
Puppet
07.03.2012 | Univention - CeBIT
NETWAYS®
Thomas Gelf
www.netways.de


KURZVORSTELLUNG
Einführung
PUPPET
Merkmale, Komponenten
Aufbau und Architektur
In der Praxis
ROLLOUT
FRAGEN UND ANTWORTEN

NETWAYS®

KURZVORSTELLUNG

Kurzvorstellung Thomas Gelf

- Seit 2010 bei der NETWAYS GmbH
- Zuvor über 10 Jahre in
  - Web (Application) Development
  - Netzwerk (Banken- und ISP-Backbone)
  - Systemplattformen im ISP-Umfeld
- Südtiroler!

NETWAYS®

NETWAYS Expertise

OPEN SOURCE SYSTEMS MANAGEMENT

- Monitoring & Reporting
- Configuration Management
- Service Management
- Knowledge Management
- Backup & Recovery

OPEN SOURCE DATA CENTER

- High Availability & Clustering
- Cloud Computing
- Load Balancing
- Virtualization
- Database Management

MANAGED SERVICES | MONITORING HARDWARE | KONFERENZEN

NETWAYS®

NETWAYS Konferenzen

OSDC.de
OPEN SOURCE DATA CENTER CONFERENCE

Open Source Data Center Conference
25 – 26 April 2012

- 100 Teilnehmer (2011)
- "Agile Infrastructures"
  - Devops & methods
  - Databases
  - Scalability & infrastructure

OPEN SOURCE MONITORING CONFERENCE on Nagios

Open Source Monitoring Conference
17 – 18 Oktober 2012

- 260 Teilnehmer (2011)
- Icinga / Nagios case studies & best practices
- Latest monitoring technologies & addons

NETWAYS®

Puppet Schulungen

NETWAYS ist

- Offizieller Schulungs-Partner von Puppet Labs
- Erster Puppet Schulungsanbieter in Deutschland

Nächster Termin:

- 27. - 29. März 2012 | Nürnberg

Weitere:

- 12. - 14. Juni 2012 | Nürnberg
- 26. - 28. Juni 2012 | Zürich

NETWAYS®

Sonstiges

www.netways.de/jobs

NETWAYS®

EINFÜHRUNG

Was ist Puppet eigentlich?

- Werkzeug für das Konfigurationsmanagement
- Automatisierungstool
- Anregung zum Reflektieren eigener Arbeitsweise
- Alle Systeme sind gleich. Oder?
- Abstraktionsschicht zwischen Admin und Systemen
  - DevOps!
- In Ruby geschrieben

NETWAYS®

Ruby?

- Hat nichts mit Ruby Rubacuori zu tun
  - http://de.wikipedia.org/wiki/Ruby-Affäre
- Interpretierte, objektorientierte Sprache
- 1995 von Yukihiro Matsumoto entworfen

NETWAYS®

Konfigurationsmanagement?

- Ist laut Wikipedia eine “Managementdisziplin”
- Management?
- Disziplin?
- KMO, KI, KÜ, KB, KA – k.A.?

NETWAYS®

DevOps?

- Ideengeber:
  - Enterprise Systems Management
  - Agile Infrastructures
- Kurz: man hat verstanden, dass es von Vorteil sein kann, wenn Sysadmins und Entwickler einander näher kommen

<3 <3 <3

NETWAYS®

Klassische Aufgabenteilung

- Entwicklung
  - Anforderungen an Funktionalität
  - Häufig direkter Zusammenhang mit Business-relevanten Anforderungen
- Betrieb
  - Versucht häufige Änderungen zu vermeiden
  - Muss stabilen Betrieb gewährleisten
- Änderungen in größeren Blöcken in größeren Abständen bedeuten aber auch höheres Risiko bei der Umstellung

NETWAYS®

Das Problem dabei...

