Tools Manual for Open AT® IDE 1.02

Revision: 005
Date: November 2006

OPEN AT

wavecom
Make it wireless

Operating Systems | Integrated Development Environments | Plug-Ins | Wireless CPUs | Services

Tools Manual for Open AT® IDE 1.02

Revision: 005

Date: November 20, 2006

Reference: WM_DEV_OAT_UGD_018

Document History

Index	Date	Versions	
001	May 12, 2006	Creation	
002	August, 2006	Update	
003	October 13, 2006	Update for Open AT® IDE 1.01	
004	November 2, 2006	Update	
005	November 20, 2006	Update for IDE 1.02	

Trademarks

Copyright

Table of Contents

List of Figures

1 Introduction

1.1 Purpose

This document is the user's guide for the Open AT® Software Development Kit.

1.2 References

I. Tutorial for IDE 1.02 (ref WM_DEV_OAT_UGD_016)
II. ADL User Guide
III. AT Command Interface Guide

1.3 Glossary

AT commands	Set of standard modem commands.
Embedded application	User application sources to be compiled and run on a Wavecom GSM product.
Embedded OS	Software that includes the Embedded application and the Wavecom library.
Embedded software	User application binary: set of Embedded application sources + Wavecom library.
External application	Application external to the Wavecom product that sends AT commands through the serial link.
Target	Open AT® compatible product supporting an Embedded Application.
Target Monitoring Tool	Set of utilities used to monitor a Wavecom product.
Remote Application	Set of libraries with which the User application can be run on a PC.
Wavecom library	Library delivered by Wavecom to interface Embedded application sources with Wavecom OS functions.
Wavecom OS	Set of GSM and open functions supplied to the User.

1.4 Abbreviations

API	Application Programming Interface
CPU	Central Processing Unit
FCM	Flow Control Manager
IDE	Integrated Drive Electronics
IR	Infrared
JVM	Java Virtual Machine
KB	Kilobyte
OS	Operating System
PDU	Protocol Data Unit
RAM	Random-Access Memory
ROM	Read-Only Memory
RTK	Real-Time Kernel
SMA	SMall Adapter
SMS	Short Message Services
WPB	Wavecom Packed Binary

1.5 Notation conventions

In this document, the following notations will be used:

- Command line: `wmmake <filename>`.
- Environment or makefile variable: **WMATHOME**
- File name: appli.c
- Directory name: **AppliName**

2 Open AT® IDE Overview

The Open AT® IDE is a software suite that can be used to configure, develop, build and debug Open AT® applications. It is made up of the following software:

- Open AT® IDE Settings application, for IDE parameter configuration
- Open AT® Project Wizard application, for the configuration of parameters on each project
- Eclipse IDE & CDT plug-in, for application development
- Compilers suites, both for target and debug modes
- Wavecom Development Toolkit, for target monitoring and debugging
- Remote Task Environment, for host debugging
- Open AT® IDE Command Line (for advanced users), to enter build and configuration commands

2.1 Application Configuration

IDE settings and each application project can be configured with the Open AT® IDE Settings and Open AT® Project Wizard applications. The interface of these applications are described in Chapters 3.1and 4.1.

2.2 Application Development

The Eclipse IDE is supplied with the Open AT IDE® package and must be used to develop Open AT® applications.

Please note that an up to date Java Virtual Machine (JVM) must be installed on the system for Eclipse to run correctly. A JVM is also supplied with the Open AT IDE®.

Open AT® IDE still supports Microsoft Visual C++® 6.0, .NET 2002 & .NET 2003.

2.3 Building Applications

Applications must be built with one of the supported IDE (please refer to chapter 5) or from the Open AT® IDE Command Line (please refer to chapter 8).

The current version of Open AT® is compatible with the following compiler suites:

- ARM Development Suite (ADS) 1.2
  This software suite must be installed separately from the Open AT® IDE
- ARM-ELF GCC cross-compiler
  Supplied with the Open AT® IDE package
- GCC compiler for Windows® (MINGW)
  Supplied with the Open AT® IDE package

Caution:

1) Wavecom does not guarantee correct operation if software is generated with any other compiler/linker versions.

2) The supplied ARM-ELF GCC compiler runs only on Windows NT/2000/XP versions (it is not supported on Windows 95/98/ME systems).

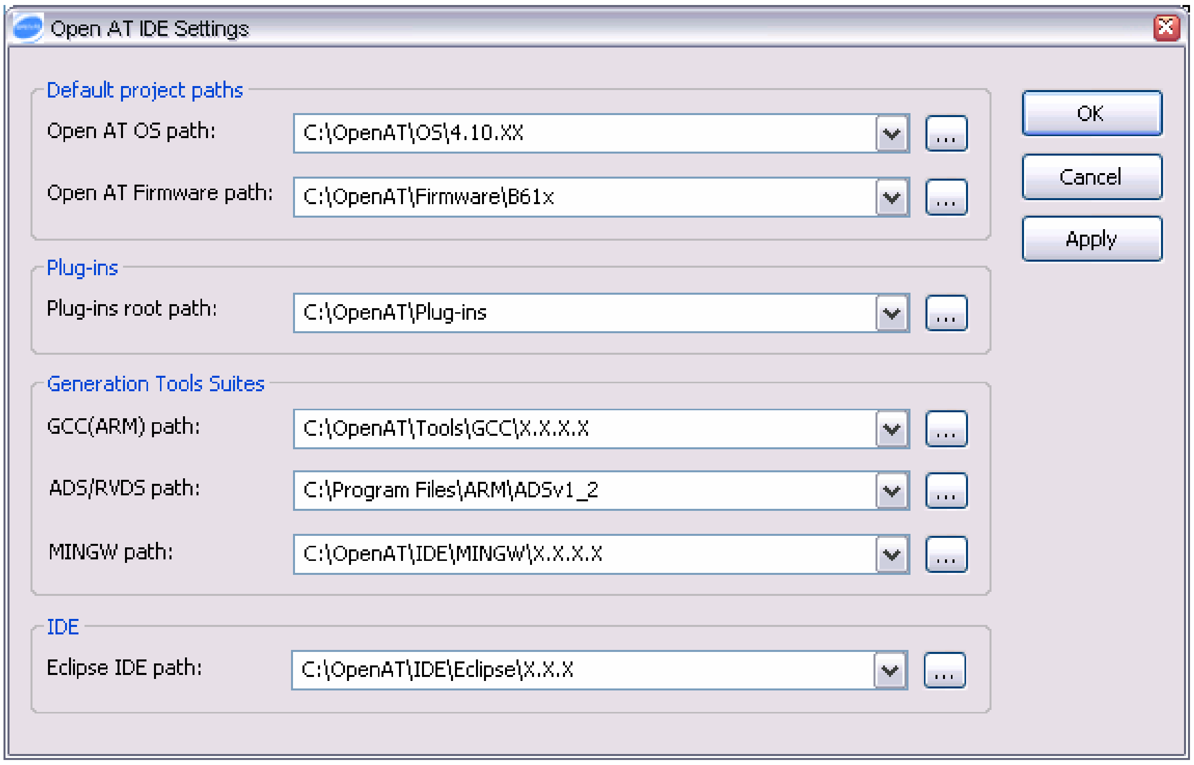
2.4 Debugging Applications

Once built, any application can be debugged with the Wavecom Development Toolkit (please refer to Chapter 6), and the Remote Task Environment (please refer to Chapter 7).

3 Open AT® IDE Settings

3.1 Tool description

The Open AT® Software Suite setup installs the "Open AT® IDE Settings" application on the system. The application should be launched from the Start menu shortcut (Wavecom\Open AT IDE\Open AT IDE Settings).


Open AT IDE Settings

Default project paths
Open AT OS path: C:\OpenAT\OS\4.10.XX
Open AT Firmware path: C:\OpenAT\Firmware\B61x

Plug-ins
Plug-ins root path: C:\OpenAT\Plug-ins

Generation Tools Suites
GCC(ARM) path: C:\OpenAT\Tools\GCC\X.X.X.X
ADS/RVDS path: C:\Program Files\ARM\ADSv1_2
MINGW path: C:\OpenAT\IDE\MINGW\X.X.X.X

IDE
Eclipse IDE path: C:\OpenAT\IDE\Eclipse\X.X.X

Figure 1: Open AT® IDE Settings application

This application is used to set the default path values for Open AT applications creation, compilers and plug-ins.

3.2 Available settings

3.2.1 Open AT® OS path

This path is the currently used OS location (where Open AT® OS Libraries, Header files documentation and samples are installed). It is the path which is proposed by default in the Open AT® Project Wizard when creating a new application.

The Open AT® settings application will automatically browse for all installed versions in a supplied directory (for example, if **C:\OpenAT\OS\4.10.XX** and **C:\OpenAT\OS\4.10.YY** versions are installed on the system, the application will automatically fill the choice list with all versions found).

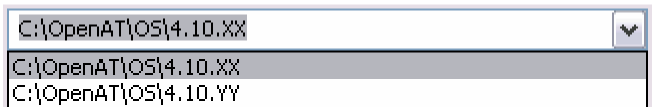

C:\OpenAT\OS\4.10.XX
C:\OpenAT\OS\4.10.XX
C:\OpenAT\OS\4.10.YY

Figure 2: Available OS installed versions sample

The browse button should also be used to find the required OS version.

3.2.2 Open AT® Firmware path

This path is the currently used Firmware location (where Open AT® Firmware binaries, RTE kernel and documentation are installed). It is the path which is proposed by default in the Open AT® Project Wizard when creating a new application.

The Open AT® settings application will automatically browse for all installed versions in a supplied directory (for example, if **C:\OpenAT\Firmware\B61x** and **C:\OpenAT\Firmware\B61y** versions are installed on the system, the application will automatically fill the choice list with all versions found).

Please note that only Firmware compatible with the currently selected OS version will be proposed in this list.

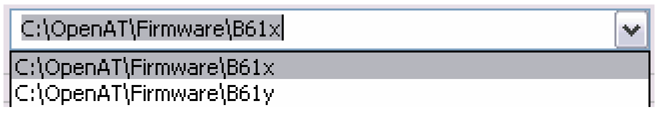

C:\OpenAT\Firmware\B61x
C:\OpenAT\Firmware\B61x
C:\OpenAT\Firmware\B61y

Figure 3: Available Firmware installed versions sample

The browse button should be used to find the required Firmware version.

3.2.3 Plug-ins root path

This path is the root directory where plug-in libraries (such as WIPor GTi) are installed. It is the path which is proposed by default in the Open AT® Project Wizard when creating a new application.

The history and browse buttons should be used to find the required plug-in libraries root path.

3.2.4 Generation Tools Suites settings

3.2.4.1 GCC (ARM) compiler path

This path is the currently used GCC compiler for ARM target version location. This path is automatically set by the Open AT® Software Suite setup if the supplied GCC compiler is installed.

The history and browse buttons should be used to find the required GCC compiler path.

3.2.4.2 ADS/RVDS compiler path

This path is the currently used ARM ADS/RVDS compiler version location. This path has to be set by the user once he has installed the ARM compiler on his computer.

The history and browse buttons should be used to find the required ADS/RVDS compiler path.

3.2.4.3 MINGW compiler path

This path is the currently used GCC compiler for Windows® version location. This path is automatically set by the Open AT® Software Suite setup if the supplied MINGW compiler is installed.

The history and browse buttons should be used to find the required MINGW compiler path.

3.2.5 Eclipse IDE path

This path is the location where the Eclipse IDE is installed.

The history and browse buttons should be used to find the required Eclipse IDE location.

3.3 Applying changes

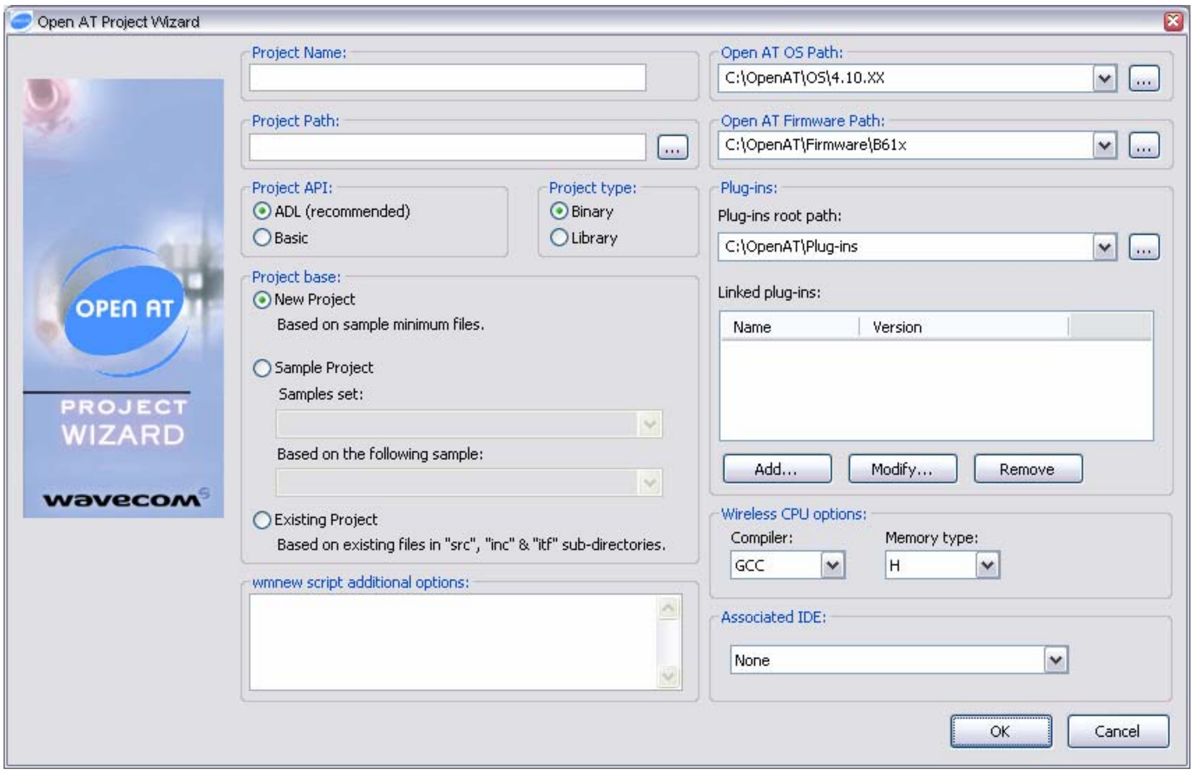
When changes have been made in the Open AT® settings, once the "Apply" or "OK" button is pressed, the new parameters values are taken into account as described below:

- OS, Firmware & Plug-in paths are the default paths proposed in the Open AT® Project Wizard. They are initialized when this wizard is launched to create a new application.
- Generation Tools Suites parameters modifications will be immediately applied on the next compilation process.
- IDE settings will be applied after launching the Open AT® Project Wizard to create or update a project.

