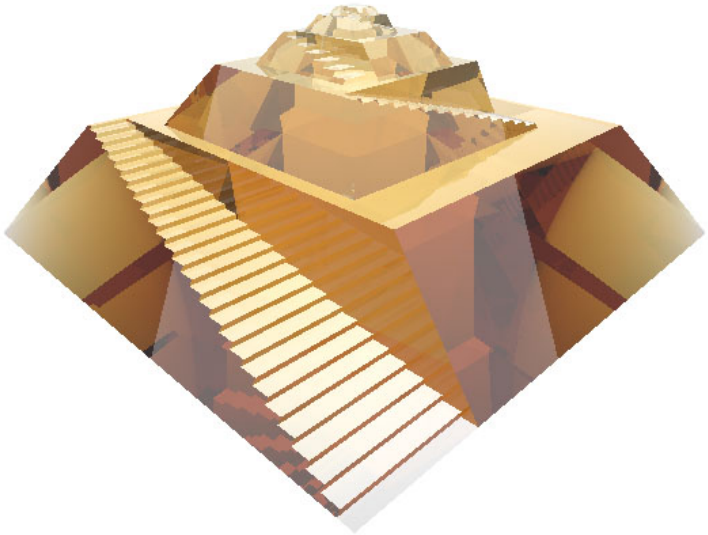
BABEL v. 3.8
Installation Manual

Hyperion Configuration

Language: C++/STL
Post-processing: Visual C++ 6.0
Execution Platforms: Ms-Widnows NT+
Communication Platforms: ACE 5.1 + TAO 1.1



by Cipriano Galindo
ver. 1.0 May 2004

System Engineering and Automation Dpt.
University of Malaga
http://babel.isa.uma.es/babel

Content

Introduction

This document aims to guide developers to the correct installation and setup of the BABEL v.3.8 development framework for the Hyperion generator (C++, VC6.0, Win32, ACE5.1+TAO1.1). The software you need before starting the installation is:

- The ACE+TAO distribution of CORBA (version ACE 5.1 TAO 1.1) that you can download at http://deuce.doc.wustl.edu/old_distribution/ACE-5.1+TAO-1.1.tar.gz for free.
- The BABEL software, that you can obtain at http://babel.isa.uma.es/babel/
- Visual C++ (or any other compiler)

Next sections detail the steps to install this BABEL configuration into your system.

CORBA Installation

The ACE+TAO distribution of CORBA is installed by unzipping the downloaded file into a folder whose name does not contain spaces, for example, C:\CORBA, and adding the following environment variables to the system:

ACE_ROOT=<ACE+TAO Installation folder>
TAO_ROOT=< ACE+TAO Installation folder\TAO>

Thus in the example, the added environment variables would be ACE_ROOT=C:\CORBA and TAO_ROOT=C:\CORBA\TAO.

A set of additional entrances in the PATH are needed for the proper execution of the components of CORBA.

PATH=< ACE+TAO Installation folder>\bin>;
< ACE+TAO Installation folder>\ace\lib>;
<ACE+TAO Installation folder>\bin\release>

Once you set all these environment variables, the next step is to compile the ACE+TAO libraries and executables. CORBA compilation is not addressed here (you should refer to the user manual of ACE+TAO), but we give you some guidelines for the correct performance of BABEL.

-Make sure you compile ACE+TAO using static libraries (not using the “debug” option).

-The result of the compilation should produce, among others, the files: tao_idl.exe, CosEvent_Service.exe, and Naming_Service.exe. You could be able to find them at: C:\CORBA\bin\Release and under C:\CORBA\TAO\orbsvcs

BABEL Installation

BABEL does not require any installation, just unzip the downloaded file into an empty folder. Inside this folder you will find the Module Designer, the Module Debugger, and the Execution Manager (see figure 1).

