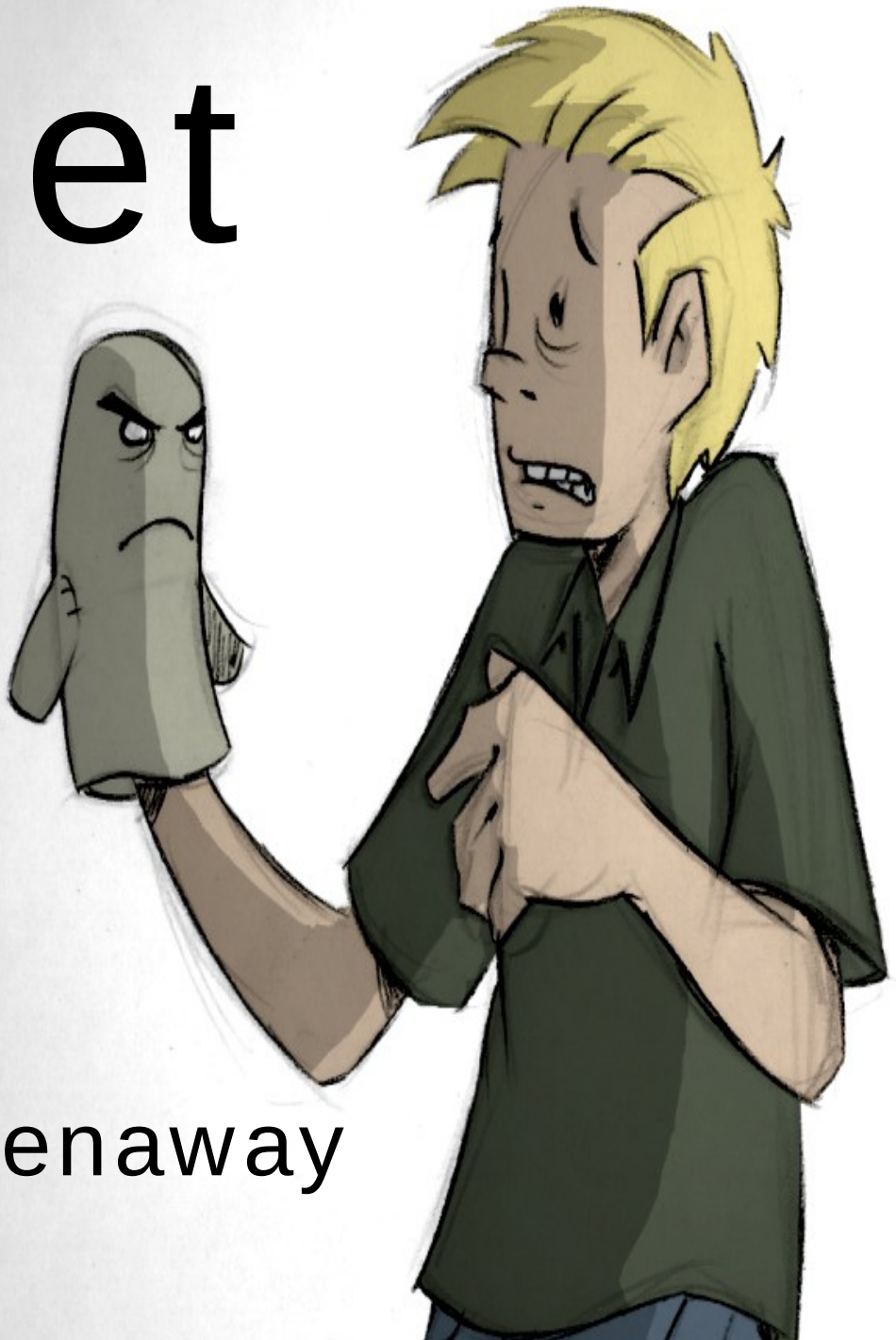