4 Open AT® Project Wizard

4.1 Tool description

The Open AT® Software Suite setup installs an "Open AT® Project Wizard" application on the system. The application may be launched from the Start menu shortcut (Wavecom\Open AT IDE\Open AT Project Wizard).



Project Name:

Project Path:

Project API: [✓ ADL (recommended)] Basic

Project type: [✓ Binary] Library

Project base:
[✓ New Project] Based on sample minimum files.
Sample Project. Samples set: Based on the following sample:
Existing Project. Based on existing files in "src", "inc" & "itf" sub-directories.

wmnew script additional options:

Open AT OS Path: C:\OpenAT\OS\4.10.XX

Open AT Firmware Path: C:\OpenAT\Firmware\B61x

Plug-ins:
Plug-ins root path: C:\OpenAT\Plug-ins
Linked plug-ins:

Name	Version

Wireless CPU options:
Compiler: GCC
Memory type: H

Associated IDE: None

Figure 4: Open AT® Project Wizard

This application has to be used to create, or edit Open AT® project software common settings (saved in the corresponding <Project>.scs file), and also the selected associated IDE workspace.

4.2 Basic settings

4.2.1 Project Name

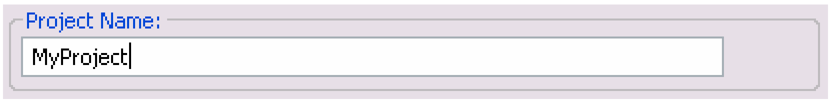


Figure 5: Project Name

This setting will be the name of the Open AT® project (software common settings will be saved in the <ProjectName>.scs file).

When using the Sample base mode, the name will be set to the selected sample name.

When browsing for an existing project directory, if an .scs file is found in the selected directory, this file name will be used as the project name.

4.2.2 Project Path

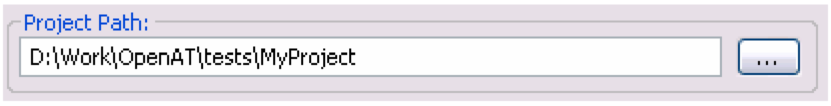


Figure 6: Project Path

This setting is the path to the location where the Open AT® project is located.

The browse button may be used to find the required project Path. New directories can be created when browsing for the project path. If the path selected by the browse button contains an .scs file, the Wizard goes to the Existing Project mode, and reads the project name and the project options from this selected directory.

4.2.3 Used API

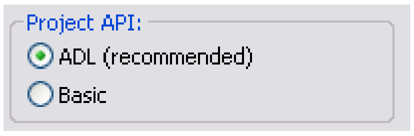
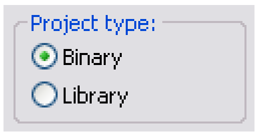


Figure 7: Project API

This setting is the API used by the current Open AT® project. The ADL API is strongly recommended (The basic API should be used by advanced users only).

4.2.4 Project Type



Project type: [✓ Binary] Library

Figure 8: Project type

This setting is the current project type (binary or library) which will be generated within the Development Environment.

4.2.5 Wmnew script additional options

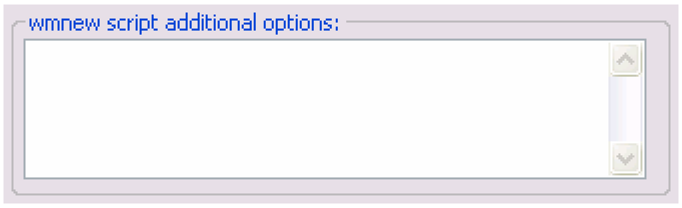
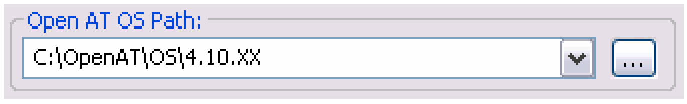


Figure 9: `wmnew` additional options

This text field may contain additional options for the `wmnew` script; please refer to the Command Line Scripts chapter for more details. Only the described starting from Chapter 8.1.6 (Project global options) may be used; the other options are all handled by the Open AT® Project Wizard.

This option is available only in New Project & Existing Project modes.

4.2.6 Open AT® OS path

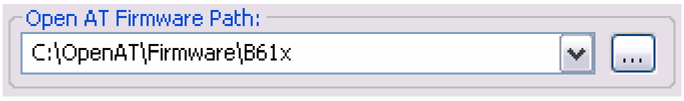


Open AT OS Path: C:\OpenAT\OS\4.10.XX

Figure 10: Open AT® OS Path

This path is the OS version used by the current project. By default for new projects, the value is the value set in the Open AT® IDE Settings. For existing projects, the value is the one saved in the Software Common Settings.

4.2.7 Open AT® Firmware path



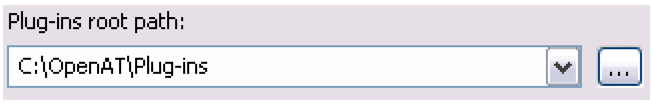
Open AT Firmware Path: C:\OpenAT\Firmware\B61x

Figure 11: Open AT® Firmware Path

This path is the Firmware version used by the current project. By default for new projects, the value is the value set in the Open AT® IDE Settings. For existing projects, the value is the one saved in the Software Common Settings.

4.2.8 Plug-ins

4.2.8.1 Plug-ins root path

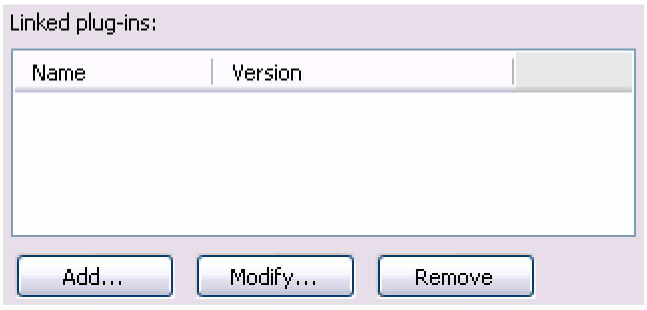

Plug-ins root path:

C:\OpenAT\Plug-ins

Figure 12: Plug-ins root path

This path is the plug-ins root directory used by the current project. By default for new projects, the value is the value set in the Open AT® IDE Settings. For existing projects, the value is the value saved in the Software Common Settings.

4.2.8.2 Linked plug-ins


Linked plug-ins:

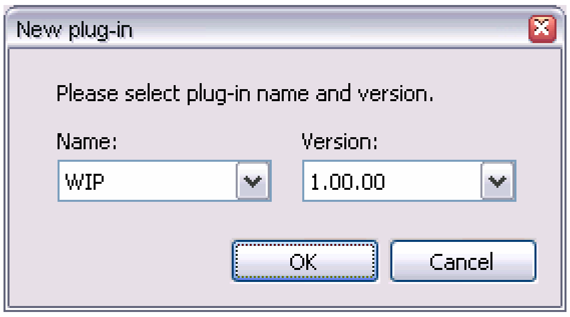
Name	Version

Figure 13: Linked plug-ins

This list is used to select the plug-ins required to be linked against the current project. Each plug-in added to this list will automatically be embedded in the Open AT® application.

Please note that for each linked plug-in, an additional sample set is available, containing application codes which demonstrate the plug-in functioning.

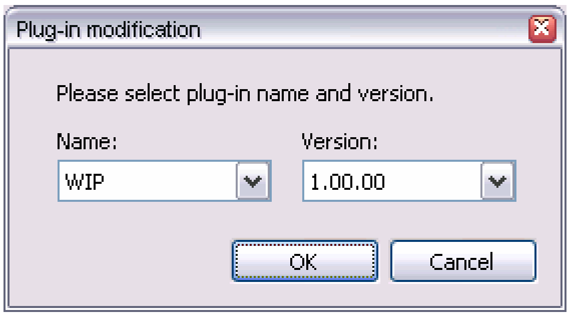
The "Add" button has to be used to add a plug-in to the list.



Please select plug-in name and version.
Name: WIP
Version: 1.00.00

Figure 14: Adding a plug-in reference

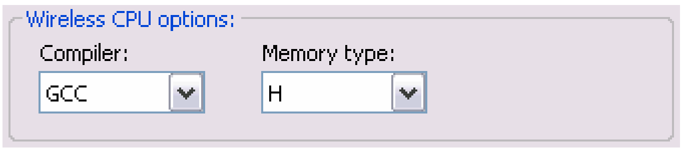
Then, any entry in the list can be removed with the "Remove" button, or modified with the "Modify" button.



Please select plug-in name and version.
Name: WIP
Version: 1.00.00

Figure 15: Modifying a plug-in reference

4.2.9 Wireless CPU® options



Compiler: GCC
Memory type: H

Figure 16: Modifying a plug-in reference

These options configure the build process for the target mode.

4.2.9.1 Compiler

This parameter is the used Generation Tools Suite used to build the target application binary. Available options are:

- GCC compiler (for ARM processor, selected by default).
- ARM Development Suite (ADS) compatible compiler, usable only if installed separately by the user on the system, and if the ADS/RVDS path has been set correctly in the Open AT® IDE Settings.

4.2.9.2 Memory type

This parameter is the target Wireless CPU® memory type, used by the IDE to link the application to the right address. Available values depend on the currently used Firmware version.

If a Firmware allowing a single link address is used, the Memory Type selection list will be disabled (see Figure 17: Disabled memory selection list), since the generated application binary will be independent of the Wireless CPU® memory type.

Figure 17: Disabled memory selection list

4.2.10 Associated IDE

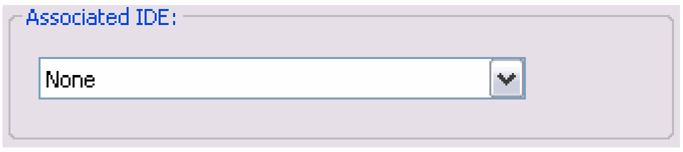


Figure 18: Associated IDE

This option is used to select the IDE required to be launched with the current project. The drop-down list contains only the installed environments on the system. Supported IDEs are:

- None: no specific IDE required, the project will have to be built from the Open AT® IDE Command Line.
- Eclipse.
- Microsoft Visual C++® 6.0.
- Microsoft Visual C++® .NET 2002.
- Microsoft Visual C++® .NET 2003.

Once generated via the Wizard, the IDE workspace can be opened from the Windows® explorer by launching the project file associated with this IDE:

- `LoadEclipse.bat` file for Eclipse.
- `.dsw` file for Microsoft Visual C++® 6.0.
- `.sln` file for Microsoft Visual C++® .NET 2002 or 2003.

4.3 Wizard modes

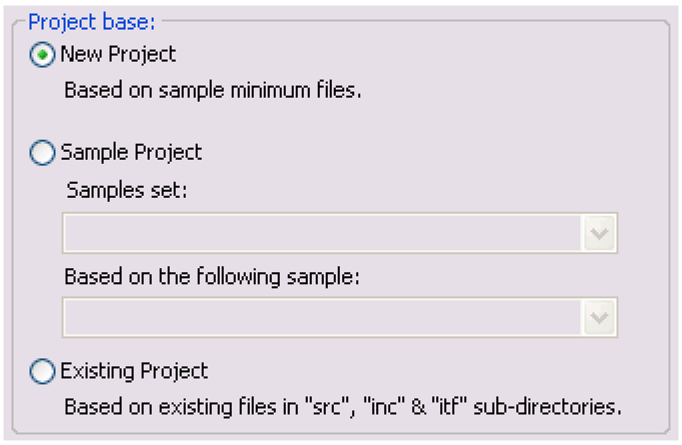

Project base:
New Project
Based on sample minimum files.
Sample Project
Samples set:
Based on the following sample:
Existing Project
Based on existing files in "src", "inc" & "itf" sub-directories.

Figure 19: Wizard modes

The Wizard behavior is determined by the mode selected in this option field.

4.3.1 New Project

In this mode, minimum code sample files will be copied in the current directory, before building the project settings.

The `wmnew` additional option field is available in this mode, to set-up other project settings options.

4.3.2 Sample Project

This mode creates a new project, based on the selected sample in the drop-down list. Available samples are sorted in sample sets, supplied with the used Open AT® OS package, or with selected Open AT® plug-ins. The drop-down list contains then all the samples in the current samples set.

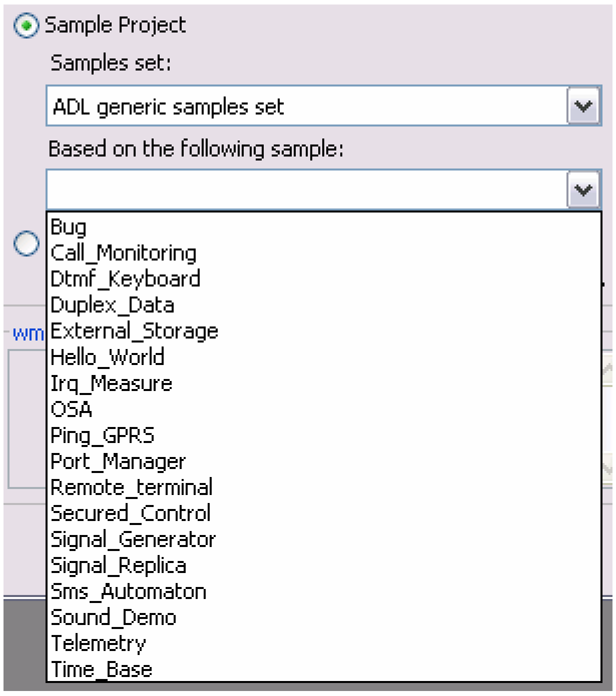


Figure 20: Samples list

When a new sample is selected in the list, the project name is updated to be renamed with the sample name (only if the project name was empty or if it was another sample name).

If the selected sample depends on one or several sample libraries, this/these library/libraries project(s) will also be copied to the Libraries subdirectory.