- Um Änderungen zu vermeiden, bremst der Betrieb neue Features aus
- Betrieb kennt nicht sämtliche Interna der Applikation:
  - Passende Laufzeitumgebung zu definieren ist schwierig
- Entwicklung fehlt Einblick in Systemumgebung:
  - Macht es schwer, die Applikation entsprechend anzupassen

NETWAYS®

DevOps-Modell führt zu:

- Führt zu
  - Häufigeren und kleineren Änderungen
  - Weniger riskanten Änderungen
  - Simplerem Rollback
- Gibt
  - Entwicklern mehr Kontrolle über die Plattform
  - Dem Betrieb mehr Einblick in die Applikationen

NETWAYS®


NETWAYS
KONFIGURATIONSMANAGEMENT

Ein Linux/Unix-System verwaltet man...

- Mit Kommandozeilentools
- Einem Texteditor
- Skript-Kenntnissen

NETWAYS®

Hat man viele solcher Systeme...

- Automatisiert man wiederkehrende Aufgaben
- Jeder hat seine so seine Skript-Sammlung in der Schublade
- Kann zum Problem werden

NETWAYS®


NETWAYS
PUPPET

Eckdaten

- Luke Kanies
- Puppet Labs
- Lizenz: GPL -> Apache
- Puppet Enterprise

NETWAYS®

Der Mitbewerb

- bcfg2
- cfengine
- chef
- ssh & for-Schleifen
- ...

NETWAYS®

Welche Vorteile bietet Puppet?

- Reproduzierbarkeit
- Konsistenz
- Einfacher Einstieg
- Portabilität (Linux, Solaris, BSD...)

NETWAYS®

Warum ist Puppet anders?

- Bestimmt nicht Abläufe, sondern Zustände
- Ist plattformunabhängig
- Jeder Ressource-Typ kümmert sich selbst um plattformspezifische Besonderheiten
- Liefert eine Bibliothek voll mit Rezepten
- Darauf aufbauend wird mit deklarativer Sprache festgelegt, wie die eigenen Systeme aussehen sollen

NETWAYS®

REZEPTE?

- Das Knödel-Problem:

NETWAYS®

Puppet Community

- Plattformen: Redhat, Debian, Ubuntu, Solaris, Mac OSX...
- Weit über 100 Code beitragende Mitglieder
- 1000e aktive User auf Mailinglisten
- Nonstop 100e User auf IRC-Channeln
- Jeweils für Benutzer und für Entwickler