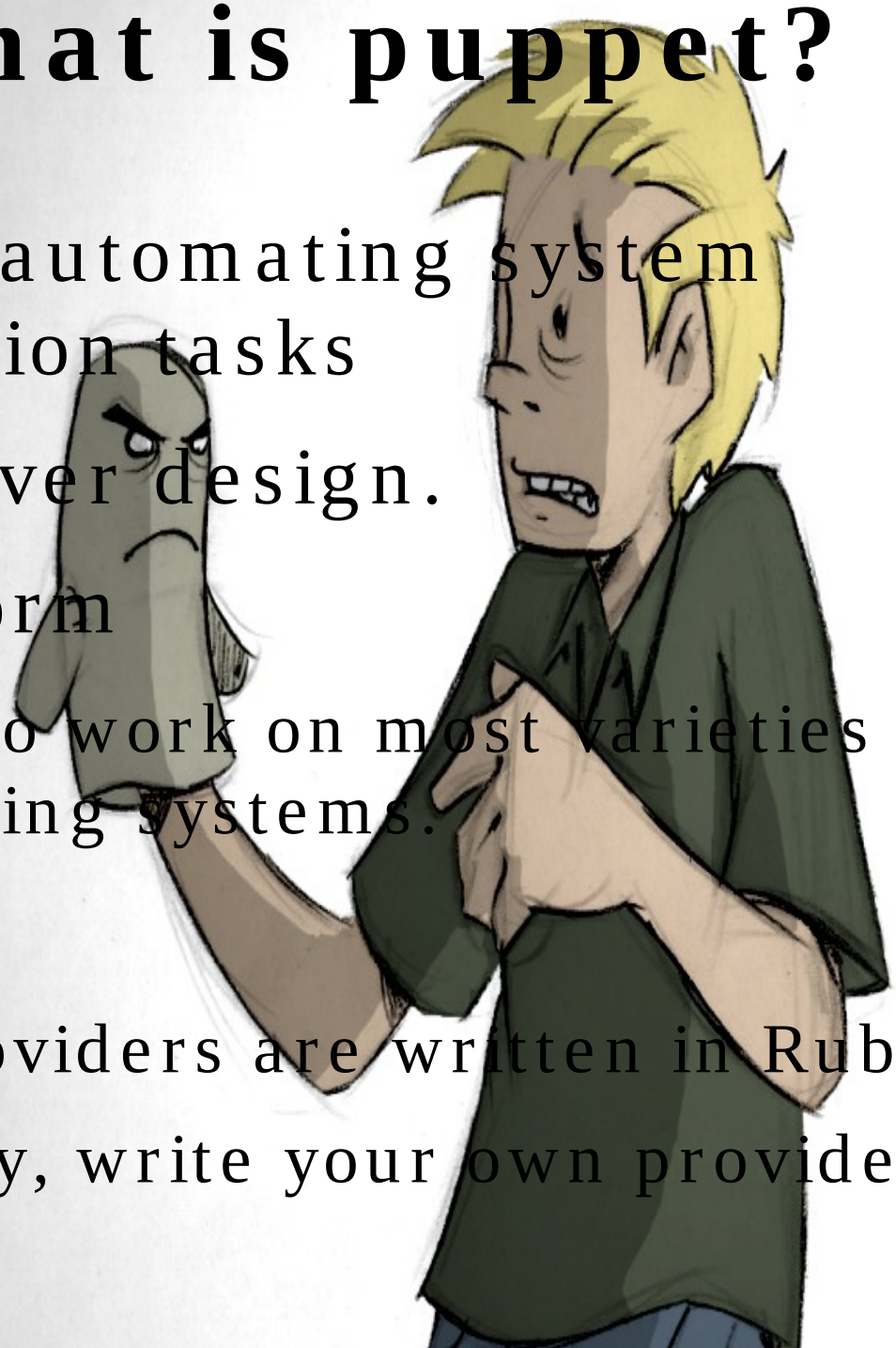
Puppet