4.3.3 Existing Project

In this mode, the Project Wizard is used to update an existing project settings. To edit an existing project, the path browse button has to be used to find the project path. The project name and the `wmnew` option field will then be updated with the values read from this directory.

The `wmnew` additional option field is available in this mode to set-up other project setting options.

4.4 Updating existing projects

Important Warning: It is strongly recommended to always use the Project Wizard to update existing projects (to add source files, include paths or others), rather than hand-editing make files, or modifying projects in development environments.

An existing project may be opened with the wizard in the following ways:

- Browse for the existing project path with the Project Wizard application;
- Directly open the project from the Windows Explorer (".scs" files are automatically associated with the Open AT® Project Wizard).

4.5 Error messages

The Open AT® Project Wizard may display the following error messages.

"Please enter a name to create the new project..."

The project name text field is empty.

"Not authorized characters (space, ".", "$") found in the project name..."

The project name field can contain any characters, except spaces (" "), dollars ("$") and dots (".").

"Not authorized characters (space, "$") found in the project path..."

The project path field can contain any characters, except spaces (" ") and dollars ("$").

"Please select a sample to create the new project..."

The Project Wizard is in Sample mode, but no sample was selected in the drop-down list.

"Please select a path to create the new project..."

The project path text field is empty.

"Bad format for provided path string..."

The project path text field does not contains an absolute path (in Windows format).

"Unable to create the folder."

A new directory was required to be created, but creation failed.

"Wmnew script error X "

Please refer to the `wmnew` *script documentation for more details.*

5 Building Open AT® applications

5.1 Open AT® application directories architecture

Each application directory contains the project software common settings (`.scs`) file, and also the different supported development environments project files. It also contains the following sub-directories:

- **rte**: contains the Remote Application binaries, generated with one of the supported development environments,
- **{compiler}/out**: contains the application binary ready to be downloaded to the Target, generated with the current compiler,
- **src**: contains the User Open AT® sources,
- **inc**: contains the User Open AT® header files.

5.2 Defined Compilation Flags

Default compilation flags are defined by the Open AT® IDE for all application & library projects. These flags are defined below.

5.2.1 Generic flags

__OAT_API_VERSION__

Numeric flag which contains the currently used OS API version level. Example: for Open AT® OS V4.10 interface, it is defined as **`__OAT_API_VERSION__=410`**.

__DEBUG_APP__

If this flag is defined (by default), the **`TRACE`** & **`DUMP`** macros (cf. traces service chapter in the ADL User Guide) will be compiled, and will display debug information on Target Monitoring Tool. Otherwise (using the **`wmmake`** script with the **`-release`** option), these macros will be ignored.

__DEBUG_FULL__

If this flag is defined (using the **`wmmake`** script with the **`-fulldebug`** option), the **`FULL_TRACE`** & **`FULL_DUMP`** macros (cf. traces service chapter in the ADL User Guide) will be compiled, and will display debug information on Target Monitoring Tool. Otherwise, these macros will be ignored.

5.2.2 Plug-ins flags

For each plug-in required by the application, the following flags are defined. In these flag names, the "XXX" statement stands for the plug-in name (the name selected in the Open AT® Project Wizard).

__XXX_PLUGIN_VERSION__

Numeric flag which contains the currently used plug-in version level. Example: for Open AT® WIP plug-in V1.10 interface, it is defined as **`__WIP_PLUGIN_VERSION__=110`**.

5.2.3 Generation environment specific flags

__GNU_GCC__

This flag is defined when the ARM GCC cross-compiler is used for the target binary generation.

__REMOTETASKS__

This flag is defined when the application is compiled in RTE mode.

5.3 Generated applications for target mode

Open AT® projects (applications or libraries) should be generated using a development environment (cf. following chapters), or using the `wmmake` script from the Open AT® Command Line (for advanced users, refer to Chapter 8).

5.3.1 Generated binary files

The generated binary files are:

- for a library:

  `[compiler]/out/[project].lib`

  where:

  [compiler] is the current used compiler,

  [project] is the project name.

- for an application:

  `[compiler]/out/[compiler]_[project]_[memory].dwl &`

  `[compiler]/out/[compiler]_[project]_[memory].wpb.dwl` where:

[compiler] is the currently used compiler,

[project] is the project name,

[memory] is the currently used memory type (please refer to "ADL User Guide", ref II, for more information about memory types). If the project uses a Firmware which allows a single link address, whatever is the memory type, the generated application binary file name will not include the [memory] tag; generated files will be `[compiler]_[project].dwl` & `[compiler]_[project].wpb.dwl`.

The "`.wpb.dwl`" is a compressed version of the "`.dwl`" file (wpb is the acronym for Wavecom Packed Binary).

Both files may be downloaded to the Wireless CPU®; as the "`.wpb.dwl`" file is smaller than the simple "`.dwl`" one, downloading this file will be faster.

5.3.2 Open AT® application download

Please refer to the Tutorial for IDE documentation (see Ref I) for information on how to download Open AT® application binaries on Wireless CPU®.

5.4 Eclipse IDE

5.4.1 Creating an Open AT® application workspace for Eclipse

Open AT® application projects are created with the Open AT® Project Wizard (see Chapter Open AT® Project Wizard). Eclipse will be launched as soon as the wizard will have finished creating the project.

5.4.2 Eclipse Projects

An Eclipse Open AT® workspace contains a single project, usable in target mode and remote mode.

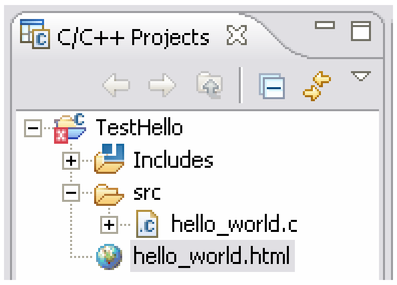


Figure 21: Open AT® project in Eclipse

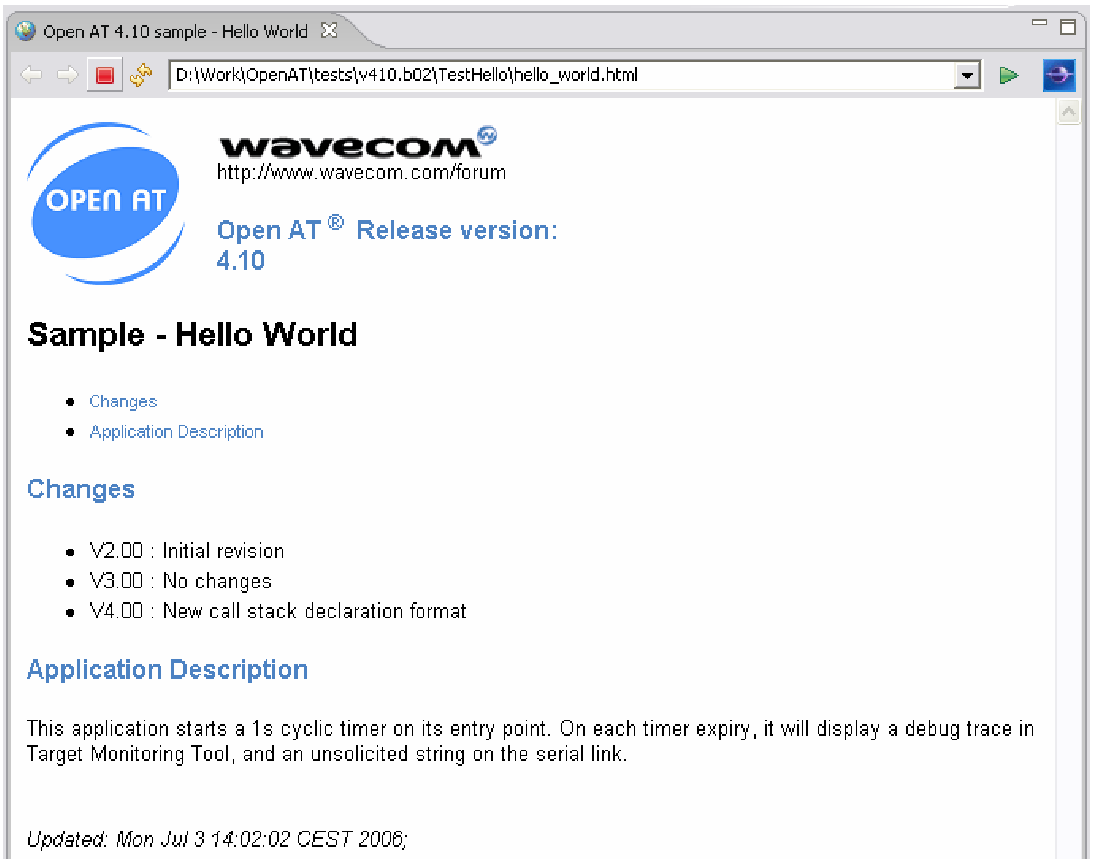
The project contains the application sources & headers, the Open AT® interface files, and the HTML help file (if the project was created from a supplied sample).

As described in the Chapter 4.4, it is strongly recommended to use always the Project Wizard to update existing projects (to add source files, include paths or others), rather than modifying projects in the Eclipse environment.

5.4.3 Displaying the HTML help file

When an Open AT® project is created from one of the provided samples, the projects includes an HTML help file describing the sample interfaces and processing.

In order to display this file from Eclipse, it simply has to be opened directly, since the IDE embeds its own HTML browser.


Open AT 4.10 sample - Hello World
D:\Work\OpenAT\tests\v410.b02\TestHello\hello_world.html
OPEN AT
wavecom
http://www.wavecom.com/forum

Open AT ® Release version:
4.10

Sample - Hello World

- Changes
- Application Description

Changes

- V2.00 : Initial revision
- V3.00 : No changes
- V4.00 : New call stack declaration format

Application Description

This application starts a 1s cyclic timer on its entry point. On each timer expiry, it will display a debug trace in Target Monitoring Tool, and an unsolicited string on the serial link.

Updated: Mon Jul 3 14:02:02 CEST 2006;

Figure 22: Application sample HTML help file display

5.4.4 Building an application for Target or RTE Mode

To build the application for target or RTE mode, the "Make Targets" view has to be opened, and then one of the "Build" / "Clean" / "Re-build all" targets has to be double-clicked to launch the clean/build process for the required mode.

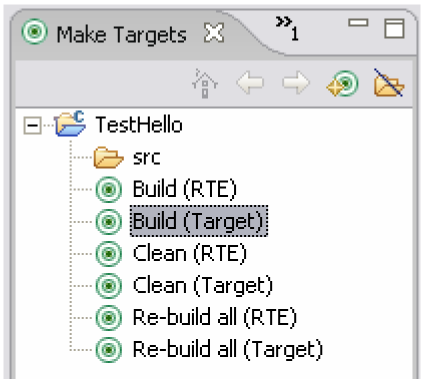


Figure 23: Make targets for clean/build process launch

The compilation results will be displayed in the Console window.

Compilation & link errors/warnings should be displayed in the Problems window (Only for GCC and MINGW compilers; errors generated by the ADS compiler are not parsed by the Eclipse IDE).

Caution:

Some detected error and warning messages may not be relevant, since the default Eclipse errors parser is not fully compatible with the Open AT® build process output.

5.4.5 Starting the Remote Task Environment

Once the project is built in RTE mode, a debug session can be launched by the toolbar button.

Note:

During the debug session, if new breakpoints are set, these ones should not be taken into account for the current run. They will be enabled only for the next debug session.

5.5 Microsoft Visual C++ 6.0 development environment

5.5.1 Creating an Open AT® application workspace for Visual C++ 6.0

Open AT® application projects are created with the Open AT® Project Wizard (see Chapter Open AT® Project Wizard). Visual C++ will be launched as soon as the wizard has finished creating the project.

5.5.2 Visual C++ 6.0 Projects

A Visual C++ 6.0 Open AT® workspace contains two projects, for target mode (named `<project>`) and remote mode (named `<project>_rte`).

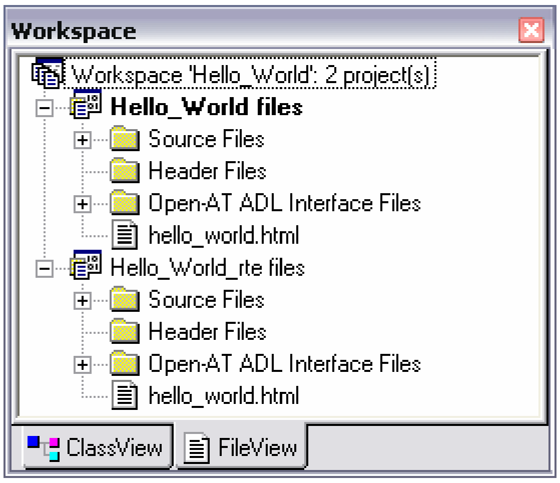


Figure 24: Open AT® projects in Visual C++ 6.0

Each project contains the same files (application sources & headers, the Open AT® interface files, and the HTML help file, if the project was created from a provided sample), which may be edited from any of these two projects.

If an Open AT® project implements libraries, all the corresponding projects (target and remote) will also be added to the main application workspace.

As written in Chapter 4.4, it is strongly recommended to use always the Project Wizard to update existing projects (to add source files, include paths or others), rather than modifying projects in the Visual C++ environment.

5.5.3 Displaying the HTML help file

When an Open AT® project is created from one of the provided samples, the projects include an HTML help file, describing the sample interfaces and processing.

In order to display this file from Visual C++, it has to be opened in the editor, and launched in the HTML browser by selecting the "Preview Page" option from the mouse right-click menu.