NETWAYS®

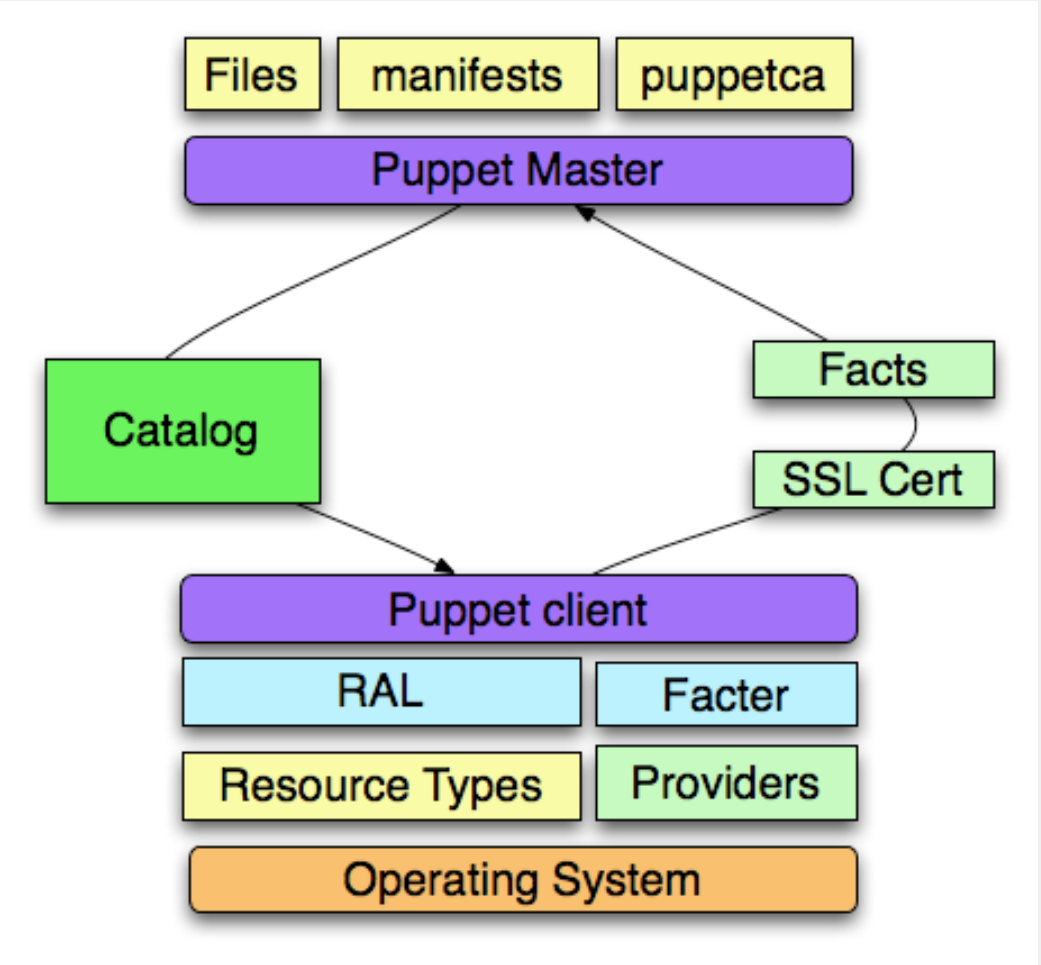
Architektur

- SSL-Verbindung vom Agent zum Master
- XML-RPC oder auch REST
- Master verpackt die Konfiguration für den Client
- Client vergleicht diese mit seinem Status Quo...
- ...und korrigiert eventuelle Abweichungen

NETWAYS®

Architektur

- Übersicht:



NETWAYS®

Fakten, Fakten, Fakten

- facter
  - Standalone-Tool
  - Plattformübergreifend
  - Ruby-Bibliothek
- System-Informationen
- Eigene facts sind problemlos möglich

NETWAYS®

Manifeste?

- Im Sinne der Ladungsliste eines Schiffes
- Was soll drauf sein
- Aber nicht: was müssen wir noch aufladen