Gareth J. Greenaway

What is puppet?

- System for automating system administration tasks
- Client / Server design.
- Multi-platform
  - Designed to work on most varieties of Un*x-like operating systems.
- Extensible
  - Puppet providers are written in Ruby.
  - Learn Ruby, write your own providers.

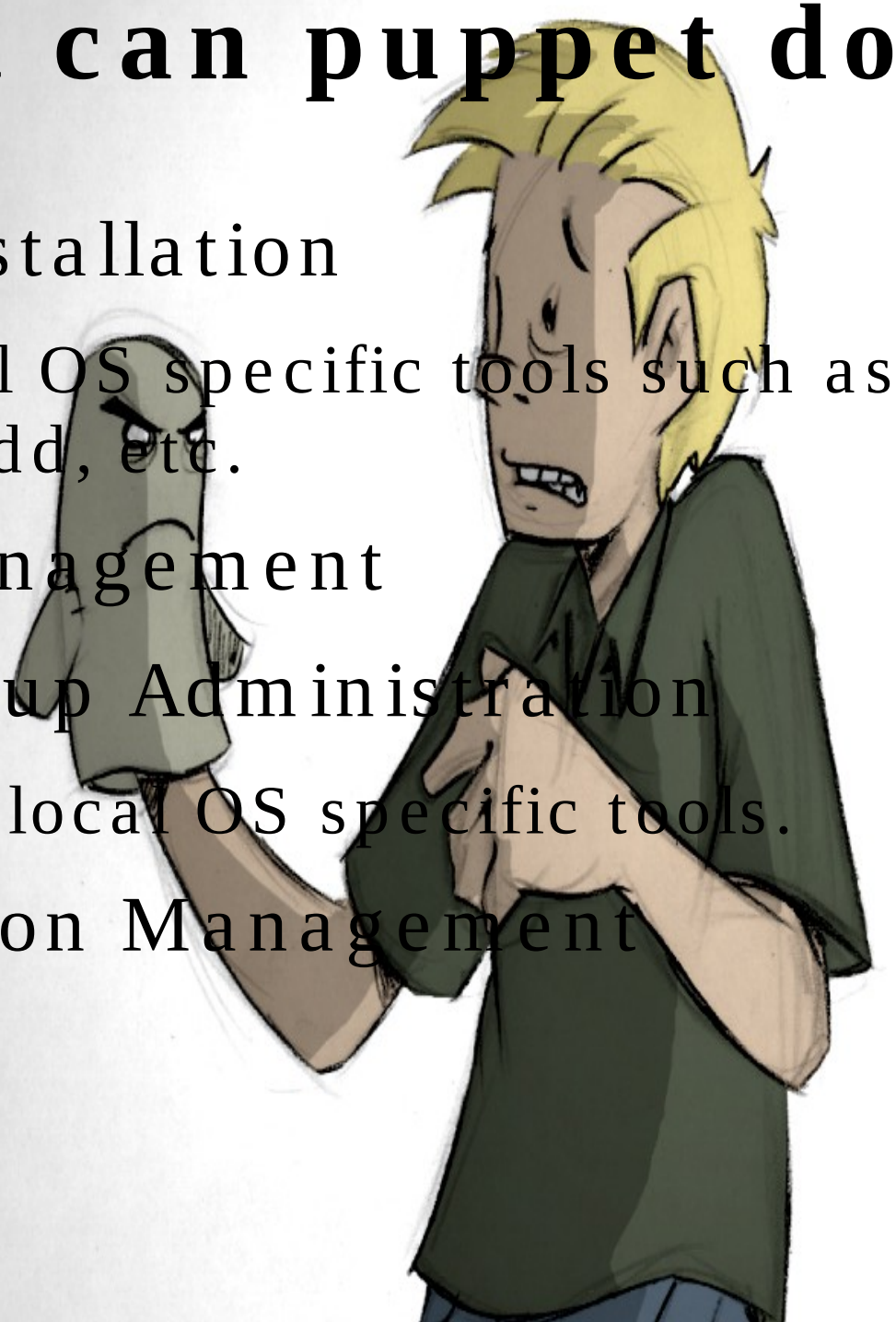


Why automation?