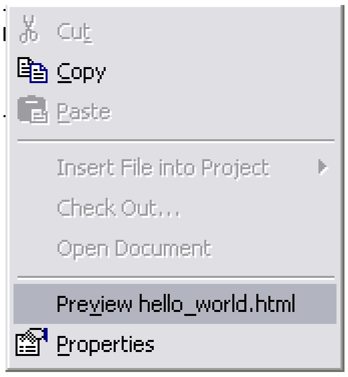
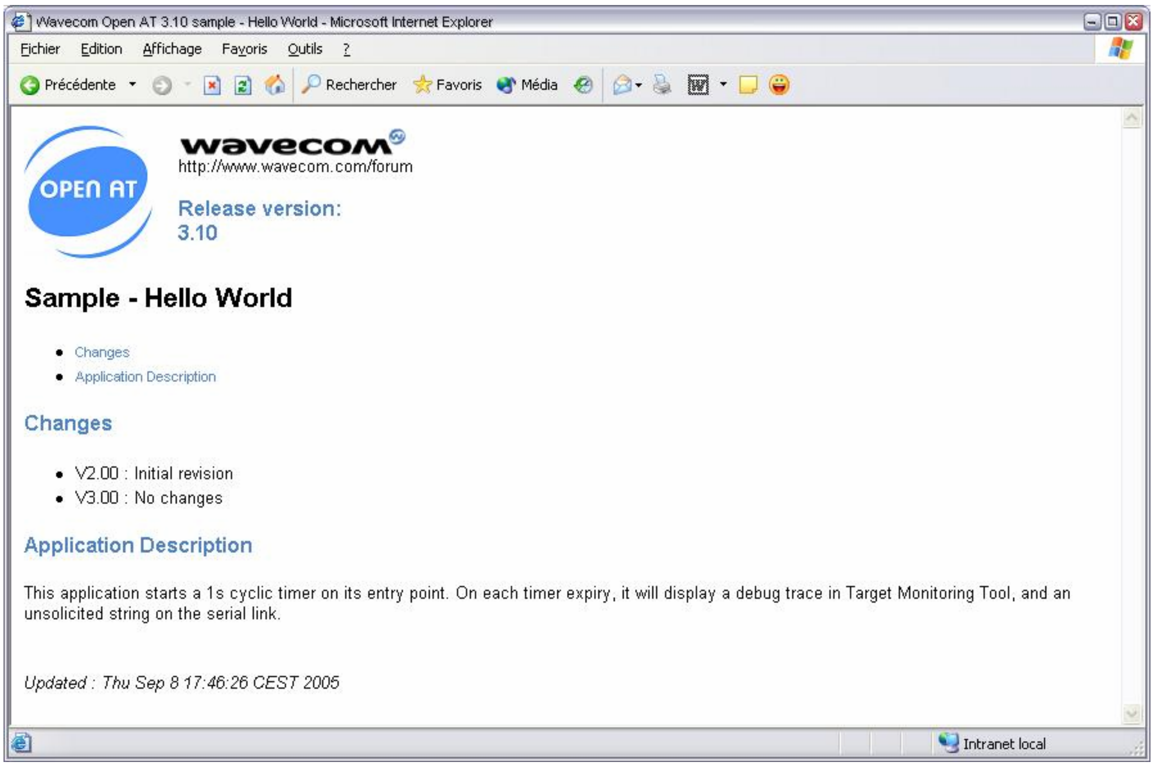


Figure 25: Preview the HTML help file from editor screen


Wavecom Open AT 3.10 sample - Hello World - Microsoft Internet Explorer
Fichier Edition Affichage Favoris Outils ?
Précédente
Rechercher
Favoris
Média
OPEN AT
wavecom
Intranet local

http://www.wavecom.com/forum

Release version:
3.10

Sample - Hello World

- Changes
- Application Description

Changes

- V2.00 : Initial revision
- V3.00 : No changes

Application Description

This application starts a 1s cyclic timer on its entry point. On each timer expiry, it will display a debug trace in Target Monitoring Tool, and an unsolicited string on the serial link.

Updated : Thu Sep 8 17:46:26 CEST 2005

Figure 26: Application sample HTML help file display

5.5.4 Building application for Target Mode

To build the application for target mode, the target project has to be selected as the active project, from the Project menu:

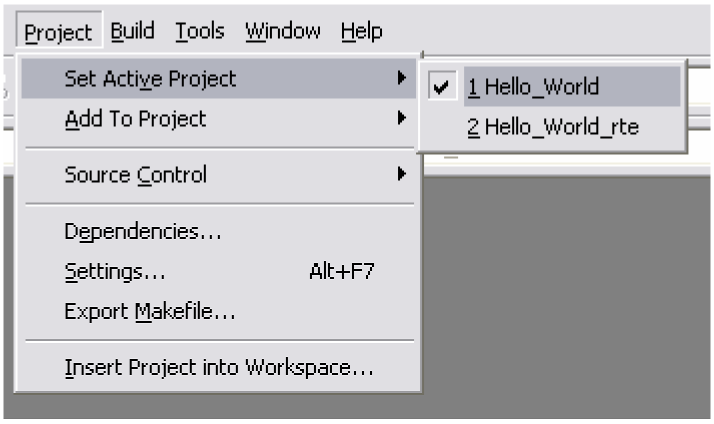


Figure 27: Select the active project – way 1

Or from the right-click mouse context menu:

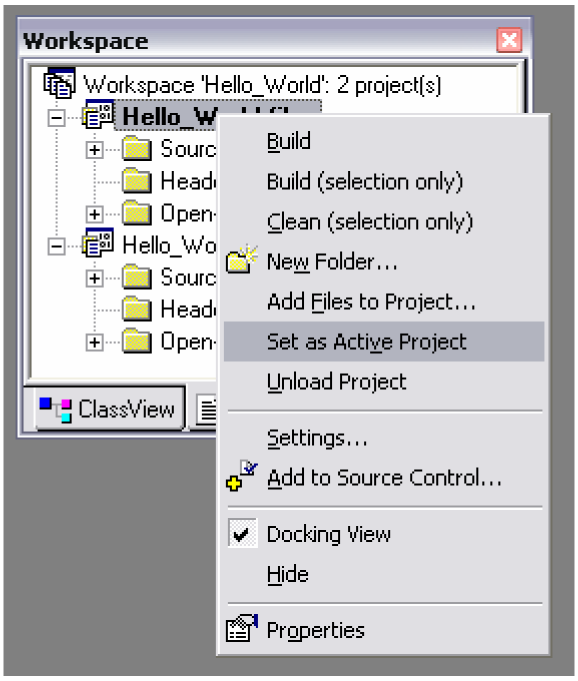


Figure 28: Select the active project – way 2

Or from the Build toolbar:

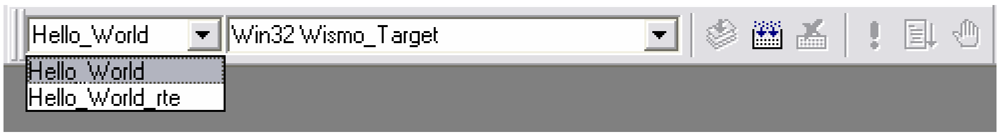


Figure 29: Select the active project – way 3

Then, the project may be built by launching the Build command, from the Build Menu, the context menu, or with the F7 key. The compilation results will be displayed in the output window.

<u>Caution:</u>

Detected error and warning messages may not be relevant, since the Visual Environment displays these messages when it detects "error" or "warning" strings in the compilation text output (i.e. for example the "no error" string will be detected as an error by the Visual Environment).

Error messages which may be displayed by the Software Generation Toolkit are listed in the **wmmake** script description.

5.5.5 Building the application for remote mode

To build the application for remote mode, the remote project has to be selected as the active project, by the Project menu, the context menu or the Build toolbar (as shown above in the previous paragraph).

Then the configuration has to be set. Both Release and Debug mode are available, but Debug mode should be chosen in order to perform step-by-step debugging in the program. The configuration may be selected from the Build->Set Active Configuration... menu:

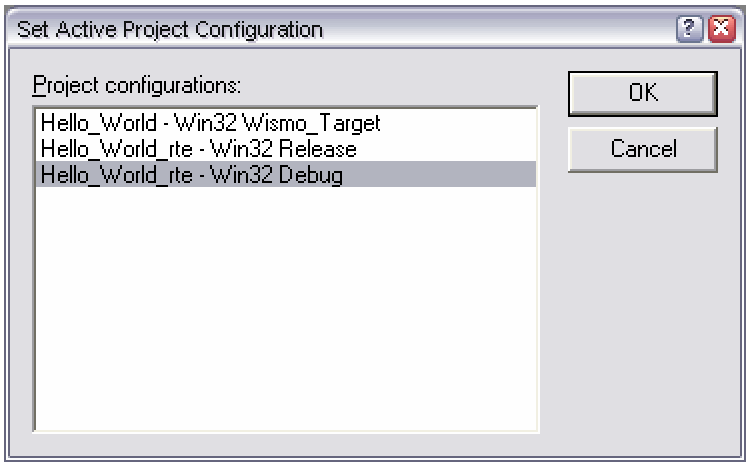

Set Active Project Configuration
Project configurations:
Hello_World - Win32 Wismo_Target
Hello_World_rte - Win32 Release
Hello_World_rte - Win32 Debug
OK
Cancel

Figure 30: Select the configuration – way 1

Or from the Build toolbar:

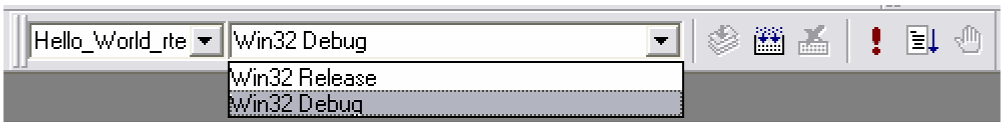


Figure 31: Select the configuration – way 2

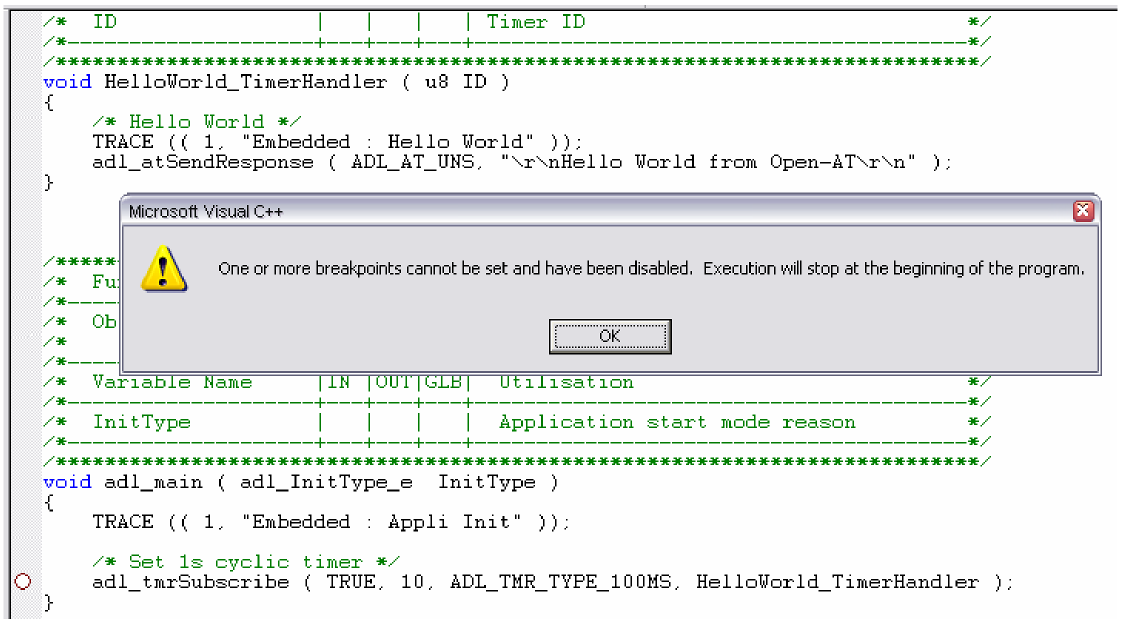
Then, the project may be built by launching the Build command from the Build Menu, the context menu, or with the F7 key. The compilation results will be displayed in the output window.

5.5.6 Setting breakpoints

Due to the Remote Task Environment architecture, breakpoints should be set once the RTE graphical interface is launched, just before pressing the Start button (see Remote Task Environment chapter).

If breakpoints are set before running the application, the Environment will inform the user that the breakpoints cannot be set (see Figure 29: Select the active project – way 3). The execution will then stop at the beginning of the program: press Go to continue running the application.

Once the application is started, the user has to return to the Visual environment, and to edit the breakpoints window (from the Edit menu, or with the Ctrl-B key). In this window, all breakpoints are displayed, and may be enabled / disabled as required (see Figure 33: Breakpoints window).



```
/*   ID                   |    |    |    | Timer ID                                    */
/*------------------------+----+----+----+---------------------------------------------*/
/***************************************************************************************/
void HelloWorld_TimerHandler ( u8 ID )
{
    /* Hello World */
    TRACE (( 1, "Embedded : Hello World" ));
    adl_atSendResponse ( ADL_AT_UNS, "\r\nHello World from Open-AT\r\n" );
}
```

Microsoft Visual C++

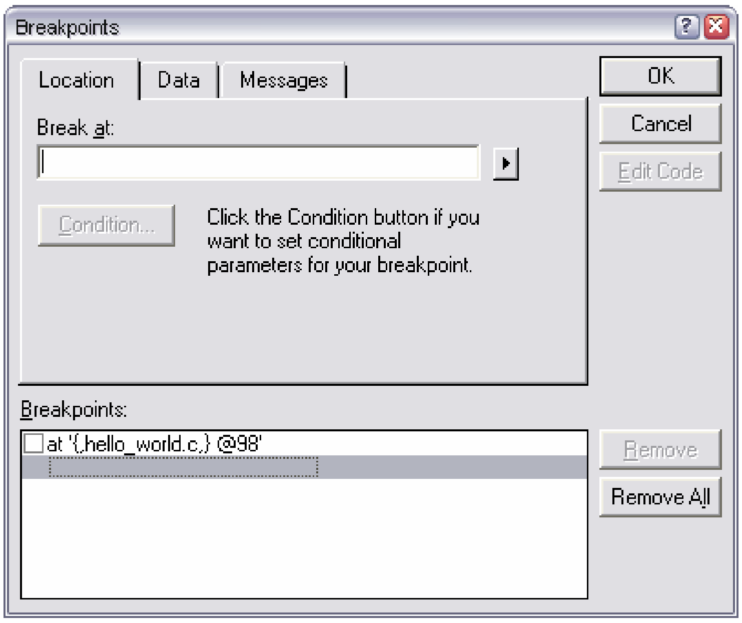
One or more breakpoints cannot be set and have been disabled. Execution will stop at the beginning of the program.

OK

```
/*   Variable Name        |IN  |OUT|GLB|  Utilisation                                   */
/*------------------------+----+----+----+---------------------------------------------*/
/*   InitType             |    |    |    |  Application start mode reason              */
/*------------------------+----+----+----+---------------------------------------------*/
/***************************************************************************************/
void adl_main ( adl_InitType_e  InitType )
{
    TRACE (( 1, "Embedded : Appli Init" ));

    /* Set 1s cyclic timer */
    adl_tmrSubscribe ( TRUE, 10, ADL_TMR_TYPE_100MS, HelloWorld_TimerHandler );
}
```

Figure 32: Unable to set breakpoints



Break at:

Click the Condition button if you want to set conditional parameters for your breakpoint.

