PCYNLITX SOFTWARE TECHNOLOGIES

PCYNLITX NOVA AUTOMATIC C++ BUILD SYSTEM GENERATOR-CONFIGURATOR

ERKAM MURAT BOZKURT

What is pcnylitx nova

I can easily describe the pcynlitx nova platform as a good start point for a fully automatic C++ build system. More specifically, the pcynlitx nova platform can determine the dependencies of each source file existing on a C++ git repository by means of a meta program without requiring any code or configuration. I am the only developer of the pcynlitx nova platform and I have developed pcynlitx nova platform completely based on my personal effort.

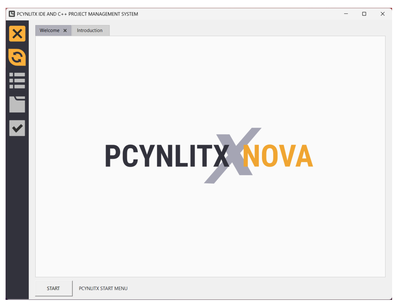

Figure 1. Pcynlitx Graphical User Interface

The Pcynlitx Nova has two main parts. These are the graphical user interface (the GUI) and the pcynlitx kernel. Even though the GUI of the pcynlitx nova can use as a classical code editor, its main responsibility is performing the necessary interaction with its user in C++ build system construction process. Differently from the technologies existing on the current software industry, pcynlitx nova only needs a small amount of information about the directory locations and the compiler to be used. Not surprisingly, this information is collected from the user by means of the GUI. Then, the kernel, which is in fact a collection of meta-programs, receives the information collected by the GUI. The pcynlitx kernel has a distributed structure and some parts of the kernel perform reading operations while the other parts perform writing (*code producing*).

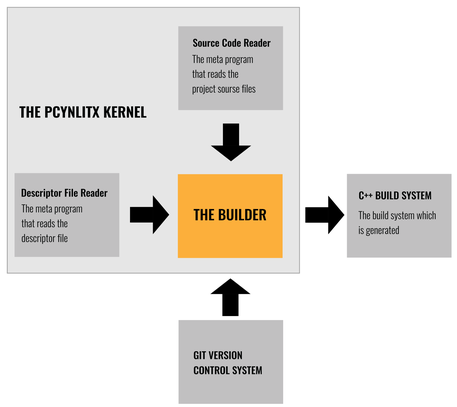

Figure 2. The structure of pcynlitx kernel

How pcynlitx nova produce C++ build systems

In fact, automatic build system construction is performed in several steps. Although the actual construction process is more complex, I have listed some steps which are followed on the automatic build system construction.

1. **Project file construction:** The user enters preferences from a GUI panel and a descriptor file (or a project file, I am not sure which is better naming) is constructed automatically.
2. **Project file reading** The preferences are read by a subsystem of the kernel.
3. **Receiveing the project file list from git:** The list of files existing on the project repository and their paths are received from the git version control system.
4. **Determination of the type of the files:** A meta-program determines the type of the files existing on the repository in terms of source file relation. In other words, the meta program determines whether the files are source files or other kinds of file.
5. **Reading the source files:** The source files are read line by line and their contexts are stored in to the memory as a proper data structure.
6. **Dependency determination:** The header declarations of the source files are determined and the dependencies of the each header existing on the source code is determined recursively. (In addition to this,the C++ modules will be added in the future. I have started from the headers for backward compatibility)
7. **Build System Construction:** After dependencies determined, build systems and/or configuration files are written.

The build system types which can be constructed

Currently, pcynlitx nova can write CMAKE list files and the build systems including GNU-Make files and script files. However, the pcynlitx nova platform can produce a C++ build system for any size of the C++ project and therefore, it can be used as a translator for one build system to the others in the future. (For instance CMAKE to Basel)

Figure 3. The build system technologies which is currently supported

What kind of information is necessary in construction process

In fact, the pcynlitx nova only needs directory information about project's source files and the construction point. In addition to this, the pcynlitx nova also needs the options of the compiler to be used. The exact list of data which is required in build system construction can be found from the project web site. Currently, pcynlitx nova only supports GNU gcc/g++ compiler. The other compilers will be added in the future.

- **Project web site:** https://www.pcynlitx.com
- **Project Repository:** https://github.com/Erkam-Murat-Bozkurt/pcynlitx.nova

How pcynlitx nova reads source file dependencies

The kernel of the pcynlitx nova has a meta-program which can read the whole git repository line by line and it extracts dependency information. More specifically, the pcynlitx nova reads each source file existing on the target git repository and it extracts every file dependency based on information which is determined using some C++ language properties such as key words, scope resolution operators, function braces and/or operators etc. Meanwhile, this operation is performed using parallel multi-threaded algorithms particularly. A simple code reading operation has been illustrated in below.

A sample code reading operation

include <sample>
include <repository/test>

–> The code lines which the keyword "include" places is determined.

–> Then, the information between the braces is extracted.

C++ modules

The project starts header reading in order to provide backward compatibility. However, the same approach can be easily applicable for C++ modules.

The usage from the command line

The usage of the pcynlitx nova from command line is possible. In addition the project file can also be written by the software developer on a classical text editor. When the project file (descriptor file) is ready, the C++ build system can be constructed with a simple shell command.

```
pcynlitx_nova [project file path] -ip
```

-ip: The options specifies the build system construction from the command line (or shell)

The project links

- **Project web site:** https://www.pcynlitx.com
- **Project Repository:** https://github.com/Erkam-Murat-Bozkurt/pcynlitx.nova

er.muratbozkurt@gmail.com