Presenting the A–A–P project

Bram Moolenaar <bram@a–a–p.org>
Stichting NLnet Labs

Open source software has the inherent possibility for an application to be ported and modified by many people. For a user it is not obvious where to obtain a version for his system, which patches to apply, what tools are required and how to upgrade to a new version. For a developer it is not easy to find his way in a project, distribute his changes in a user–friendly way and write portable code. A–A–P presents a solution for this by forming a framework in which existing tools can work together.

A–A–P for end–users

When a user wants to install an application he only needs to find a recipe. It contains instructions for A–A–P how to install this application. A–A–P will download all the required files, configure, build and install the application. When there are patches for this specific Operating System, these are downloaded and applied first. The user does not need to know where to find all the files and tools involved.

This is similar to how the FreeBSD ports system works. But A–A–P is portable over many Operating Systems. Additionally, A–A–P handles dependencies on other applications (with a range of versions) and can update an application to a new version.

A–A–P for developers

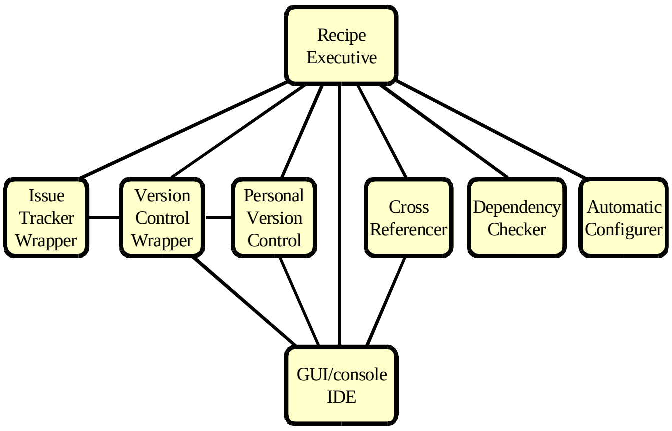
A developer starts, like an end–user, with installing the application. He can then browse the code, lookup documentation and make changes. A–A–P helps him by providing an IDE to manage the project, provide a smart interface to a version control system, lookup symbol references and definitionsi, etc. All this is also available at the command–line. Most of the actual work is done by existing tools.

What is different from other IDEs is the integrated support for working together with other developers over the internet. Differences between versions of various developers can be viewed and merged in. Patches can be generated and distributed. Parts of the project may only be downloaded when needed, while the code browser knows beforehand which symbols are used there. Documentation may be located elsewhere.

The recipe

The central element in A–A–P is the recipe. In many ways it looks like a Makefile. It contains rules with sequences of commands that are invoked by the user or executed when a target is older than a source file. It can be edited as a normal text file and generated for automatic dependencies and configuration. The main differences are:

- Python is used instead of shell commands, to be portable over many Operating Systems and provide powerful expressions.
- There is specific support for downloading archives, patches and other files on demand.
- The format is designed to be usable as a project file in an IDE as well.
- Support for version control, such as updating a project from CVS and checking in updated files.
- Many default rules exist for building a program, version control, code browsing, etc.



Modules

The central module of A–A–P is the Recipe Executive. It reads recipes and executes the commands contained in them. Most of the actual work is passed on to existing tools, such as a C++ compiler, ftp program and CVS. Other noteworthy modules:

Dependency Checker
Parses source files to find dependencies on other files (e.g., "#include" lines) and generates a recipe for this. Similar to "make depend". Works for many languages.

Cross Referencer
Generates cross reference databases and extracts symbol locations. Works for any file type (for some better than for others). Can be used to find where a symbol is defined, like ctags, and to find where a symbol is used, like grep. But it knows about programming languages, context, string constants, comments and documentation.

Automatic Configuration
Like "autoconf", but generates Python scripts to be portable over many Operating Systems and supports many languages.

Personal Version Control
Maintain versions without effort. This is like keeping a backup of every version of a file, but smarter. When sharing work with others the Version Control Wrapper is used.

The project

The A–A–P project has started March 2002. There is no working code yet. The first developer version is planned for September 2002. Until then it's vaporware.

The project leader is currently doing most of the work. Everybody is invited to contribute to the project. Not only to work on the implementation, also to discuss the features and choices. Or just to suggest a tool that can be used for A–A–P.

More info about the project on **www.a–a–p.org**. A–A–P is funded by stichting NLnet (www.nlnet.nl).

About the author

Bram Moolenaar has worked on open–source software for more than ten years. He is mostly known as the creator of the text editor Vim. His background is in computer hardware, but these days mostly works on software. He still knows on which side to hold a soldering iron though. In the past he did inventions for digital copying machines, until open–source software became his full–time job. He likes travelling, and often visits a project in the south of Uganda. Bram founded the ICCF Holland foundation to help needy children there. His home site is www.moolenaar.net.