Breakpoints:

at '{,hello_world.c,} @98'

Figure 33: Breakpoints window

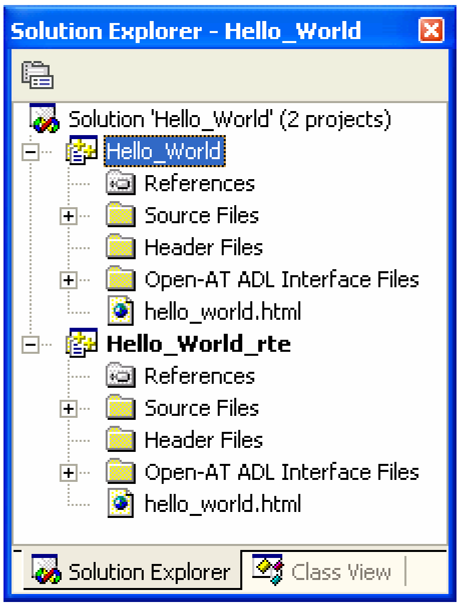
5.6 Microsoft Visual C++ .Net development environment (2002 & 2003 versions)

5.6.1 Creating an Open AT® application workspace for Visual C++ 6.0

Open AT® application projects are created with the Open AT® Project Wizard (see Open AT® Project Wizard chapter). Visual C++ will be launched as soon as the wizard will has finished creating the project.

5.6.2 Visual C++ .NET ProjectsVisual C++ .NET Projects

A Visual C++ .NET Open AT® workspace contains two projects, for target mode (named `<project>`) and remote mode (named `<project>_rte`).


Solution Explorer - Hello_World

Solution 'Hello_World' (2 projects)
- Hello_World
  - References
  - Source Files
  - Header Files
  - Open-AT ADL Interface Files
  - hello_world.html
- Hello_World_rte
  - References
  - Source Files
  - Header Files
  - Open-AT ADL Interface Files
  - hello_world.html

Solution Explorer | Class View

Figure 34: Open AT® projects in Visual C++ .NET

Each project contains the same files (application sources & headers, the Open AT® interface files, and the HTML help file, if the project was created from a provided sample), which may be edited from any of these two projects.

If an Open AT® project implements libraries, all the corresponding projects (target and remote) will also be added to the main application workspace.

As written in the Chapter 4.4, it is strongly recommended always to use the Project Wizard to update existing projects (to add source files, include paths or others), rather than modifying projects in the Visual C++ environment.

5.6.3 Displaying the HTML help file

When an Open AT® project is created from one of the provided samples, the projects include an HTML help file, describing the sample interfaces and processing.

In order to display this file in the Visual C++ Web Browser, it has to be launched by selecting the "Display in Browser" option in the mouse right-click menu.

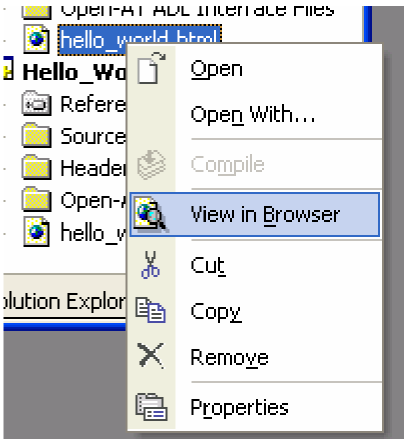
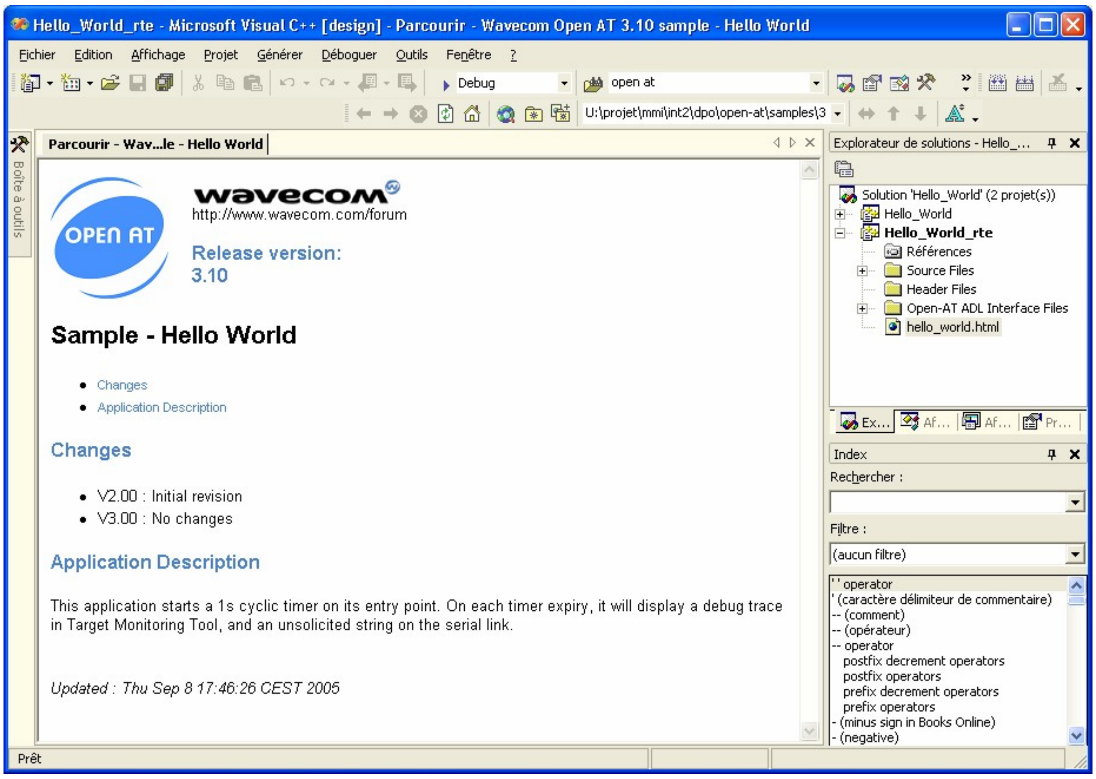


Figure 35: Preview the HTML help file from solution explorer


Hello_World_rte - Microsoft Visual C++ [design] - Parcourir - Wavecom Open AT 3.10 sample - Hello World
Fichier Edition Affichage Projet Générer Déboguer Outils Fenêtre ?
Debug
open at
U:\projet\mmi\int2\dpo\open-at\samples\3
Parcourir - Wav...le - Hello World
Explorateur de solutions - Hello_...
Solution 'Hello_World' (2 projet(s))
Hello_World
Hello_World_rte
Références
Source Files
Header Files
Open-AT ADL Interface Files
hello_world.html
Index
Rechercher :
Filtre :
(aucun filtre)
Prêt

OPEN AT

wavecom
http://www.wavecom.com/forum

Release version:
3.10

Sample - Hello World

- Changes
- Application Description

Changes

- V2.00 : Initial revision
- V3.00 : No changes

Application Description

This application starts a 1s cyclic timer on its entry point. On each timer expiry, it will display a debug trace in Target Monitoring Tool, and an unsolicited string on the serial link.

Updated : Thu Sep 8 17:46:26 CEST 2005

'' operator
' (caractère délimiteur de commentaire)
-- (comment)
-- (opérateur)
-- operator
postfix decrement operators
postfix operators
prefix decrement operators
prefix operators
- (minus sign in Books Online)
- (negative)

Figure 36: Application sample HTML help file display

5.6.4 Building an application for Target Mode

To build the application for target mode, the target WISMO Target configuration has to be selected as the active one, from the Build toolbar:

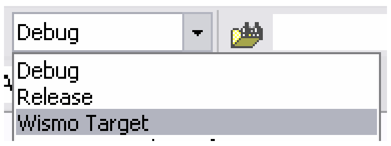


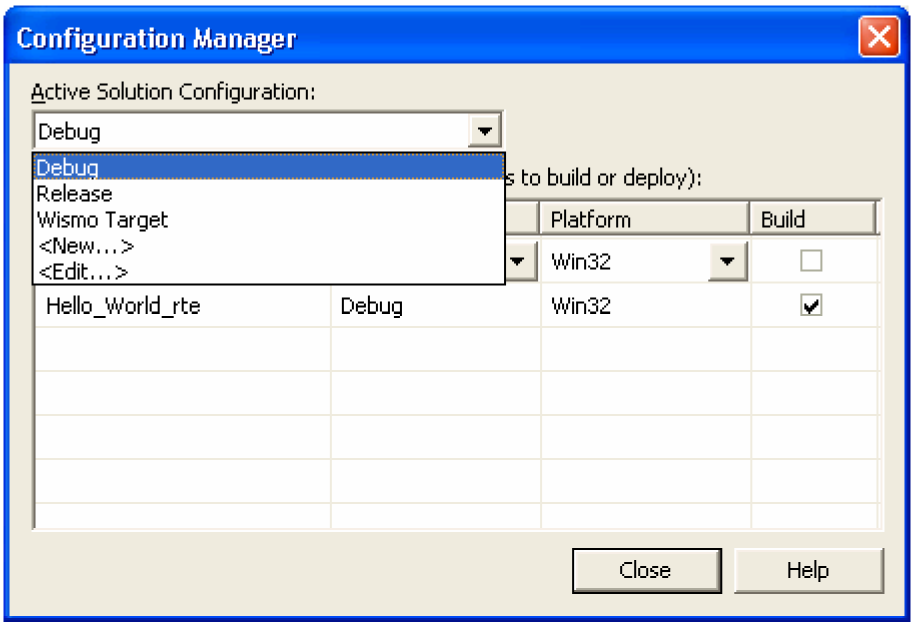
Figure 37: Select the WISMO Target configuration

Then, the project may be built by launching the Build Solution command, from the Build Menu, the contextual menu, or with the Ctrl+Shift+B key combination. The compilation results will be displayed in the output window.

Error messages which may be displayed by the Software Generation Toolkit are listed in the wmmake script description.

5.6.5 Building application for remote mode

To build the application for remote mode, the configuration has to be set. Both Release and Debug mode are available, but Debug mode should be chosen in order to perform step-by-step debugging in the program. The configuration may be selected from the Build->Configuration Manager... menu:



		Platform	Build
Hello_World_rte	Debug	Win32	✓

Figure 38: Select the configuration – way 1

Or from the Build toolbar:

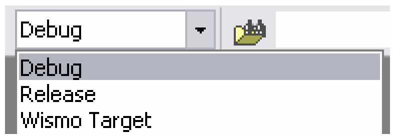
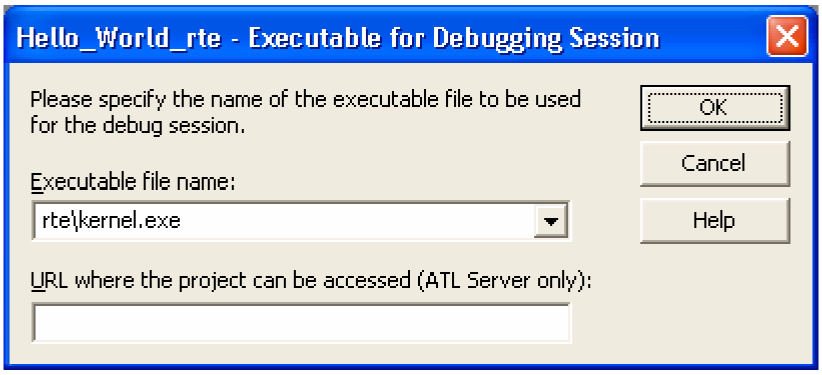


Figure 39: Select the configuration – way 2

Then, the project may be built by launching the Build Solution command, from the Build Menu, the contextual menu, or with the Ctrl+Shift+B key combination. The compilation results will be displayed in the output window.

5.6.6 Starting the Remote Task Environment

The first time the RTE project is executed (by the Start command / F5 key), the environment will prompt for the debug executable file.



Hello_World_rte - Executable for Debugging Session

Please specify the name of the executable file to be used for the debug session.

Executable file name:

rte\kernel.exe

URL where the project can be accessed (ATL Server only):

Figure 40: Provide RTE kernel file name

The file to provide in order to start debug the application is "**rte\kernel.exe**". Further debug sessions (for this project) will not prompt again for this executable name. If a new project is created, or if the current project is updated with the Open AT® Project Wizard, the kernel executable name will be prompted again.

6 Development ToolKit

6.1.1 Overview

The Development ToolKit is a set of 4 tools running under Windows, designed and supplied by Wavecom.

The Development ToolKit environment is presented in Figure 41.

