



Product:	CDPEventManager
Product version:	V1.2
Document ID:	PM-CDPEventManager
Doc revision:	A
Written/Appr.:	RE /
Date:	24.10.2008

Industrial Control Design AS



CDPEventManager V1.2

Programmers Manual

The content of this document is confidential information not to be published without the consent of Industrial Control Design AS.

Industrial Control Design AS, www.icd.no, support@icd.no, forum.icd.no

Contents

1. INTRODUCTION.....	3
1.1. About.....	3
1.2. Terms and Definitions.....	3
<hr/>	
2. INSTALLATION.....	5
2.1. Prerequisites.....	5
2.2. Quick setup of a project utilizing the SNMPManager library.....	5
2.3. Modify the project xml files.....	10
2.4. Install on a Controller.....	10

1. Introduction

1.1. About

This document describes how to set up CDPEventManager in a development environment and briefly how to start and run the it.

More detailed description about usage of the CDPEventManager system is described in UM-CDPEventManager.pdf.

Some key features of CDPEventManager:

- CDPEventNode can log events from any custom objects that reports events.
- Any custom objects can report events by implementing the IEventLogger interface.
- CDPEventLogger can log event from several event nodes and provide events from the whole distributed system from a single/redundant point in the system.
- CDPEventSubscriber can subscribe events from e.g. CDPEventLoggers, and users can do whatever they want to do with the event. E.g. store the events in event log to trace events in the system.
- All reported events are guaranteed to be received by the subscriber. (As long as circular event buffer is not overloaded. If overrun occurs is detected by the protocol and alarm is triggered.)
- Event-message protocol takes care of resending if event-message is lost in network

1.2. Terms and Definitions

CDPEventNode

A CDPComponent that stores reported events in a memory database, and also provides the reported events to subscribing subscribers like e.g. CDPEventLoggers.

CDPEventLogger

A CDPComponent that subscribes for events from event providers like e.g. CDPEventNode. CDPEventLogger stores events in lokal memory database, and provides the events to subscribers. The key feature for an event logger is to collect events from several nodes and providing the events for the users from one single point for easy access for users.

CDPEventSubscriber

A CDPComponent that subscribes events from providers like e.g. CDPEventLogger. This component can be used as base component for users. Users can override virtual methods and implement callbacks to do whatever they want to do with the events. E.g. store the events in an event log.

2. Installation

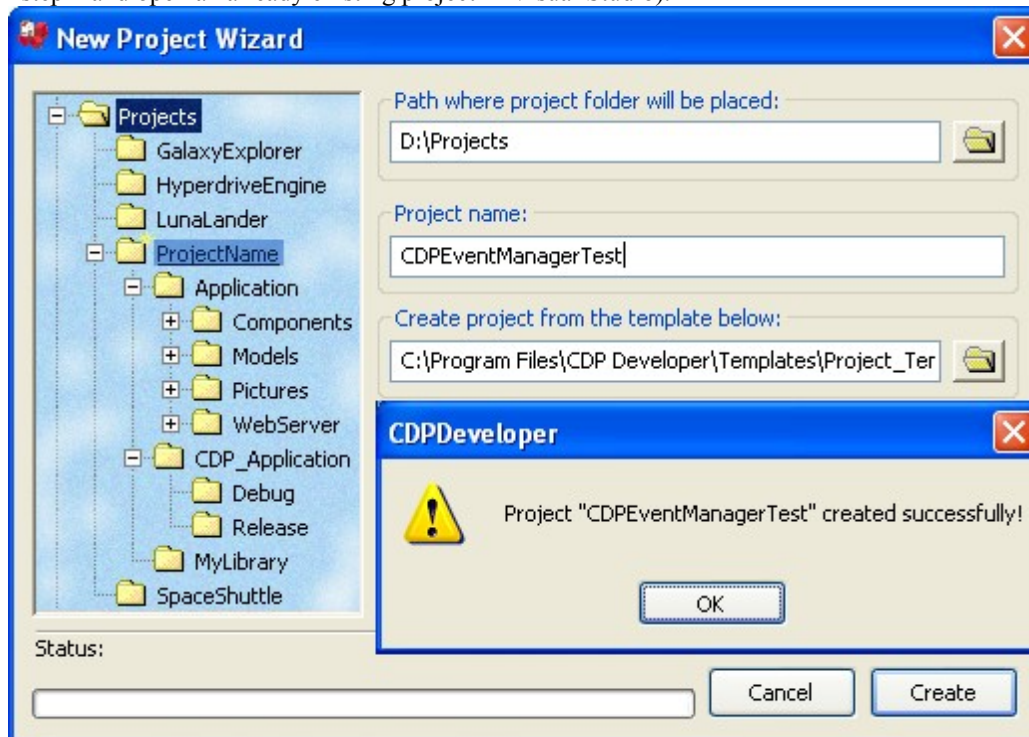
The CDPEventManager can be delivered both as source code and as a separate library which is linked into the application. If delivered as separate library, the CDPEventManagerLib Setup will by default install all files in sub-folders of “CDP Developer”.