NETWAYS®

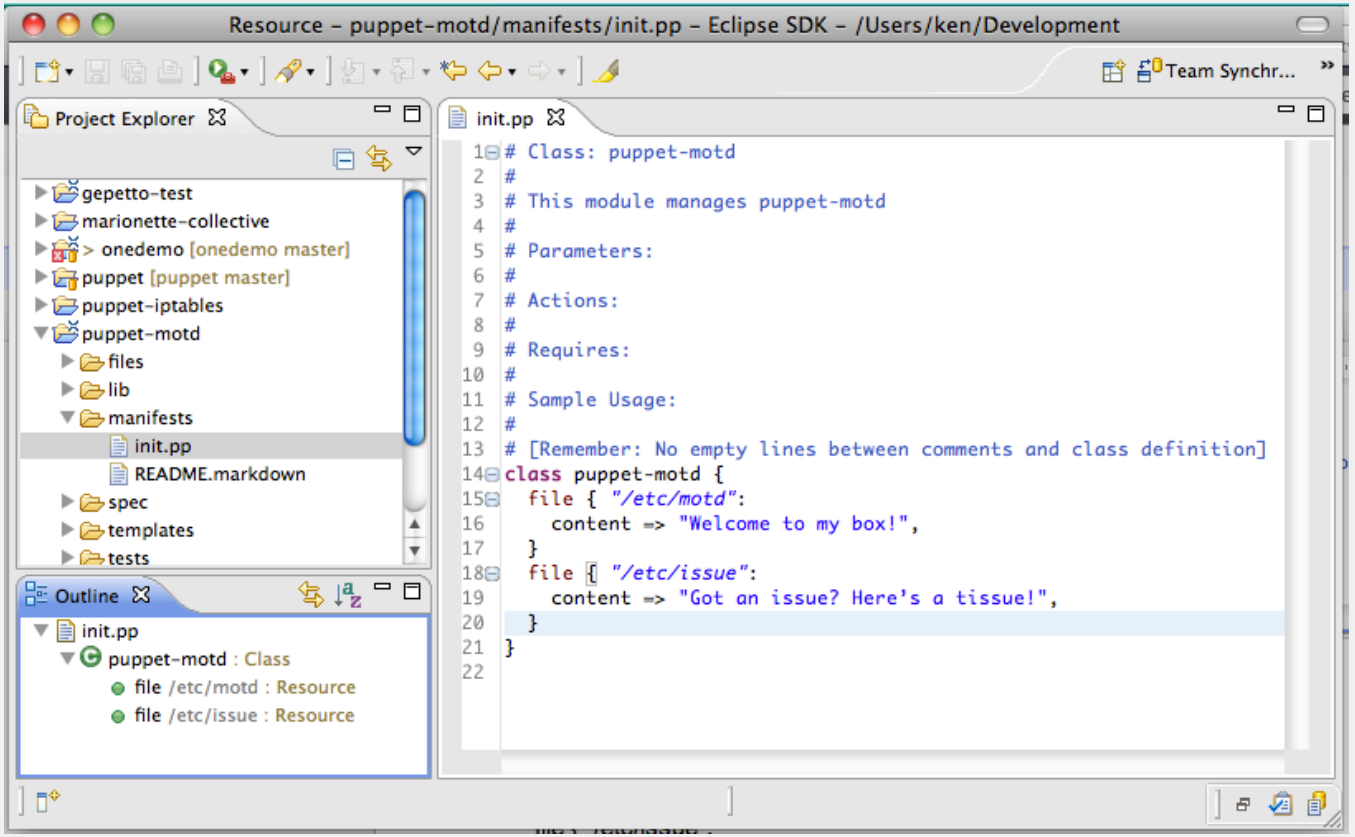
Wie soll so ein Manifest aussehen?

- Hübsch. Syntax Highlighting → vim-puppet
- Puppet lässt uns ziemlich freie Hand
- Sich an die "Best Practices" zu halten ist eine gute Idee:
  - http://projects.puppetlabs.com/projects/puppet/wiki
- Alles in "Modules" packen
- Beispiele suchen! (http://forge.puppetlabs.com/ etc)

NETWAYS®

VIM???

- Es gibt auch andere, für Eclipse z.B. Gepetto:



Ein erstes vollständiges Manifest

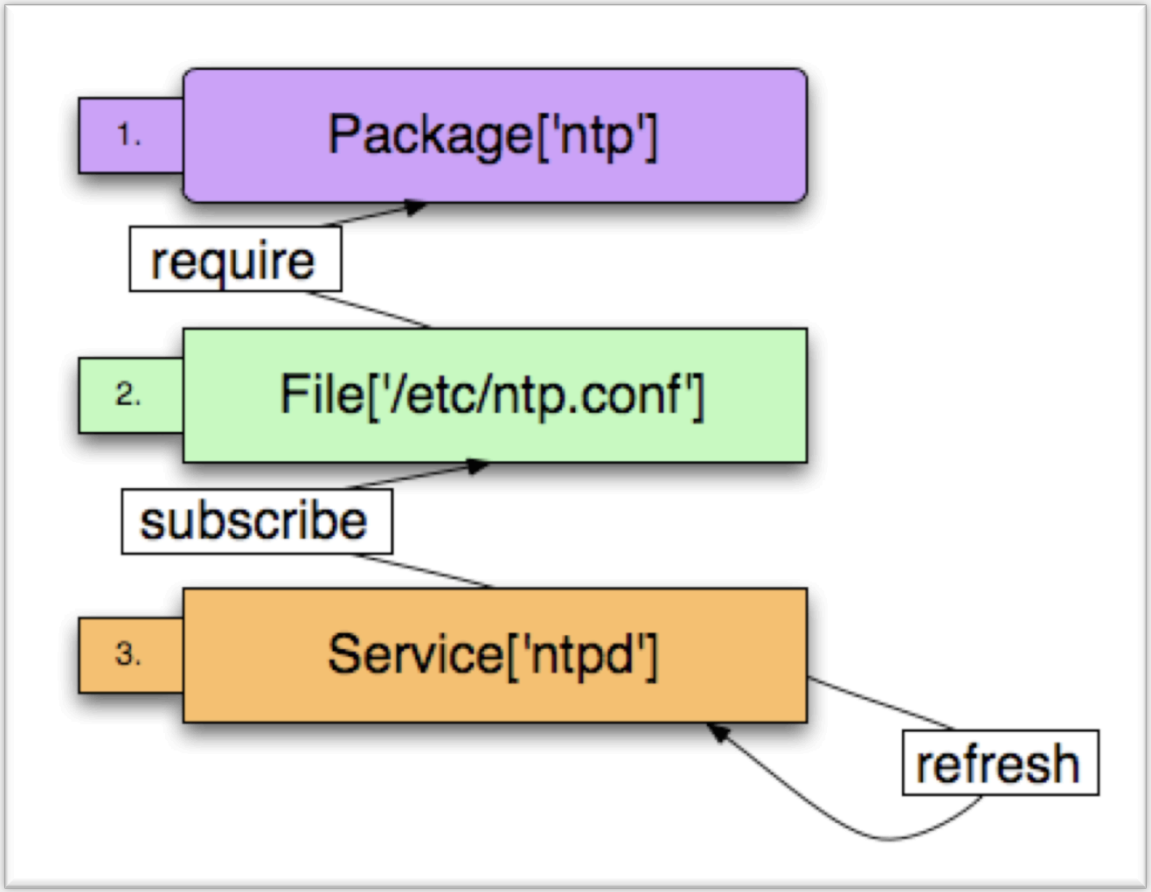
```
class hosting {
    case $operatingsystem {
        fedora: { $httpd_packages = ["httpd", "php"] }
        debian: { $httpd_packages = ["apache2", "libapache2-mod-php5"] }
        ubuntu: { $httpd_packages = ["apache2", "libapache2-mod-php5"] }
        default: { $httpd_packages = ["apache", "php"] }
    }
    package { $httpd_packages: ensure => installed }
    service { "webserver":
        name => $operatingsystem ? {
            debian => "apache2",
            ubuntu => "apache2",
            redhat => "httpd",
            default => "apache",
        },
        ensure => running,
    }
}

node "client2.example.com" {
    include hosting
}
```

NETWAYS®

Abhängigkeiten

- Dependencies
- Subscriptions



NETWAYS®

Schnelltest

- Es reicht eine einfache site.pp:

```
node default {
    notice("Unbelievable, it's really working!”)
}
```

- Auf dem Master muss im Syslog eine entsprechende Meldung erscheinen, in etwa so:

```
(Scope(Node[default])) Unbelievable, it's really working!
```