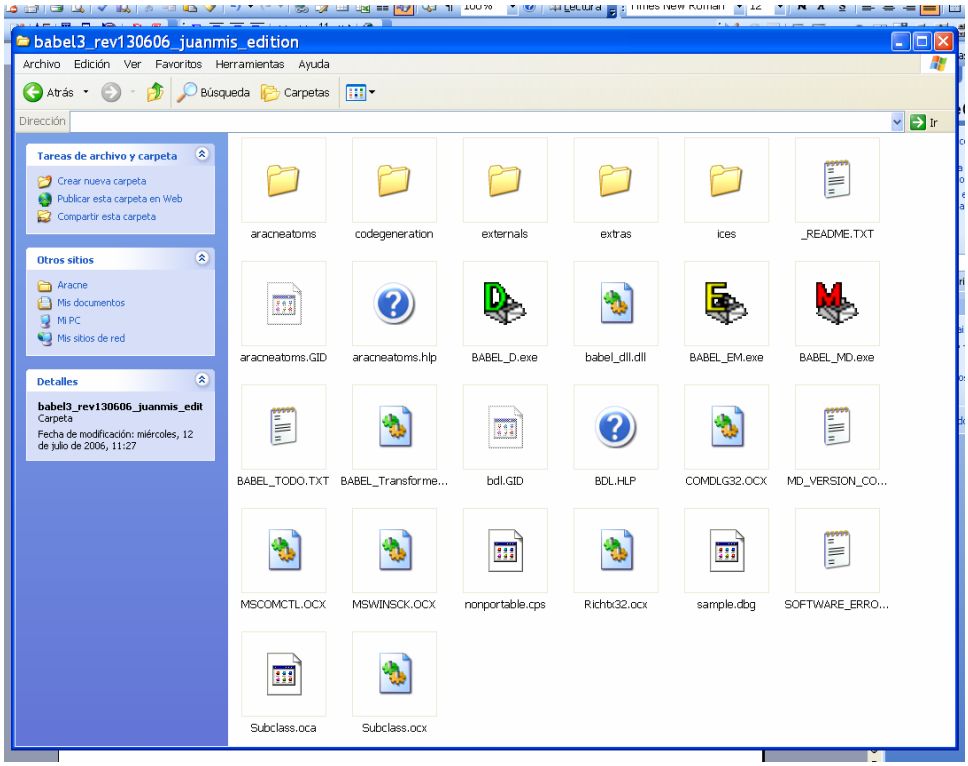


Figure 1. Snapshot of the BABEL folder

It is strongly recommended to include the Execution Manager application into the Start-Up folder in all the machines of the network to automatically run the application when

windows starts. In this way, all the users running the Execution Manager may call remote services, as well as to receive requests from other users (see figure 2).

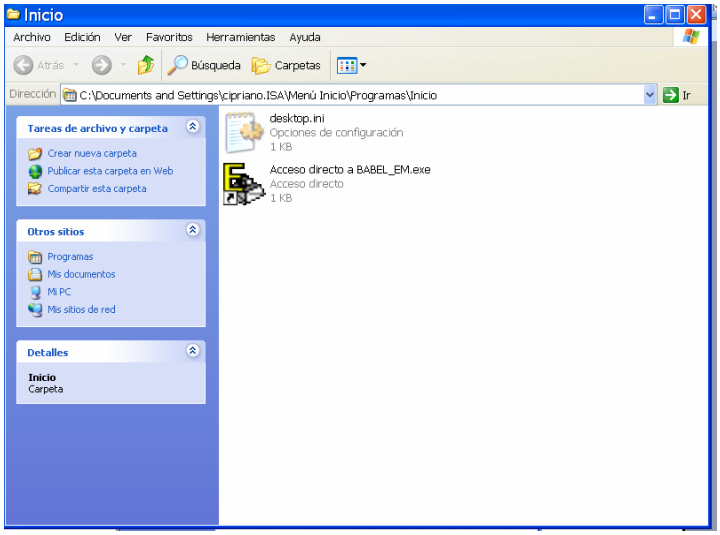


Figure 2. Including a direct access to the Execution Manager application into the start-up folder.