2.1. Prerequisites

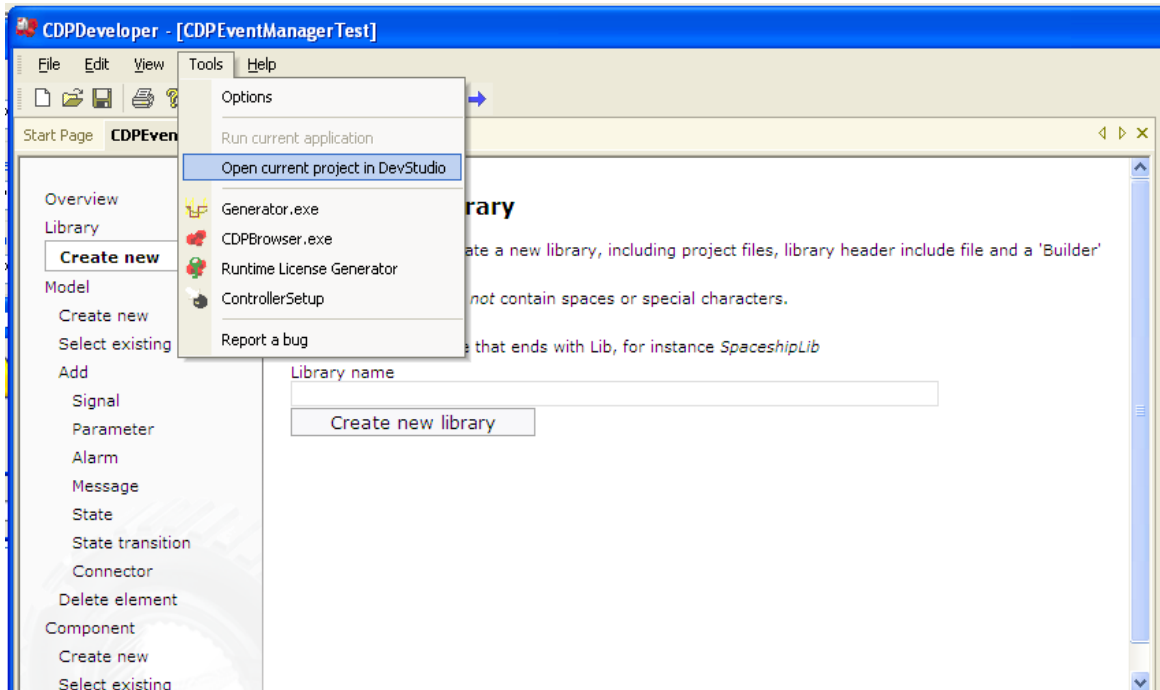
- A valid CDP license
- Familiar with CDP
- CDP version: 2.3.1.0
- OS: “Windows” or “RTOS version 5.11” or “Linux (based on gcc-4.1.1 and glibc-2.3.6 or glibc-2.6.1)”.

