theide :

```
OCC
  -FoundationClasses
      -aToolkit
          aPackage
  -anotherModule
.....       -
```

Max