- Laziness
  - System Administrators are lazy, we want to do one thing once and move on.
- Efficiency
- Correctness

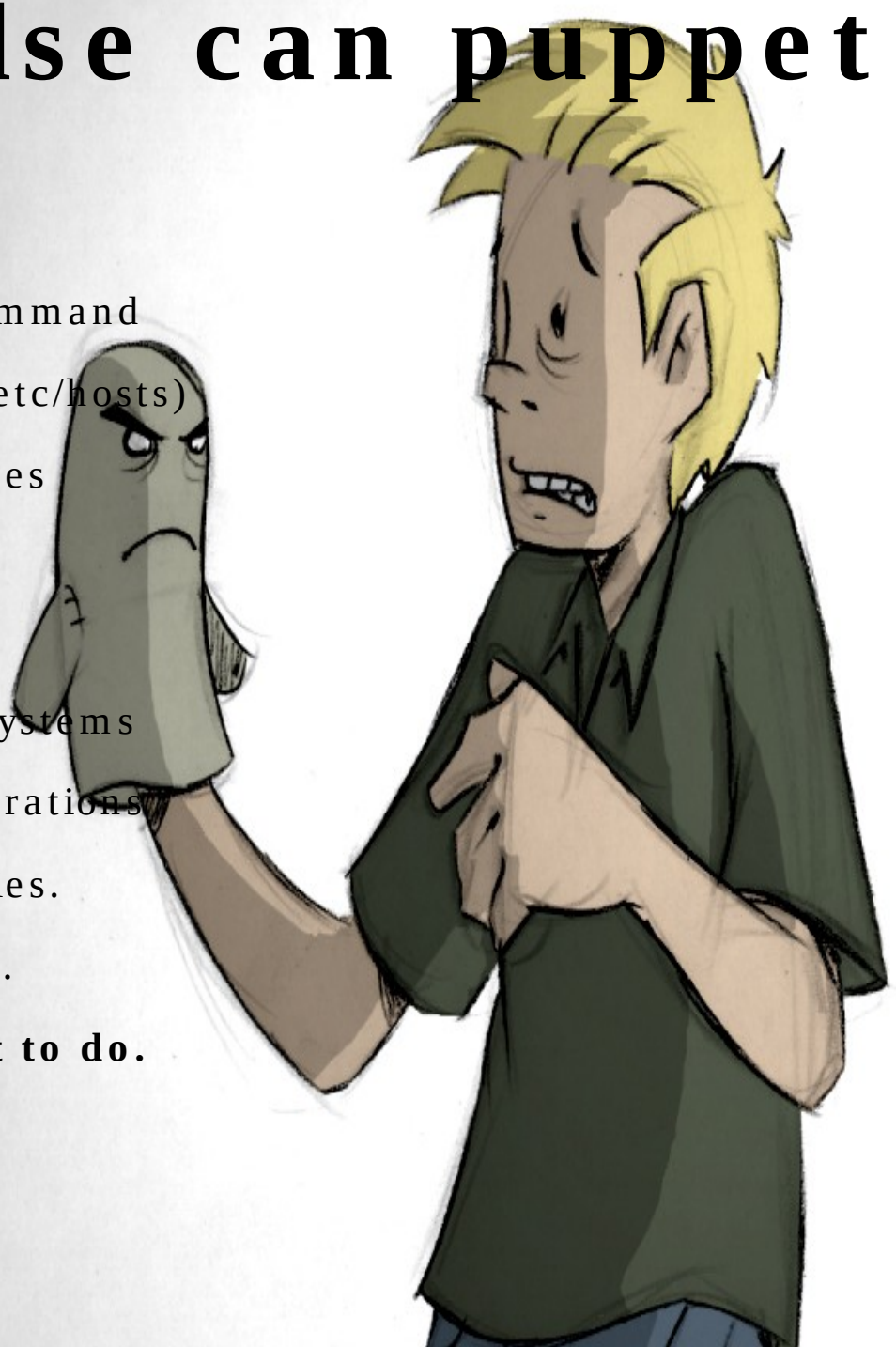
What can puppet do?