2.2. Quick setup of a project utilizing the CDPEventManager library

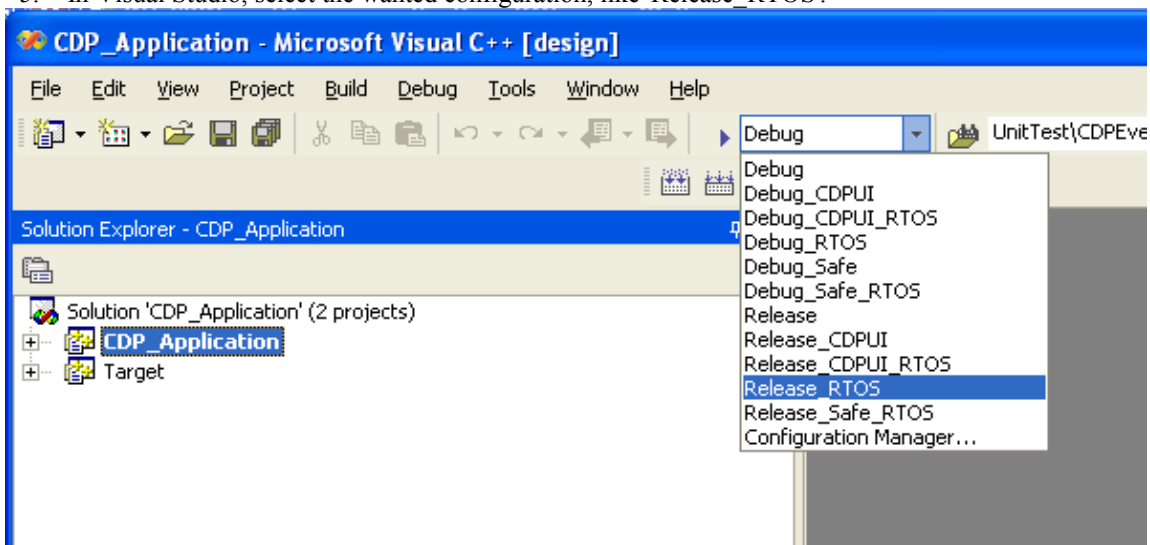
1. Start CDP Developer
2. Make a new project by selecting File->'New Project', type in project name and click Create (or skip to step 4 and open an already existing project in Visual Studio).



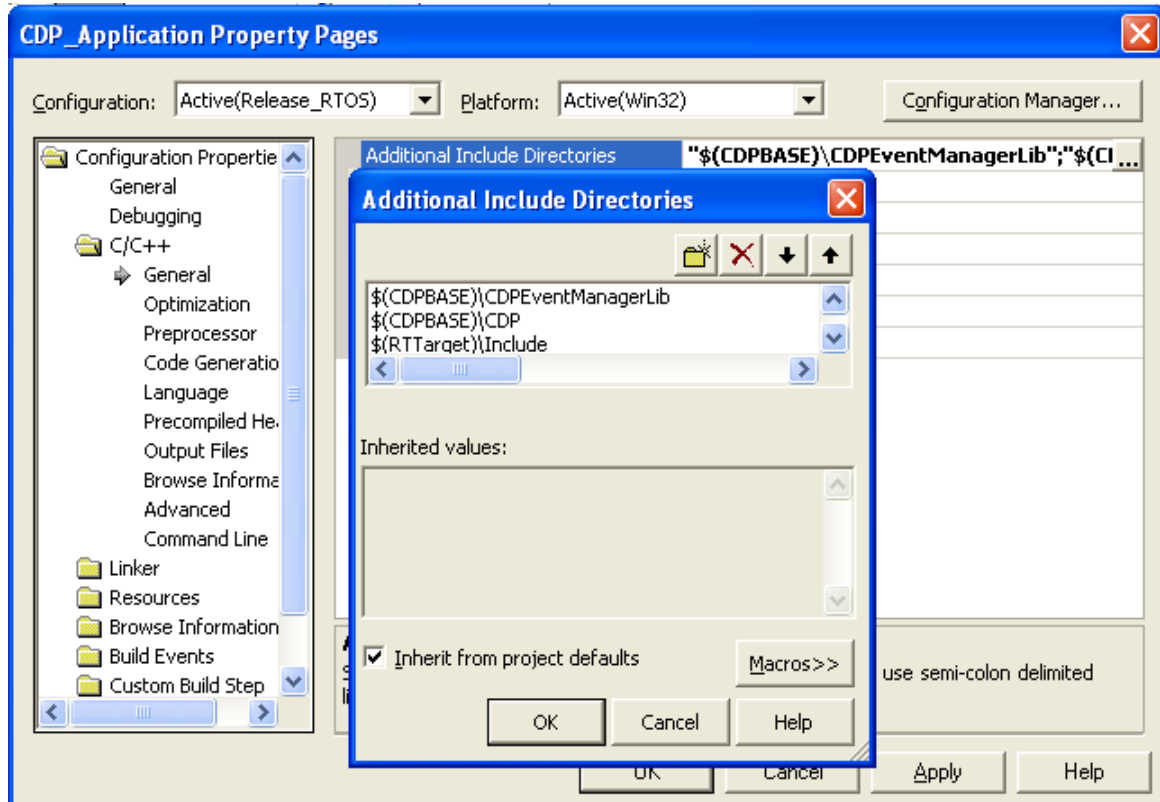
3. Choose Tools->'Open current project in DevStudio':