Visual Studio Configuration

In order to compile and link the resultant code automatically generated by BABEL, the C compiler must be properly configured. In our case, the following steps are needed:

-Add the following directories for external executable files. To do that, click on Tools →Options → Directories, and for the "Show directories for" chose "Executable Files" (see figures 3 and 4). Then add the following paths:

<ACE+TAO Installation folder>\bin>;
<ACE+TAO Installation folder>\bin\release>;
<ACE+TAO Installation folder>\tao>

-For "Library Files" add:

<ACE+TAO Installation folder>\ace>;
<ACE+TAO Installation folder>\tao\orbsvcs\orbsvcs>;
<ACE+TAO Installation folder>\tao\tao>

-In the path of the system, the following directories must appear:

<Visual Studio Installation folder\VC98\bin>;
<Visual Studio Installation folder\Common\MSDev98\bin>;
<Visual Studio Installation folder\VC98\Include>;
<Visual Studio Installation folder\VB98>

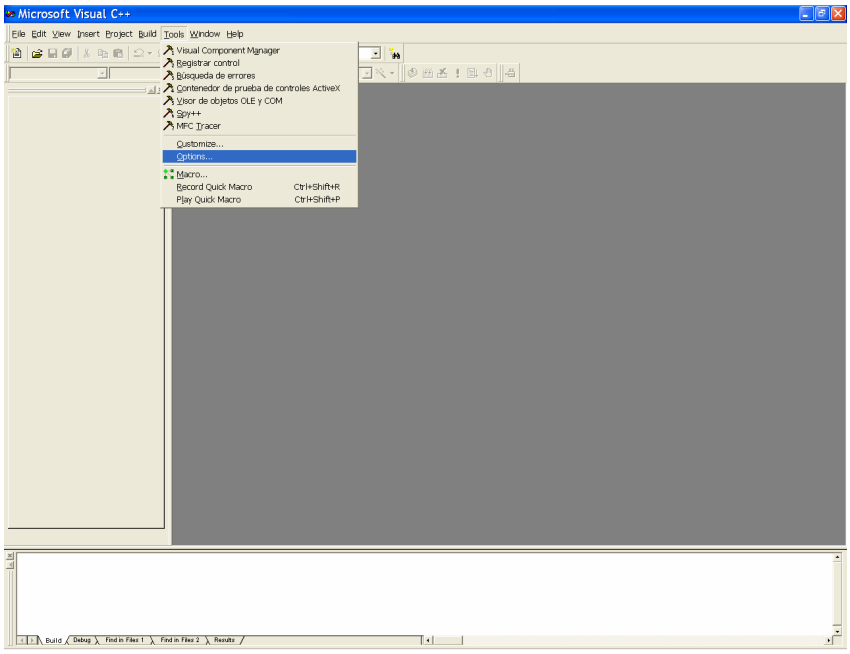