- Package installation
  - Using local OS specific tools such as yum, apt, pkg_add, etc.
- Service Management
- User & Group Administration
  - Also using local OS specific tools.
- Configuration Management



What else can puppet do?

- Manage cron jobs
- Execute any system command
- Manage host entries (/etc/hosts)
- Manage IP address alises
- Manage email aliases
- Manage mailing lists
- Manage mounted file systems
- Manage nagios configurations
- Manage yum repositories.
- Manager Solaris Zones.
- **Whatever you want it to do.**



How it works

- Classes
  - Classes determine what actions should be taken for a particular machine.
- Definitions
  - Definitions determine the actions that will take place within a particular class.
- Functions
  - Peform the actions
- Types

Puppet Pieces

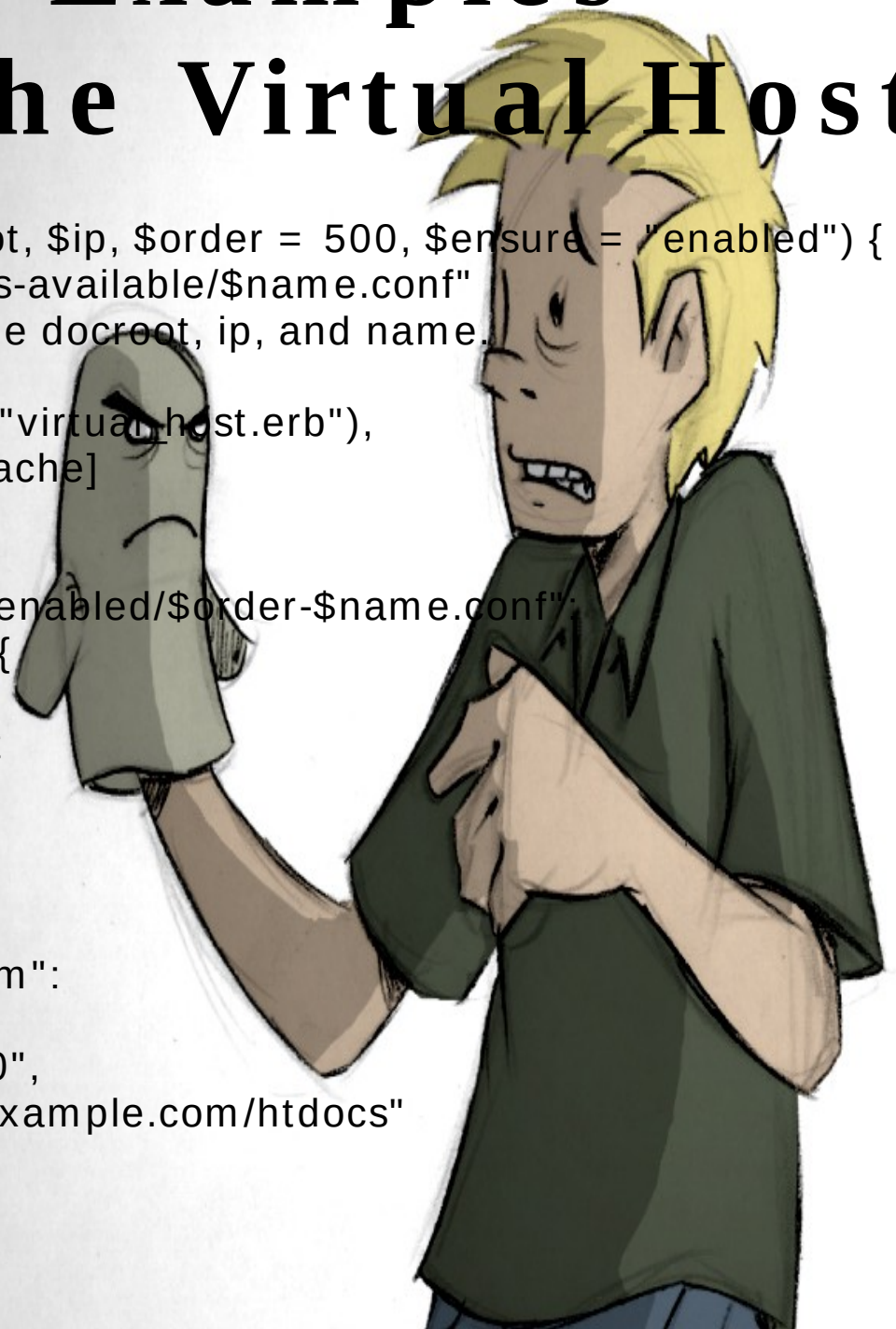
- Puppet Client (puppetd)
  - Runs on the client
  - Checks for updated configuration
- Puppet Master (puppetmasterd)
  - Stores all puppet manifests and acts as file server for puppet clients.