4. If Visual Studio asks to convert the project, accept this by selecting 'Next'/'Finish'/'Close' until done. Close the conversion report.
5. In Visual Studio, select the wanted configuration, like 'Release_RTOS'.

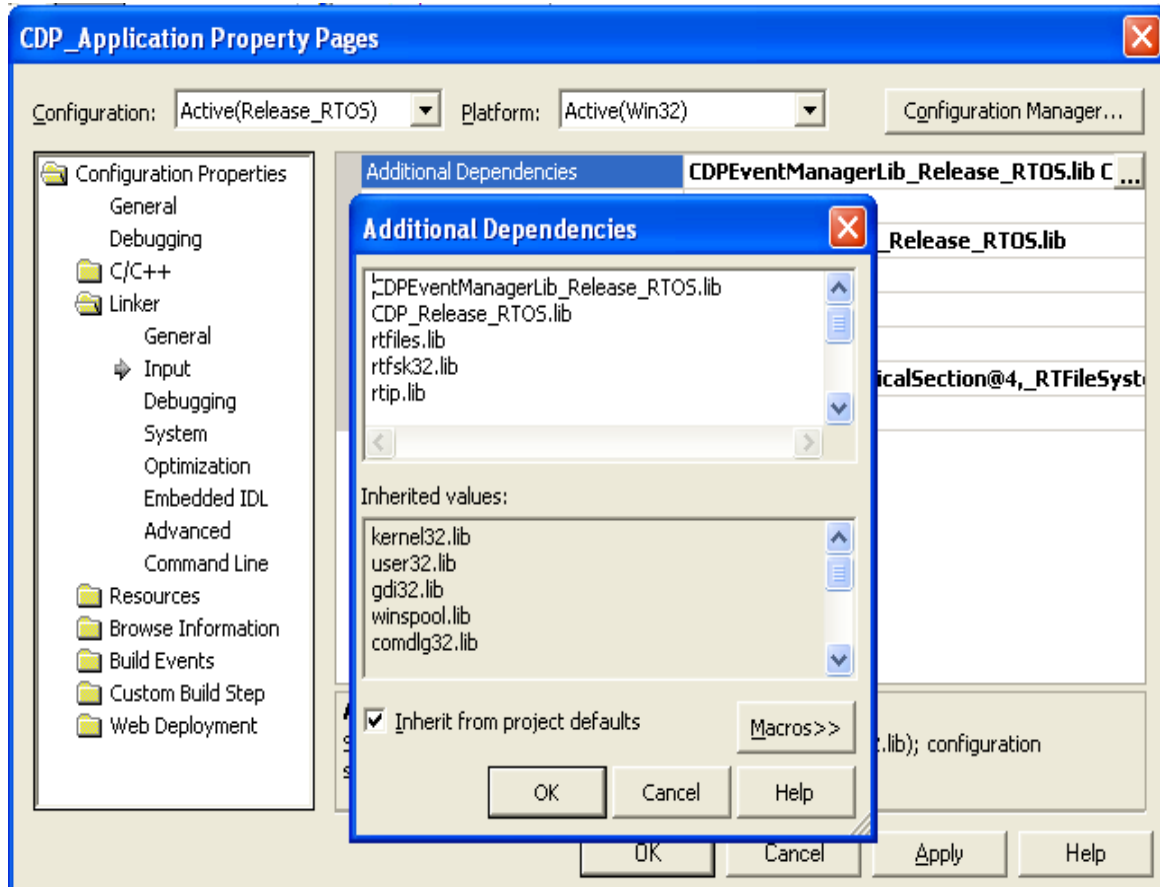


6. Select 'CDP_Application' from the 'Solution Explorer', right-click and select Properties.
7. In C++/Additional Include Directories, make sure it says:
"\$\$(CDPBASE)\CDPEventManagerLib";"\$(CDPBASE)\CDP";"\$(RTTarget)\Include"

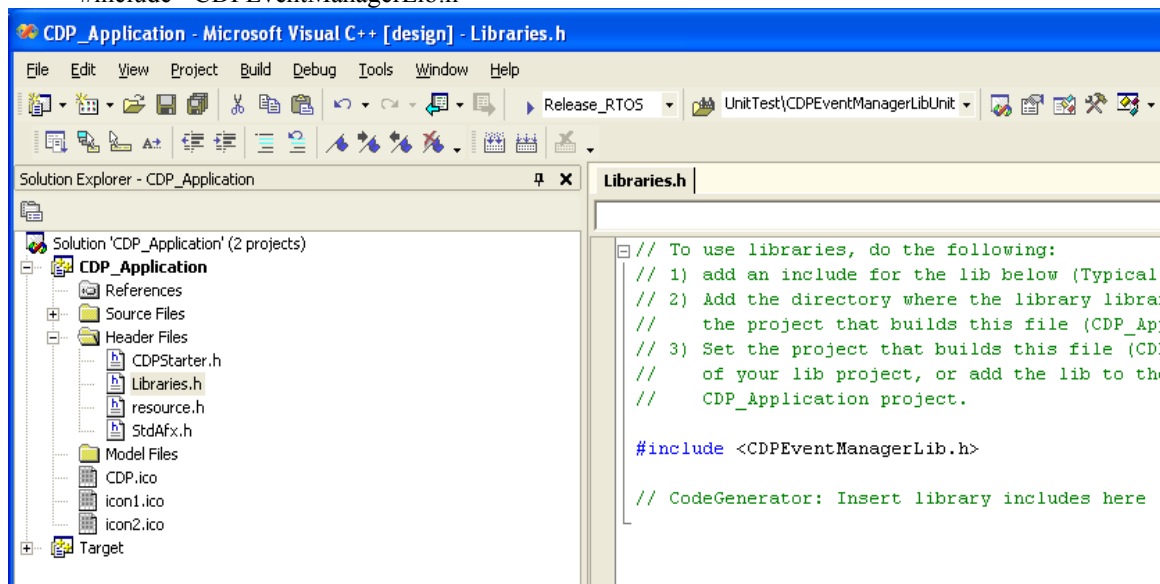


The compiler will look in the directories specified in 'Additional Include Directories' for files that you #include in your .cpp and .h files. If you get an error 'Can not open include file...', then it is most likely caused by a missing include directory, or that the file you #include does not exist.