Figure 3. Visual C++ Studio Tool Configuration

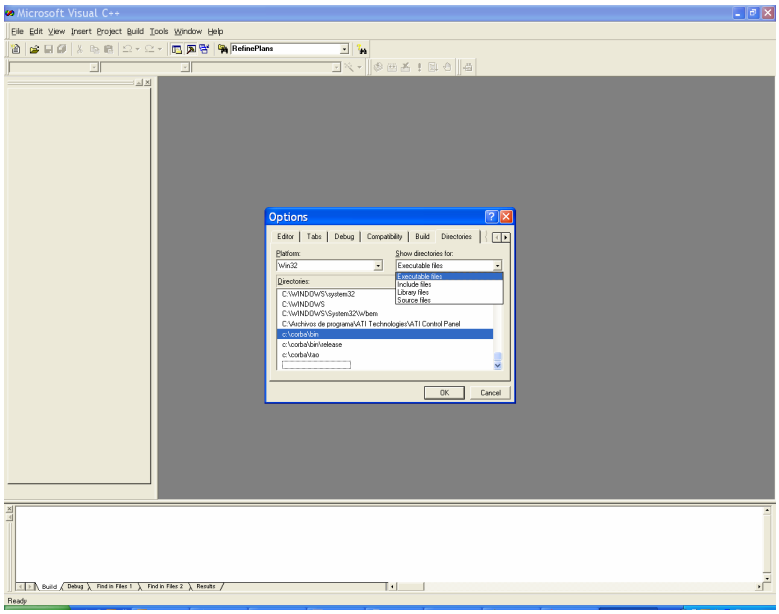


Figure 4. Visual C++ Studio Directories Configuration

Compilation of code automatically generated by BABEL

Code generated by BABEL can be compiled and linked in two manners: 1) automatically from the Module Designer and 2) manually from Visual C++.

1. Automatic Compilation of Code from the Module Designer

In the Module Designer and after the implementation of the portable design of a module, it can be automatically generated by selecting the code generator, and pressing the button “GENERATE” as shown in figure 5.

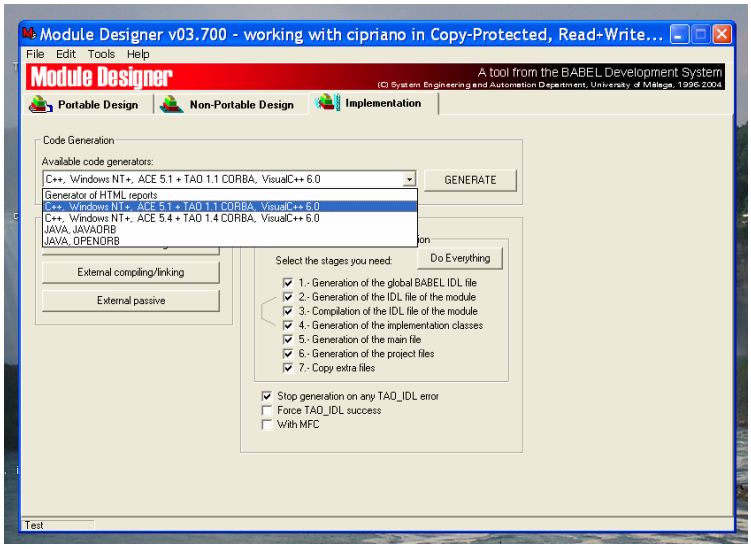


Figure 5. Automatic code generation by the Module Designer

After code generation, it can be automatically compiled and linked by pressing the button “ACCEPT” in the emergent window (see figure 6).

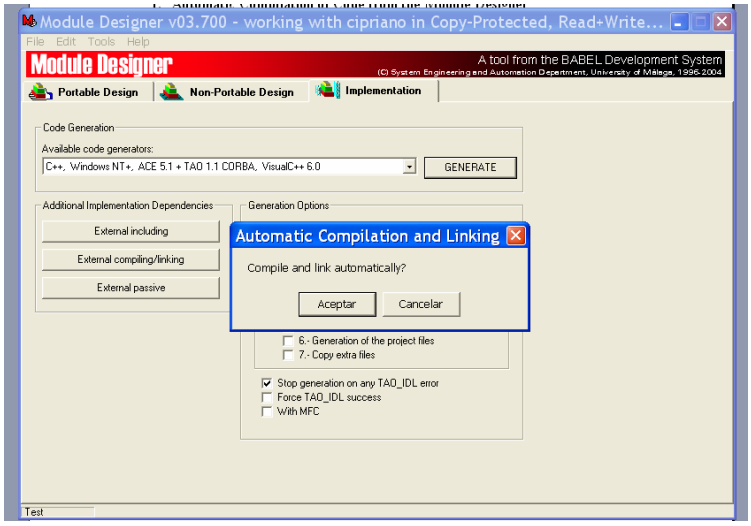


Figure 6. Accepting and automatic compilation and linkage of the generated code.