NETWAYS®

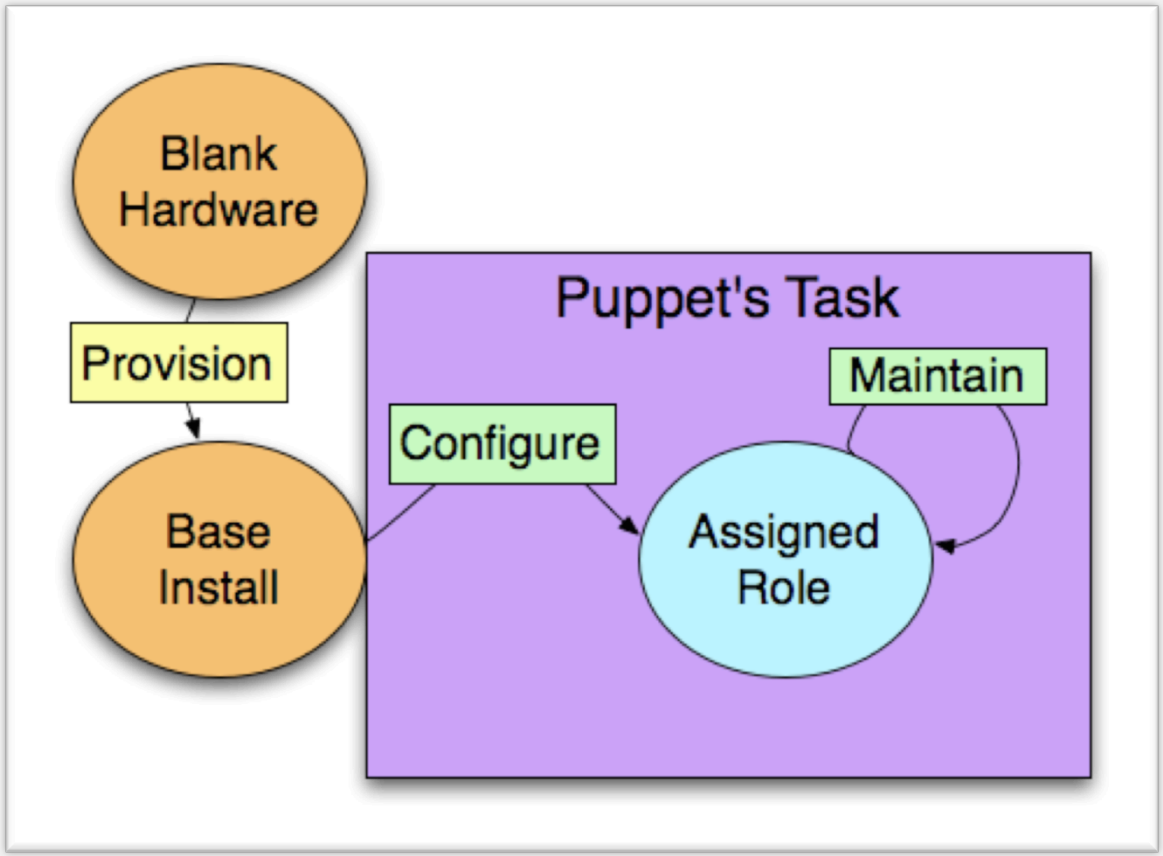
Wichtige Begriffe

- Catalog: Gesamtheit der Ressourcen, Dateien, Eigenschaften für ein bestimmtes System
- Class: Behälter für Ressourcen
- Definition, Defined Type: auf Anwendungsebene definierter Resource-Typ (!= native type)
- Plugin: eigene Typen, mit Ruby erstellt
- Templates: ERB-Dateien, um systemspezifische Konfigurationsdateien erstellen zu können
- Variablen: Variablen, aber unveränderlich

NETWAYS®

Zuständigkeiten

- Puppets Aufgaben beginnen NACH dem Bare Metal Deployment:



NETWAYS®

NETWAYS®

ROLLOUT

Auspacken, loslegen

- Konfigurationsmanagement will mit Bedacht eingeführt werden
- Aber vorher:
  - Rumspielen
  - Ausprobieren
  - Kennenlernen
- puppet apply!

NETWAYS®

Wo liegt die Konfiguration?

- Bei Bedarf auch LDAP oder SQL
- Aber: am attraktivsten mit klassischen Textfiles
- Versionskontrollsystem:
  - Sämtliche Änderungen historisch nachvollziehbar
  - Rollback
  - Changemanagement