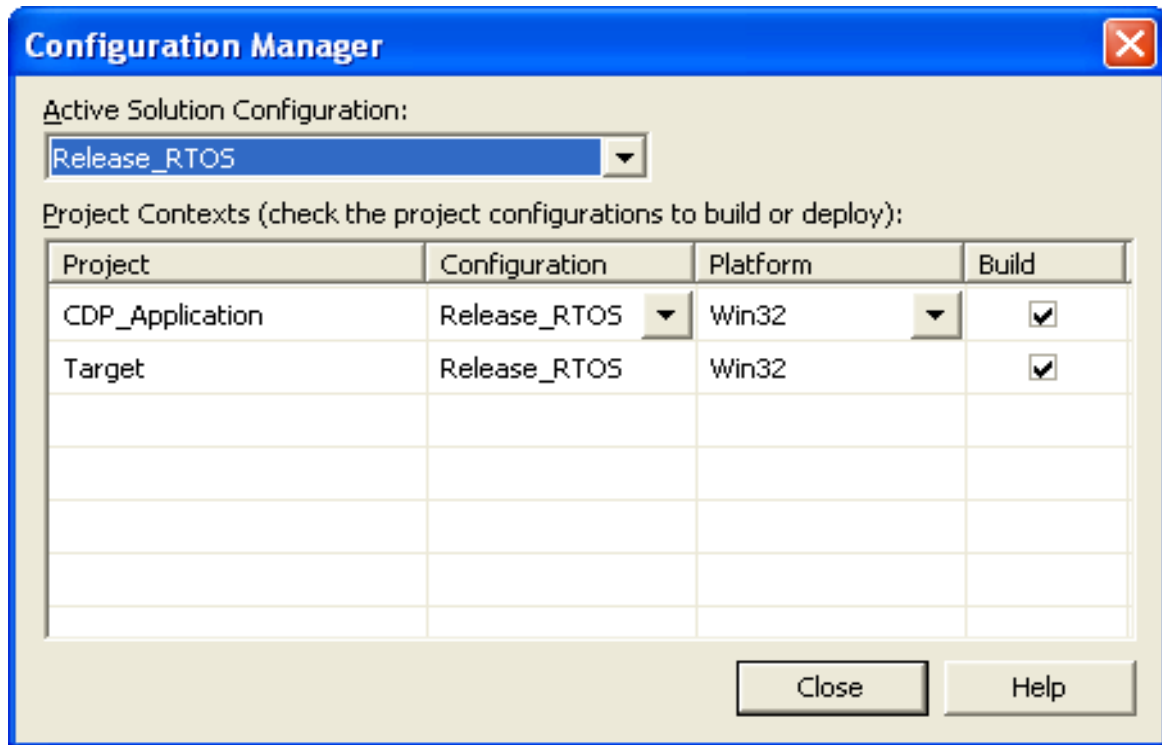
8. In Linker->Input, make sure you list CDPEventManagerLib_Release_RTOS.lib CDP_Release_RTOS.lib in addition to all the on-time rtos32 libraries you need. This will ensure that the linker finds all the functions that are referenced in the code.



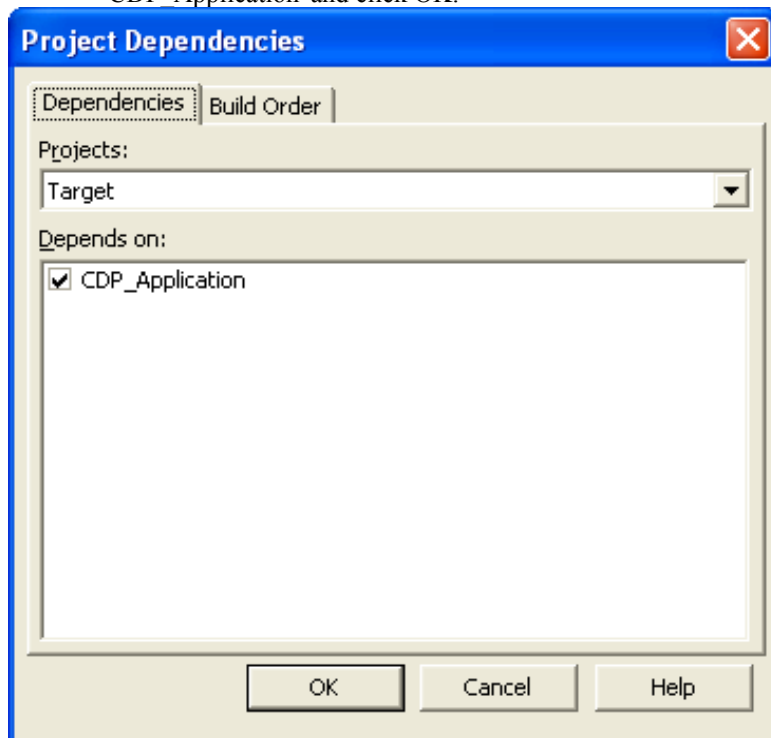
9. In the 'Solution Explorer', inside the 'CDP_Application' project, locate the Header file 'Libraries.h', and add the line
`#include <CDPEventManagerLib.h>`