Finally, when the compilation phase ends, the results are shown (see figure 7)

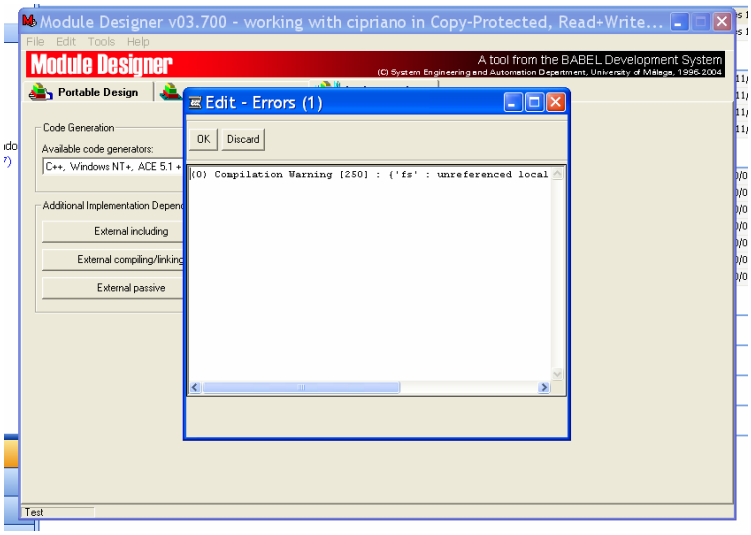


Figure 7. Compilation errors reported to the user.

2. Manual Compilation of Code

After the generation phase, the resultant code is store in a folder specified by the user (see figure 8). Inside you will find a Visual Studio project file (.dsw) that contains the compilation and linkage information of the generated code. The user can open this file and manually compile the code from Visual C++ 6.0.

Make sure the active configuration for the project is set to “Release” (see figure 9). After that, press the *Build* button (or F7) to compile and link the code (see figure10).

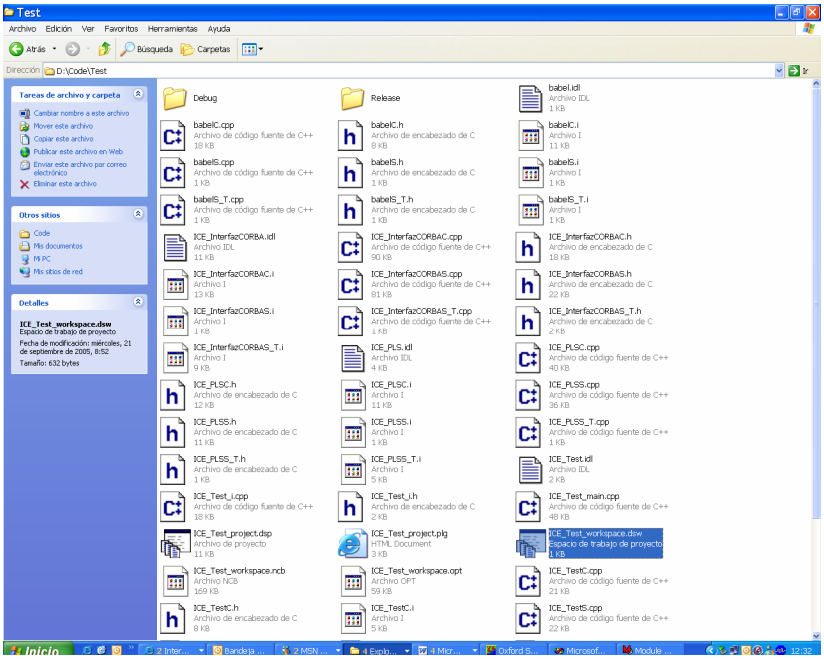


Figure 8. The code generated folder.