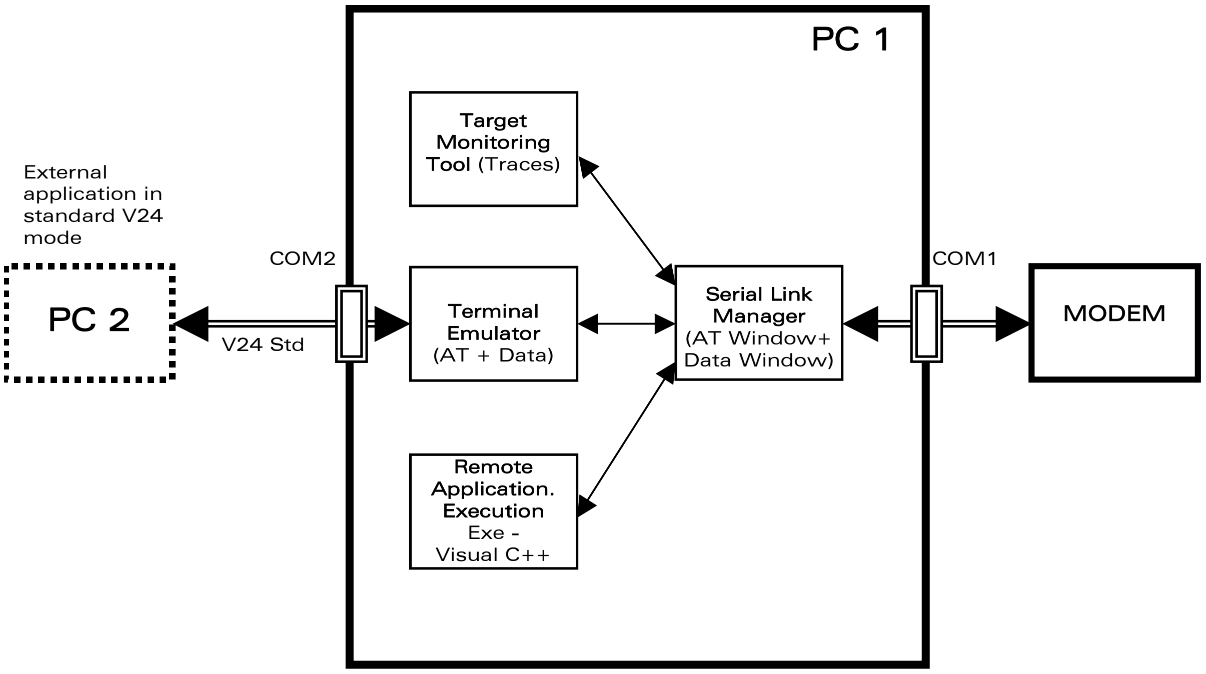

PC 1
Target Monitoring Tool (Traces)
External application in standard V24 mode
COM2
COM1
PC 2
V24 Std
Terminal Emulator (AT + Data)
Serial Link Manager (AT Window+ Data Window)
MODEM
Remote Application. Execution Exe - Visual C++

Figure 41: The Development ToolKit Environment

These tools are described here-below:

6.1.2 Serial Link Manager

6.1.2.1 Description

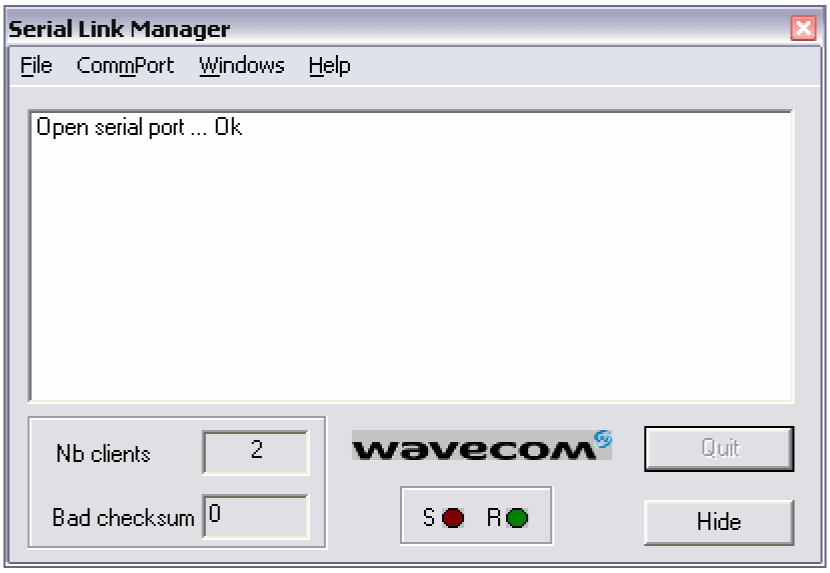
The **Serial Link Manager** uses the PC serial link (COM1) to communicate with Wavecom products. It behaves like a server between the different applications.

The Serial Link Manager dispatches data received from the serial link to the Wavecom PC applications (Target Monitoring Tool, Terminal Emulator and Remote Task Environment). It also multiplexes the frames sent by these applications and forwards them to the product.

The other serial port (COM2) can be used to connect another PC, using the serial link in nominal mode (cf. Terminal emulator description).

6.1.2.2 Getting Started

The Serial Link Manager is automatically launched as soon as the **Target Monitoring Tool**, the **Terminal Emulator** or the **Remote Application** starts: the serial port is opened and the Serial Link Manager icon ( ) is displayed on the task bar. Double-clicking on this icon will launch the Serial Link Manager dialog box.



Open serial port ... Ok

Figure 42: Serial Link Manager window

With this Dialog Box, the user can manage the Serial Port (with the help of the "**CommPort**" menu: open/close the port, set the port configuration). It can be closed with the "**Window**" -> "**Hide Tray**" command. Refer to the Target Monitoring Tool help for more information on the Serial Link Manager.

<u>Note:</u>

The Serial Port configuration used in the Serial Link Manager will be also used in the other programs of the Development ToolKit.

6.1.2.3 Playing back backtrace buffer files

If an application has used the ADL Error Service in order to retrieve the stored backtraces in the product, the obtained binary files may be plaid back in the Serial Link Manager following the steps below:

- In the Target Monitoring Tool, select the current application workspace (cf the Target Monitoring Tool description) ;
- Close the serial port (CommPort -> Close command) ;
- Open the re-play frames window (File -> Play Log File command) ;

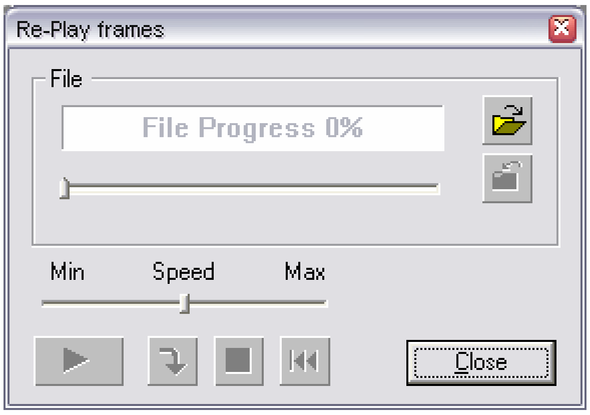


Figure 43: Re-play frames window

- Use the Open ( ) button to select the binary file to play ;
- Hit the Play ( ) button to start the backtrace buffer play-back process.
- Backtraces should now appear in the Target Monitoring Tool.

6.1.3 Target Monitoring Tool

6.1.3.1 Description

The **Target Monitoring Tool** is used to display the Embedded application trace messages (from the Target or the Remote Application).

6.1.3.2 Getting Started

In order to use the **Target Monitoring Tool**, please follow these steps:

1) Download an Embedded OS to the target, using one of the provided samples,
2) Make sure the COM1 port is not used by another application,
3) Start the **Target Monitoring Tool** application from the Windows Start menu,
4) Check the PC serial link settings:
   Click "**Settings**" in the "**Preferences**" menu,

   Select the "**Com**" tab,

   Check default mode parameters:

   Baud rate: 115200

   Data bits: 8

   Parity: none

   Stop bits: 1

<u>Notes:</u>

- The serial port settings can be changed using either the Target Monitoring Tool or the Windows Serial Link Manager.
- The serial port speed can be detected by the **Commands->Auto Detect** command of the Target Monitoring Tool.

5) Check the communication with the Target:
   Click the "**Terminal Emulator**" button in the Target Monitoring Tool toolbar, and select the "**AT1**" window.

   Type AT and press `<Enter>`:

   A blue Ok response must appear. Otherwise, select "**Init Target**" from the "**Commands**" menu of the **Target Monitoring Tool** and type AT again,

6) Close the **Terminal Emulator** window.

6.1.3.3 Get Traces Sent by the Wavecom Target

1) Select "**Open**" from the "**Traces**" menu,
2) Select "**Get Information about Target**" from the "**Commands**" menu,
3) Select "**Set and Request to the Target...**" from the "**Commands**" menu,
   Select **CUS** from the **Parameter** list,

   Click on all levels from 0 to 31 in **Bitmap**,

   Click on the "**Send Level**" button,

   Click on the "**Close**" button,
4) The Trace window then displays the traces sent by the Wavecom product.

Notes:

- The traces are strings sent by the embedded application to the Target Monitoring Tool thanks to the `TRACE()` & `DUMP()` functions. This function takes the display level, and the string to display as parameters. See the ADL User Guide for more information (See Ref. II).
- If the serial link speed is set to 9600 bps, some traces may not appear (lost). The user should, to avoid this, use the speed of 115200 bps on the serial link (see the "**AT+IPR**" command in the AT Commands Interface Guide, Ref. III).
- In the case of an embedded application using the FCM API, the traces flow from the Target Monitoring Tool may conflict with the Terminal Emulator data block flow: some of the data or of the traces may be not displayed. When sending many data on an IO flow, the user should limit the number of traces requested to the Target Monitoring Tool.

6.1.3.4 Select the Current Embedded Application Workspace

When an embedded application binary file is generated, a workspace file MyProject.wks is also produced.

Use the **File->Open Workspace...** menu.

Select the .wks file the **out/tmt** directory of the current downloaded embedded application (in **arm** or **gcc** directory, according to the used compiler).

The Target Monitoring Tool does not need to be restarted.

The Target Monitoring Tool can use this file to display the function's names when the software crashes and leads to a BackTrace. (for more information on software security, see the ADL User Guide for more information, Ref.II)

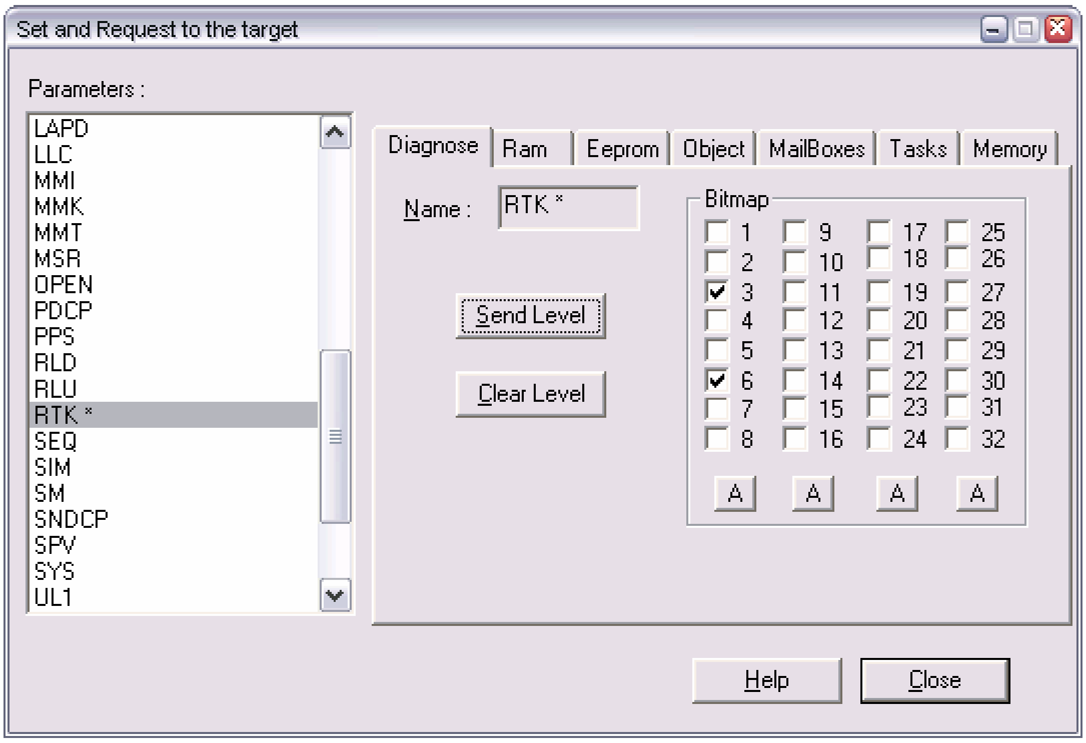
The "BackTraces" log can be retrieved by using the Commands->Get Debug Info->Request EEPROM logged errors menu.

The workspace defines also a diagnose tip file, usable to display tooltip windows according to the current downloaded application, during the selection of trace levels. This file is named DiagnoseTips.ini and is copied in the **out/tmt/config** directory. The user should update this file with its own tip labels to use them with the target monitoring tool.

6.1.3.5 Heap Memory Blocks owners list

At any time, the Target Monitoring Tool may be used to retrieve a list of the currently allocated Heap Memory blocks owners.

1. Enable the level 6 & 3 in the RTK trace element,



Bitmap: [✓ 3] [✓ 6] checked; 1, 2, 4, 5, 7–32 unchecked.

Figure 44:Set and Request to the target window

2. Launch the "Request RTK Status" command from the menu.

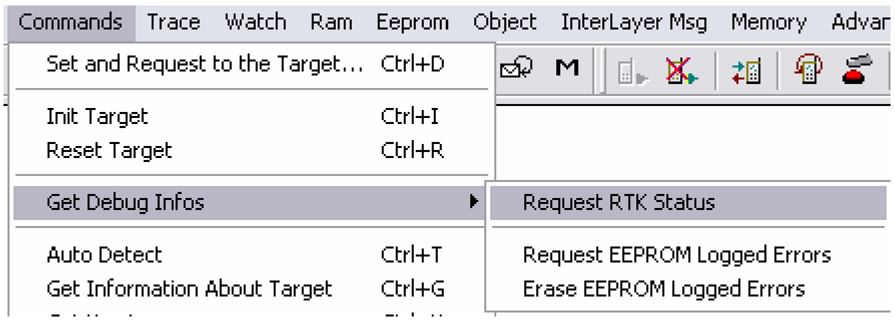


Figure 45: Request RTK status command

Each time this command is used, the current heap memory block is then displayed in the trace window with the following fields:

- S: block status
  (A is "allocated", F is "free")
- Size: block size in bytes
- Link: reserved
- Task/It: Task or interruption context identifier where the block was allocated
- HHMMSS: block allocation time stamp
- Address: block memory address
- Caller: name of the function which has allocated the block (please note that function name will be correctly displayed only if the application workspace has been loaded in the Target Monitoring Tool)

6.1.4 Terminal Emulator

6.1.4.1 Description

The **Terminal Emulator** is an AT command terminal. It is used by the User to send and receive string commands to or from a customer task (on the Target or in Remote mode). It can use either the serial link Nominal mode (the AT window is then open) or the Wavecom Debug mode.

The **Terminal Emulator** connects an External application in standard V24 mode. It transforms a data flow from standard mode to debug mode. An External application can also get information in nominal mode.

6.1.4.2 Getting Started

If the **Target Monitoring Tool** is running, start the **Terminal Emulator** using the corresponding button in the toolbar (check "**Toolbar**" in "**View**" menu), or "**AT cmd terminal**" from the "**Tools**" menu.

You can also launch the **Terminal Emulator** using the shortcut from the Windows Start menu.

