

# Making My Own CPAN

How CPAN works, and how to do that myself

by brian d foy

brian.d.foy@gmail.com



There is nothing really magical about the Comprehensive Perl Archive Network (CPAN). It's a collection of files and directories, and all of the CPAN tools build on that so the end user doesn't have to know about it. That's one of the reasons that CPAN has been so successful: uploading is basically putting a new file in your author directory, and downloading is finding the right file. The CPAN tools do this for us, and they can do the same thing for a CPAN that I make for myself.

I want to create my own CPAN, and I have a couple different reasons to do it. First, many of our clients at Stonehenge Consulting Services want to maintain a local version of CPAN and put their own private modules in it. Some people want to maintain just the particular versions of CPAN distributions along with their private modules so they aren't surprised by updates or changes to popular modules on the public CPAN.

Even though my privately-created CPAN won't be comprehensive, I'll still call my creation MyCPAN, and say the C stands for "Comprehensive-enough". It will be just what I need, and nothing more.

## ■ The history of CPAN -----

The Comprehensive Perl Archive Network isn't a single system. Besides the actual archive, to most people the term CPAN effectively refers to the archive as well as the tools that work with the archive. The time line of CPAN's development shows how everything fits together. I'll only cover the points relevant for this article, and defer to the references at the end of this article for a more complete history.

CPAN began in late 1993 as an idea of Jared Rhine on the Perl Packrats mailing list, based on the Comprehensive TeX Archive Network (CTAN, <http://www.ctan.org>). In 1995, Andreas König suggested that there should be a single, central archive for Perl. Around that time, Jarkko Hietaniemi started the actual work of collecting and categorizing the available Perl source and external code on a server in Finland. The initial archive included the Perl sources and various scripts, but wasn't set up for Perl modules yet. It was already 390MB. Jarkko makes it public in October 1995.

A structure wasn't enough, though. Contributors needed a way to add new work, mostly modules, to the archive. Around the same time, Andreas starts work on the Perl Authors Upload Server (PAUSE) as a gateway for uploading to CPAN. Perl authors upload to PAUSE, which then CPAN mirrors (so CPAN is a superset of PAUSE). In turn, other servers mirror the content of CPAN to keep things quick and available. You are hardly ever talking to the master CPAN server in Finland, but probably one of the hundreds of mirrors publicly available

(and after this article, your own specialized mirror).

At the beginning, people had to download the modules they needed and resolve any dependencies themselves by downloading additional modules. In 1997, Andreas created `CPAN.pm`, a module to install modules and their dependencies automatically. His module is a programmatic interface to CPAN, and provides a shell for interactive use.

```
% perl -MCPAN -e shell
% perl -MCPAN -e 'install Business::ISBN'
```

In 1999, Graham Barr created CPAN Search (<http://search.cpan.org>), which is now what many people mean by "CPAN". Although relatively simple at the beginning, it now includes links to the various other projects built on CPAN, such as CPAN Testers (<http://testers.cpan.org>) and RT (<http://rt.cpan.org>).

CPAN Search is only one interface, though. Kobes' Search (<http://cpan.uwinnipeg.ca/>) is another one that provides a façade to the network. CPAN is really just the archive and the network, and it doesn't require any specific set of tools.

## ■ The CPAN architecture -----

The brilliance of the CPAN architecture is that it's almost trivial. CPAN is simply directories that have files in them. CPAN puts everything together from different sources so everything in Perl is in one place. The reason it works so well is that there isn't that much to maintain or create. Make some directories and place files in them. At the top-level, there are two directories that most users care about, and these come from PAUSE:

```
authors/
modules/
```

Inside the modules directory there are various index files that PAUSE creates so the CPAN tools can use them to find the right file in `authors/`.

```
02packages.details.txt.gz
03modlist.data.gz
06perms.txt.gz
```

The `02packages.details.txt.gz` contains the map from a particular module name to which distribution it is in. The first column is the module name, the second column is the latest version number of that module (not the distribution), and the last column is the distribution file name relative to the CPAN root. *Listing 1* shows four lines of `02packages.details.txt.gz` that show