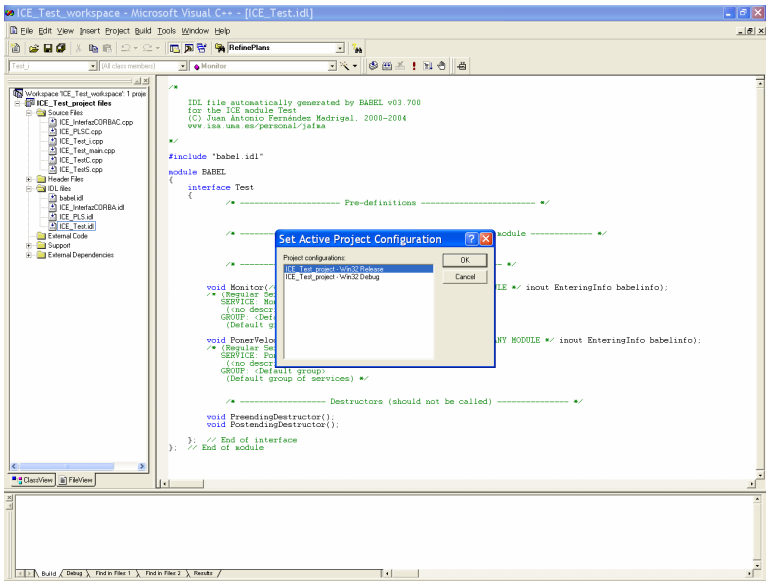


Figure 9. Changing the active configuration to Release. In VC this option can be change by clicking in “Build→Set active Configuration”

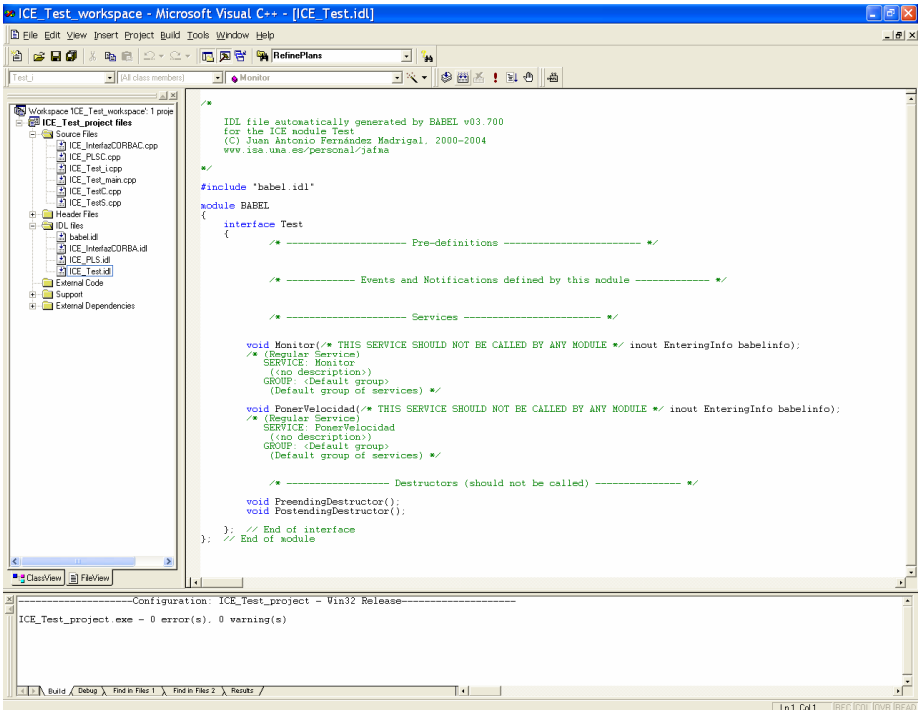


Figure 10. Compilation of the code in Visual C++