The Terminal Emulator provides two windows: one for AT commands and one for the V24 Data flow. Refer to the Terminal Emulator "**help**" menu for more information.

Note: in case of an embedded application using the FCM API, the target should be initialized in Debug Mode ("**Commands**" -> "**Init Target**") before any FCM operation; otherwise the Terminal Emulator and the Target Monitoring Tool will not run properly. Moreover, if the external serial port (COM2) is used, the "**External Com**" -> "**Open…**" command must be used before any FCM operation.

Note:

If there is a data call, or if the serial link is switched into DATA mode with the FCM API, the "+++" sequence does not work in the Terminal Emulator.

6.1.5 Remote Application Execution

This tool is a set of libraries with which the User application can be executed on a PC by means of Visual C++ ® while communicating with the target through a serial link. This tool is detailed in the next paragraph.

7 Remote Task Environment

7.1 Basic Concepts

An Embedded application can be executed from a PC, while communicating with the rest of the Target software through a serial link.

An Embedded application is implemented in a specific task (customer task). This task can be executed on a PC, using the Windows version of RTK. In this case, the customer task is deactivated on the target.

This concept is described in Figure 46: Project Architecture.

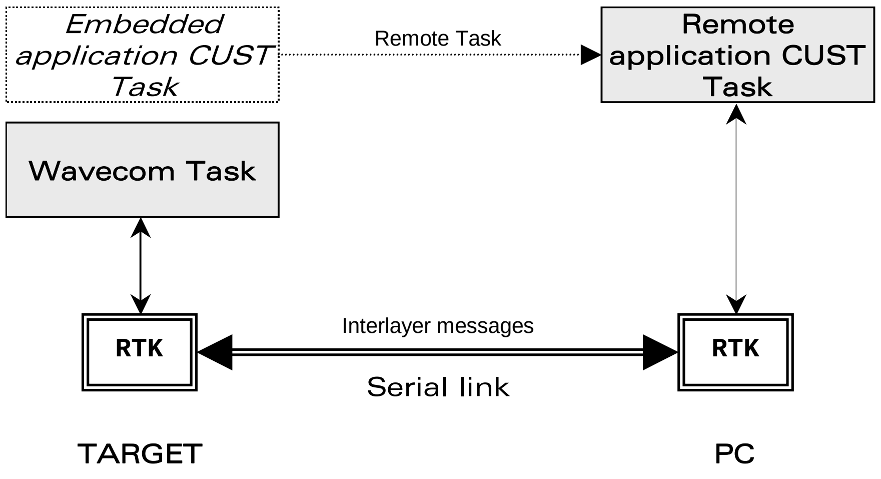


Figure 46: Project Architecture

Without changing the code, an Embedded Application becomes a Remote Application after recompilation.

The Remote process (development, compilation, link, execution) operates in one of the supported development environments.

7.2 Project Creation

Please refer to the Chapter "Building Open AT® applications" for more details on how to create and build projects in remote mode.

7.3 Remote Application Execution

7.3.1 Serial Port Configuration

Using the **Serial Link Manager**, the user has to check that the serial port speed is set to the same value as the Target port speed.

The **Auto Detect** command from the **Target Monitoring Tool** can be used to retrieve this speed. In order to get better results, the speed should be kept at **115200 bauds** (default value).

When connecting the target to the computer through the serial link, the connection may not work, because the Target and the computer serial interfaces may use different baud rates.

Selecting **Auto Detect** will then make the **Target Monitoring Tool** send frames with different successive baud rates. If the target sends them back, it means the target speed and the computer speed match. The **Target Monitoring Tool** then stops scanning.

An **Auto Detect** sequence typically lasts five to ten seconds. The **Target Monitoring Tool** may seem slower during this time.

The Terminal Emulator has to be launched before starting the remote application, to check whether the communication with the target is correct or not.

7.3.2 Remote Application Launch

To launch the user interface, from the Development Environment menu, select the following command:

Debug toolbar button (Eclipse)
or
build -> Start Debug -> Go (F5) ***Visual C++ 6***
or
Debug -> Start (F5) ***Visual C++ .NET***

The Development environment then starts in Debug mode, and launches the **Remote Application Execution Tool**.

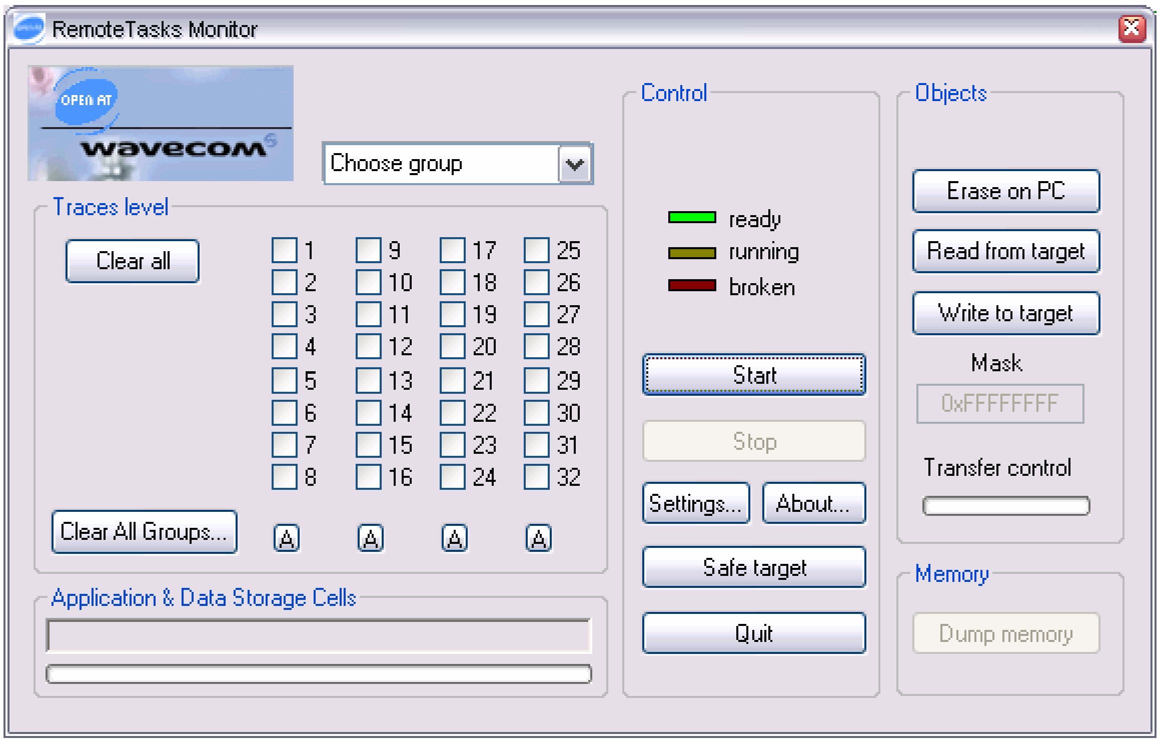


Figure 47: Remote Application Controller

The remote process is started as soon as the **Start** button is clicked from the Remote Application Controller (see Figure 47 – step 1, then step 2).

The Initialization sequence is described below:

1. Initialization of the serial link (in Serial Link Manager mode),
2. Target configuration,
3. Target RTK Re-initialization,
4. Remote Application Execution creation and activation.
5. The **Target Monitoring Tool** should then display the selected trace levels, showing that the application is running correctly (in Figure 47, the step 3 is reached)

Note:

During step-by-step execution, trace messages are not displayed at once by means of the **Target Monitoring Tool**. They are only displayed after an RTK scheduling.

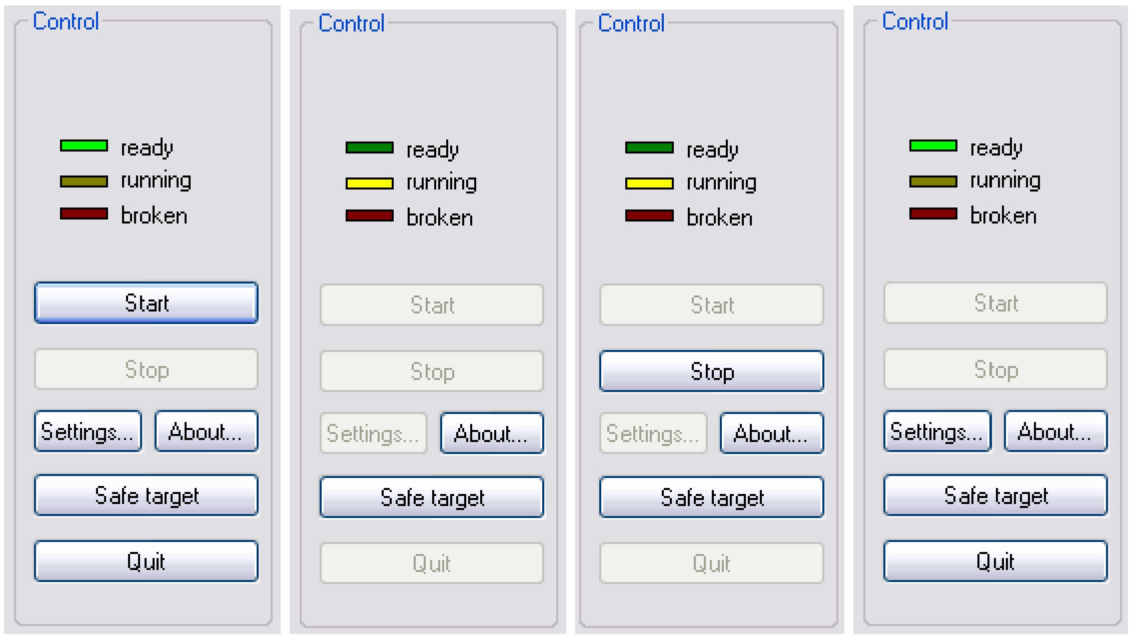


Figure 48: RTE Monitor control steps (1 to 4)

To stop the remote application execution, the Stop button has to be clicked (Figure 47, step 4). Then, a Flash Object synchronization may be done (if wished), and the RTE monitor may be closed by the Quit button.

If the RTE application does not end by this way (Stop button, and then Quit button), due to an exception during the process, or by using the "Stop Debug" option of the Development Environment, any target Open AT® application will still be disabled.

In order to restore the target application normal running mode, the RTE monitor has to be launched again, and the Safe Target button has to be clicked.

7.3.3 Traces Configuration

Embedded application trace messages are displayed using the **Target Monitoring Tool**, by the way of a trace window opened from the **Traces** menu.

The Target application traces then appear **in black** and the Remote Application traces appear **in blue**.

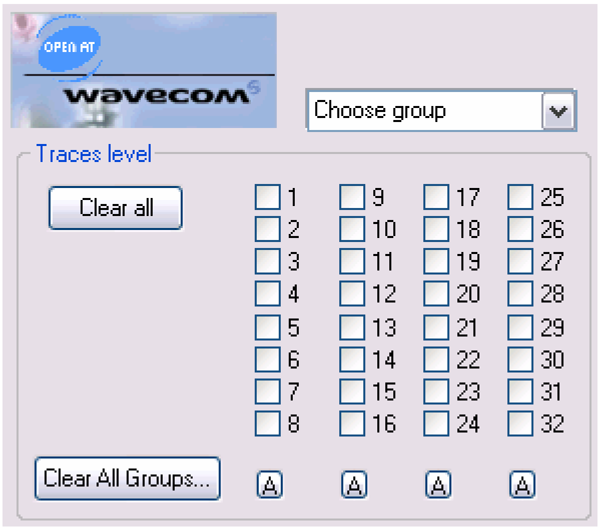


Figure 49: Trace Level Selection

Each trace instruction corresponds to a trace level. The user has to select this level using the Remote Application Controller before using the **Target Monitoring Tool** to display the traces.

Firstly, the Open AT® task has to be selected (ADL applications use always CUS4 task).

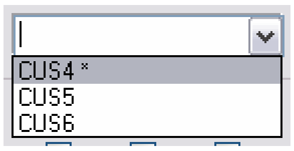


Figure 50: Select Open AT® task

Then each trace level may be selected separately, or a whole column by pressing a A button. Trace levels may be also cleared separately, or together, for the selected task (Clear all button), or for all tasks (Clear All Groups... button).

Enabled trace levels may be changed at any time, during or out of the application execution. Changes will immediately be applied in the **Target Monitoring Tool**.

7.3.4 Data Flash Objects synchronization

Data Flash Objects are entities in which the User application can save data such as addresses, phone numbers, etc. An object is referred to by a 16-bit hexadecimal identifier (this is NOT a physical address).

In remote application, these Data Flash Objects are emulated through physical files, located in the **[project path]\rte\objects\** directory.

In order to have the same Data Flash Objects configuration as on the Target, the user has to launch a read synchronization (Read from target button) before starting the Remote Application.

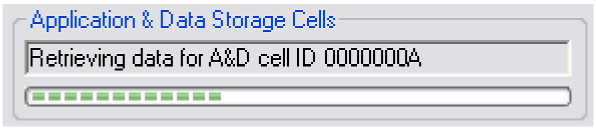
Note:
Because both size and number of Data Flash Objects present on the Target, read synchronization may take up to one or two minutes.

After a remote application execution, objects written on PC side may be synchronized to target using the Write to target button.

Finally, synchronized objects on PC side may be erased using the Erase on PC button.

7.3.5 Application & Data Storage Cells synchronization

Application & Data storage cells are fully available within RTE mode. Writing to cells process is completely transparent. On "GetInfo" (reading) process, cells are synchronized and shadowed on PC each time the data are updated. The synchronization process is monitored by the corresponding progress bar in the RTE monitor interface.



Retrieving data for A&D cell ID 0000000A

Figure 51: A&D cell synchronization

8 Command-line scripts

Open AT® Project Wizard and development environment are graphical interfaces which use the following command-line scripts to respectively create and build Open AT® application projects.

These scripts should also be invoked from the Open AT® IDE Command Line. This tool should be started:

- the Start Menu shortcut (Wavecom\Open AT IDE\Open AT IDE Command Line).
- from any Open AT® project directory **`LoadIde.bat`** file.

The Open AT® Command Line is based on a light Cygwin bash shell: common bash and Linux commands should be used in this environment.

Warnings:

- if any other Cygwin version is installed on the system, it is not recommended to use this Cygwin version and the Open AT® IDE Command Line at the same time.
- Moreover, it is not recommended to use an IDE and the Open AT® IDE Command Line, or several Open AT® IDE Command Line windows at the same time.