10. If there is no 'Target' project in your 'Solution Explorer', select 'Solution'->'Add'->'Existing Project'. Choose the 'Target.vcproj' from the same folder in which your 'CDP_Application.vcproj' is at.
11. Right-Click Solution, select 'Configuration Manager...'. Make sure that all configurations have 'Release_RTOS' selected in 'Configuration':



12. Choose 'Solution'->'Project Dependencies...', choose 'Target' from the pull-down list, check 'CDP_Application' and click OK.



13. Rebuild the entire solution to generate the CDP.rtb file in the CDP_Application\Release_RTOS\ folder.

2.3. Modify the project xml files

Make sure that the IPAddress and SubnetMask in the 'NetworkInterface'-section are configured to contain an unique IPAddress and correct subnet, e.g.:

```
<NetworkInterface Name="ETH0" MAC="" IPAddress="10.0.2.50" SubnetMask="255.255.255.0"></NetworkInterface>
```

Add the following to your project's Application.xml:

Inside the <Subcomponents> element, add an instance of the main CDPEvetnManager components, for instance:

```
<Subcomponent Name="CDPEventNode" Model="CDPEventNode" src="Components\CDPEventNode.xml"></Subcomponent>
<Subcomponent Name="CDPEventLogger" Model="CDPEventLogger"
src="Components\CDPEventLogger.xml"></Subcomponent>
<Subcomponent Name="CDPEventSubscriber" Model="CDPEventSubscriber"
src="Components\CDPEventSubscriber.xml"></Subcomponent>
```

This will tell CDP to initialize the components named “CDPEventNode”, “CDPEventLogger” and “CDPEventSubscriber” from a component files located at respectively “Components\CDPEventNode .xml”, “Components\ CDPEventLogger .xml” and “Components\ CDPEventSubscriber .xml”. Make sure that your Models\ folder contains an CDPEventNode.xml, CDPEventLogger.xml, CDPEventSubscriber.xml model file, or the component will not be initialized correctly.

The CDPEventManager Model files CDPEventNode .xml , CDPEventLogger .xml and CDPEventSubscriber .xml can be found in \$(CDPBase)\Templates\Project_template\Application\Models\, and an example component files CDPEventNode .xml , CDPEventLogger .xml and CDPEventSubscriber .xml can be found in \$(CDPBase)\Templates\Project_template\Application\Components\.

2.4. Install on a Controller

Assuming you have a controller with PXE boot and a disk, you can use ControllerSetup.exe to download a bare-bone CDP application to the controller. See the document 'ControllerSetup.pdf' in the Doc folder where you installed CDP for more information on setting up a controller.

When the Controller has a CDP Application installed, you can use CDPFileManager.exe to upload files to the controller. If you have built for RTOS, upload these files to the root of the controller disk.

- CDP_Application\Release_RTOS\CDP.rtb.
- Everything inside the Application folder.

Make sure that you also upload rttboot.com from your “\$(RTTarget)\bin” folder to the root folder of the controller. This ensures that the RTOS32 loader is the most recent version