Examples
Apache Virtual Hosts

```
define virtual_host($docroot, $ip, $order = 500, $ensure = "enabled") {
   $file = "/etc/apache/sites-available/$name.conf"
   # The template fills in the docroot, ip, and name.
   file { $file:
      content => template("virtual_host.erb"),
      notify  => Service[apache]
   }

   file { "/etc/apache/sites-enabled/$order-$name.conf":
      ensure => $ensure ? {
         enabled  => $file,
         disabled => absent
      }
   }
}

virtual_host { "example.com":
   order   => 100,
   ip      => "192.168.0.100",
   docroot => "/var/www/example.com/htdocs"
}
```



Examples
Create Subversion Repository

```
# Create a new subversion repository.
define svnrepo($path) {
   exec { "create-svn-$name":
      command => "/usr/bin/svnadmin create $path/$name",
      creates => "$path/$name" # only run if this file does not exist
   }
}

svnrepo { puppet: path => "/var/lib/svn" }
```

Examples
Installing Custom Postgresql

```
class postgres {
	file { "/ecdb00/pgdata": ensure => directory,mode => 700, owner => postgres, group =>
postgres, }
	exec { "postgres-ldconfig":
		command => "/sbin/ldconfig", subscribe => File["/etc/ld.so.conf.d/postgres.conf"],
refreshonly => true}
	configfile { "/etc/ld.so.conf.d/postgres.conf":
		source => "/postgres/postgres.conf.ld.so",ensure => present,mode => 644,owner => root,
group => root,
	}
}

class postgres-8_3 inherits postgres {
	package { "postgresql-8.3.1-1.AMG": ensure => latest }
	service { postgresql:
		ensure => running,
		enable => true,
		require => [ Package["postgresql-8.3.1-1.AMG"], Exec["/etc/init.d/postgresql initdb"],
				Configfile["/ecdb00/pgdata/postgresql.conf"]
		],
	}
	exec { "/etc/init.d/postgresql initdb":
		path => "/usr/bin:/usr/sbin:/sbin:/bin:/usr/local/postgresql/bin/",
		onlyif => "test ! -f /ecdb00/pgdata/postgresql.conf",
		require => [ Configfile["/etc/ld.so.conf.d/postgres.conf"], Package["postgresql-8.3.1-
1.AMG"] ],
	}
	configfile { "/ecdb00/pgdata/postgresql.conf":
		source => "/postgres/postgresql.conf",
		ensure => present,mode => 600,owner => postgres, group => postgres,
		notify => Service[postgresql]
```


Real Live
Example!