8.1 Create Open AT® projects: wmnew script

The wmnew script should be used in order to create/update Open AT® application projects. This script has to be called from the project root directory. The available options are described below.

8.1.1 Script control

Option	Use
-h	Displays help on wmnew script and exits.
-v	Verbose mode (displays information on each project generation step).
-c	Cleans previously generated workspace files before process.
-i	Ignores the current directory wmnew.opt file if any (by default, this file content is added to the command line options).
-ns	Does not save the currently required options in the wmnew.opt file (by default, command line options are saved in a wmnew.opt file in the current directory).
-s	Synchronizes the enabled workspaces in the Open AT® Settings application (by default, the only generated/updated file is the .scs file)

8.1.2 Script modes

Option	Use
[default]	Updates the existing project in the current directory with existing options.
-n	Creates a new project (similar as –sample New_Project option)
-sample SAMPLE	Creates a copy of the required SAMPLE in the local directory ; if the sample depends on some sample libraries, these are also copied in a Libraries directory (located in the same directory as the project one, for example –sample /Hello_World)

8.1.3 Supplied library options

Option	Use
-samplelib SMPLIB	Adds dependencies to the SMPLIB sample library in the current project. (copy the SMPLIB sample library files in the ../Libraries/SMPLIB directory, and then similar to: -inc ../Libraries/SMPLIB/itf -rootlib ../Libraries/SMPLIB -libpref SMPLIB)
-plugin OLIB	Adds dependencies to OLIB plug-in. OLIB has to be one of the installed plug-ins on the system.

8.1.4 Project interface options

Option	Use
-adl	Uses the ADL interface (default).
-basic	Uses the Basic interface.

8.1.5 Project type options

Option	Use
-app	Defines an application project (default).
-lib	Defines a library project.

8.1.6 Project global options

Option	Use
-name NAME	Defines the project name (for new projects only ; existing and sample projects names are automatically defined).
-os OS_PATH	Uses the OS_PATH value as the Open AT® OS version for the current project, instead of the default value.
-fw FW_PATH	Uses the FW_PATH value as the Open AT® Firmware version for the current project, instead of the default value.
-pr PR_PATH	Uses the PR_PATH value as the plug-ins root directory for the current project, instead of the default one.
-flag FLAG	Adds the FLAG compilation flag to the project flags list.
-gssf FACTOR	Sets the GCC stack size factor to the FACTOR value (default 2). This value is the factor with which the source code stack size has to be multiplied for the GCC compiler (since GCC needs more room in the call stack on run time than the ARM compiler).
-optionfile FILE	Add SGT options FILE content to compilation flags list. A SGT options file is a text makefile containing a PP_OPT_COMMON variable (all compilation flags set in this PP_OPT_COMMON variable will be added to the compilation flags list).
-gts USED_GTS	Selects USED_GTS as the project's default Generation Tools Suite. Allowed values are: • GCC (for ARM GCC compiler -- default). • ADS (for ARM Developer suite compatible compiler).
-mem USED_MEM	Selects USED_MEM as the project's default memory type. Allowed values depend on the selected Open AT® Firmware version.
-ide USED_IDE	Selects USED_IDE as the project's IDE, for which the workspace has to be generated. Allowed values are: • cdtproject for Eclipse. • dsp for Microsoft Visual C++® 6.0. • vcproj for Microsoft Visual C++® 2003. • _2002.vcproj for Microsoft Visual C++® 2002.
-dlv PATH	After build process, on successful build, copies output (library or binary) to PATH sub-directory.
-itfdlv PATH	After a successful build process, copies interface PATH content to the delivery directory (-dlv option mandatory).

8.1.7 Project source paths options

Option	Use
-src SRCPATH	Adds SRCPATH to the source paths list (always contains the ./src current sub-directory by default). All .c files in these directories will be considered as a source file for the project.
-inc INCPATH	Adds INCPATH to the include paths list (always contains the ./inc & ./itf current sub-directories by default). All .h files in these directories will be considered as an header file for the project.
-noppinc INCPATH	Adds INCPATH to the standard include paths list; usable to supply paths for standard include statements (E.g. **`#include <header.h>`**). Header files in these directories will not be copied in the ./{compiler}/out directory on compilation phase.
-path FULLPATH	Similar to: -src FULLPATH/src -inc FULLPATH/inc -inc FULLPATH/itf
-copy SRC@DEST	Copies SRC file or directory content to the local DEST sub-directory (the destination directory tree is created if needed)

8.1.8 Project object paths & names options

Option	Use
-extobj OBJPATH	Adds the OBJPATH path to the absolute objects paths list. Object files will be browsed exactly in the provided directory.
-rootobj OBJPATH	Adds the OBJPATH path to the root objects paths list. Object files will be browsed in the sub-directory which matches the currently used compiler (for example arm/out for ARM compiler, gcc/out for GCC compiler, etc...)
-objname OBJNAME	Adds the OBJNAME template to the real object names filter list. Object files will be browsed while trying to match exactly the provided template (including the extension), whatever is the currently used compiler. (for example, the –objname "m*.o" option will browse for all object file names beginning with the "m" letter).
-objpref OBJNAME	Adds the OBJNAME template to the prefixed object names filter list. Object files will be browsed wile trying to match the provided template, according to the currently used compiler. (for example, the –objpref "myobject" option will browse for object file names which match the "arm_myobject.o", "gcc_myobject.o" or "rte_myobject.obj", according to the currently used compiler).

8.1.9 Project library paths & names options

Option	Use
-extlib LIBPATH	Adds the LIBPATH path to the absolute libraries paths list. Library files will be browsed exactly in the provided directory.
-rootlib LIBPATH	Adds the LIBPATH path to the root libraries paths list. Library files will be browsed in the sub-directory which matches the currently used compiler (for example arm/out for ARM compiler, gcc/out for GCC compiler, etc...)
-libname OBJNAME	Adds the LIBNAME template to the real library names filter list. Library files will be browsed while trying to match exactly the provided template (including the extension), whatever is the currently used compiler. (for example, the –libname "mylibraries*.lib" option will browse for all library file names beginning with the "mylibraries" template).
-libpref LIBNAME	Adds the LIBNAME template to the prefixed library names filter list. Library files will be browsed while trying to match the provided template, according to the currently used compiler. (for example, the –libpref "mylibrary" option will browse for library file names which match the "arm_mylibrary.lib", "gcc_mylibrary.lib" or "rte_mylibrary.lib", according to the currently used compiler).

8.2 Building Open AT® projects: wmmake script

The `wmmake` script should be used in order to build an Open AT® project. This script has to be called from the project root directory. The script syntax & options are defined below:

```
wmmake [<Project>] [options…]
```

where `<Project>` is the project name, without any extension or with the `.scs` or `.mak` one.

If the `<Project>` name is not provided, `wmmake` will automatically use the current directory project file.

8.2.1 Script control

Option	Use
-h	Displays help on `wmmake` script and exits.
-n	Do not browse for library projects dependencies. By default, for all libraries with a prefixed name (`-libpref` option in `wmnew` script) and a root path (`-rootlib` option in `wmnew` script), the `wmmake` script (re-)builds also the corresponding library project.
-v	Do not check the IDE version versus the current project one. By default, if IDE & current project versions do not match, a warning message is displayed, and the user is prompted to continue the build process or not.
-zip	If successful, compress build output to a zip file.
-D[RESOURCEPATH]	Generate HTML documentation for the current project. If RESOURCEPATH directory is supplied, copy graphical resources to it.

8.2.2 Clean options

Option	Use
all_clean	Cleans the output directory generated files and exits.
-c	Performs a "all_clean" process, and re-launches the build process.

8.2.3 Compiler options

Option	Use
[default]	Builds the project using the compiler defined by the Open AT® Project Wizard.
-X	Builds the project using the ARM X compiler. Please refer to `wmnew` script for more information about allowed compiler values.

8.2.4 Memory options

Option	Use
[default]	Builds the project using the memory type defined by the Open AT® Project Wizard.
-X	Builds the project for X type memory products; please refer to the ADL User Guide for more information about the available memory types (See Ref. II).

8.2.5 Debug Options

Please see the ADL Trace Service description for more information about the debug flags.

Option	Use
-debug	Defines the __DEBUG_APP__ flag at compilation time (default value).
-fulldebug	Defines both the __DEBUG_APP__ & __DEBUG_FULL__ flags at compilation time.
-release	Does not define any debug flag.

8.3 Check current Open AT® IDE settings: wmcheck script

The wmcheck script should be used in order to display the current environment configuration, and detect what is going wrong if other tools (as the Open AT® project wizard, or the wmmake script) do not run properly. The script syntax & options are defined below:

```
wmcheck [<output file>]
```

8.3.1 Script control

Option	Use
<output file>	Name of the text file to which the configuration dump is to be copied. This text file should be used to report any configuration problem.

8.3.2 Configuration dump sample

When running the wmcheck script, a configuration dump similar to the one below should be displayed.

```
Open AT IDE parameters list
---------------------------
[WM OAT IDE]: "C:/OpenAT/IDE/IDE/X.XX.XX"
[WM OAT IDE WIN]: "C:/OpenAT/IDE/IDE/X.XX.XX"
[WM OAT IDE ENV]: "cygwin"
[WM OAT IDE GTS GCC]: "C:/OpenAT/IDE/GCC/X.X.X.X"
[WM OAT IDE GTS ARM]: "M:/ads/ADS V12"
[WM OAT IDE GTS MINGW]: "C:/OpenAT/IDE/MINGW/X.X.X.X"
[WM OAT IDE OS DEFAULT]: "C:/OpenAT/OS/4.10.XX"
[WM OAT IDE FW DEFAULT]: "C:/OpenAT/Firmware/B61x"
[WM OAT IDE PLUGINS]: "C:/OpenAT/Plug-ins"
[WM OAT IDE ECLIPSE]: "C:/OpenAT/IDE/Eclipse/X.X.X"
[SGT DIR]: "/cygdrive/C/OpenAT/IDE/IDE/X.XX.XX/sgt"
[SGT_SCRIPT]: "/cygdrive/C/OpenAT/IDE/IDE/X.XX.XX/sgt/script_sgt"
[SGT_VER]: "v1.2.11oat"
```

```
Current directory project parameters list
-----------------------------------------
[WM OAT PRJ NAME]: "Hello World"
[WM OAT PRJ GTS]: "GCC"
[WM OAT PRJ GTS IDE REFPATH]: "WM OAT IDE GTS GCC"
[WM OAT IDE GTS GCC]: "C:/OpenAT/IDE/GCC/X.X.X.X"
[WM OAT PRJ API]: "ADL"
[WM OAT PRJ MEM]: "H"
[WM OAT PRJ IDE]: "vcproj"
[WM OAT PRJ TYPE]: "bin"
[WM OAT PRJ DELIVERY]: ""
[WM OAT PRJ OBJ DELIVERY]: ""
[WM OAT PRJ ITF DELIVERY]: ""
[WM OAT PRJ PLUGINS]: ""

Current directory project's Firmware parameters list
----------------------------------------------------
[WM OAT PRJ FW]: "C:/OpenAT/Firmware/B61x"
[WM OAT PRJ FW MEM DEFAULT]: "H"
[WM OAT PRJ FW MEM LIST]: "H "
[WM OAT PRJ FW BASEBAND]: "swift"
[WM OAT PRJ FW ROBASE]: "00210000"
[WM OAT PRJ FW RWBASE]: "180C0000"

Current directory project's OS parameters list
----------------------------------------------
[WM OAT PRJ OS]: "C:/OpenAT/OS/4.10.XX"
[WM OAT PRJ OS BASEBAND]: "swift"
[WM OAT PRJ OS GCC STACKFACTORSCRIPT]: "wmgccstackfactor.awk"
[WM OAT PRJ OS RTE EXPORTSCRIPT]:
"C:/OpenAT/IDE/IDE/X.XX.XX/mak/export.def"
[WM OAT PRJ OS API VERSION]: "410"
[WM OAT PRJ OS HEADER FILE]: ""
[WM OAT PRJ OS HEADER AREA]: ""
[WM OAT PRJ OS GTS LIST]: "ADS GCC RTE "
[WM OAT PRJ OS ITF LIST]: "C:/OpenAT/OS/4.10.XX/ADL/itf
C:/OpenAT/OS/4.10.XX/ADL/basic "
[WM_OAT_PRJ_OS_SAMPLESET_LIST]: "C:/OpenAT/OS/4.10.XX/ADL/samples"
```

9 TCPIP debug tool

A script tool is provided with the Open AT® IDE, in order to trace IP packets when using the TCPIP Add-on library.

Note:

IP packets can be only monitored on the GPRS flow, not on the GSM Data flow.

9.1 Enable the traces

In order to display the IP packet in the Target Monitoring Tool, the +ADLDBG command may be used. This command syntax is defined below:

AT+ADLDBG=<FlowId>

Where <FlowId> is the ADL FCM flow ID (in hexadecimal base) to be monitored, in order to display IP packets: please refer to the ADL User Guide (see Ref II) to get the full flow ID list.

E.g.: in order to monitor the IP packets sent and received on the GPRS flow, the AT+ADLDBG=50 command has to be used.

FCM data packets traces are displayed on level 25 of the CUS4 task, so in order to retrieve the IP packet traces, the CUS4 task's 25th level has to enabled in the "Set and request to the target" dialog box of the Target Monitoring Tool.

9.2 Write traces to disk

In order to save retrieved IP packet traces to a file, the user may:

- Either enable the "Saving" checkbox in the Target Monitoring Tool; the file name is defined by the "New File" options of the right-click menu in the Traces window.
- Or copy all the trace lines of the Traces window, and paste them into a new text file, in a text editor.

9.3 TCPDUMP format conversion

From the Cygwin command line, the `wm_trc2tcpdump.awk` script may now be used to convert the traces text file to the TCPDUMP format. The syntax is:

```
Usage: wm_trc2tcpdump.awk <TraceFile> <TcpDumpFile>
```

Where <TraceFile> is the text traces file saved from the Target Monitoring Tool, and <TcpDumpFile> is the TCPDUMP format destination file. This file will be overwritten without any warning.

If the text traces file does contain any IP packets, a warning will be displayed, and the output file will not be written.

9.4 IP packets analysis

Now the TCPDUMP format file may be opened by any software able to read this format. The best known is Ethereal, which may be downloaded free from http://www.ethereal.com URL.