NETWAYS®

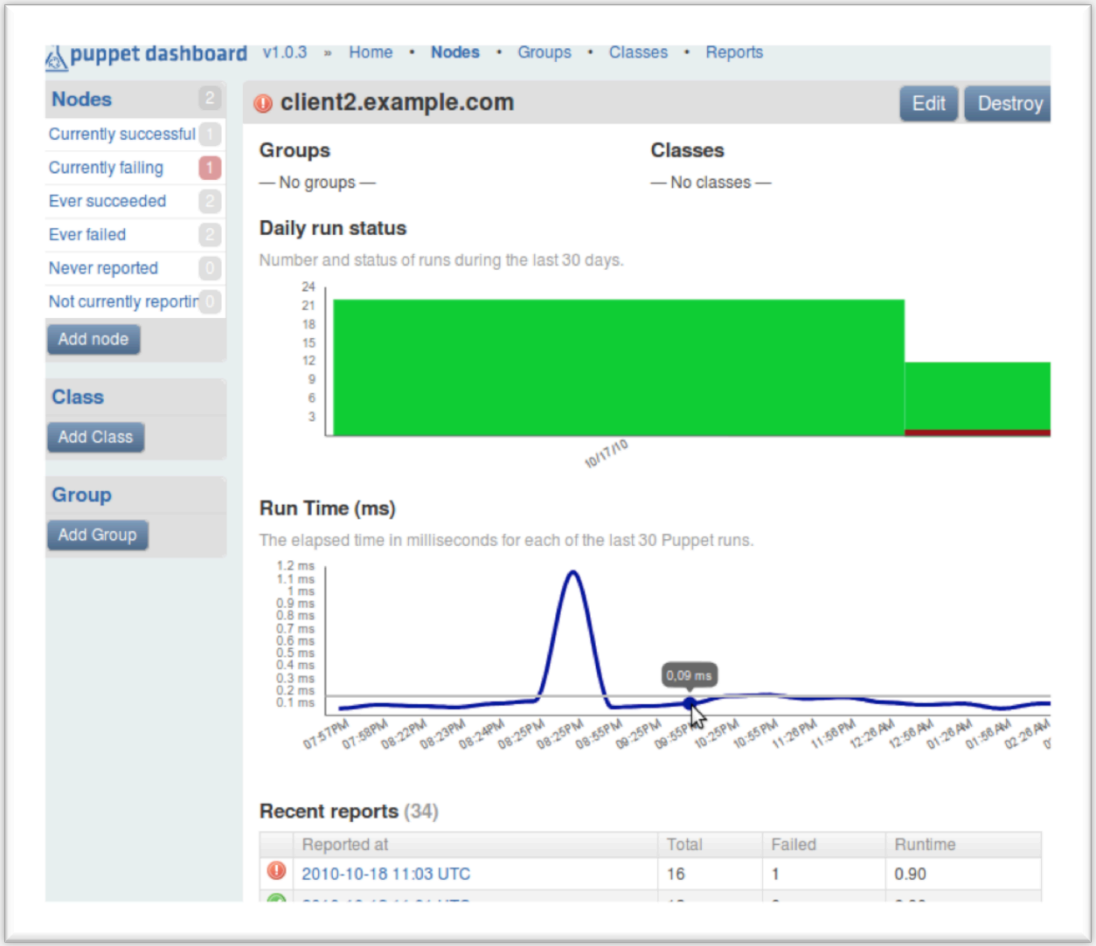
Puppet kann VCS? SVN? GIT?

- Jain.
- Use the right tool for the right job!
- Puppet stellt „nur“ die Infrastruktur zum Verteilen der Konfiguration bereit
- Wer diese liefert, ist ihm ziemlich egal

NETWAYS®

Webfrontends

- Foreman
- Puppet Dashboard



Testing

- “Include”-Tests
- Unit-Tests: rake
- BDD: Cucumber
- “Draufsicht”: z.B. automatisierte Nagios/Icinga Checks

NETWAYS®

Cucumber, Beispiel

```
Scenario: Confirming package installation
   When a machine has been puppeted
   Then the bash package should be installed


 Scenario Outline: Compile and verify catalog
   Given a node specified by "features/yaml/<hostname>.example.com.yaml"
   When I compile its catalog
   Then compilation should succeed
   And all resource dependencies should resolve
```

NETWAYS®

Cucumber, Beispiel

```
Feature: Base repositories
  In order to have a system that can install packages
  As a sysadmin
  I want all of the CentOS repositories to be available

  Scenario: CentOS yum repositories
    Given a node of class "yum::base"
   When I compile the catalog
    Then there should be a yum repository "Base"
    And there should be a yum repository "Updates"
    And there should be a yum repository "Extras"
```

* Shamelessly stolen from http://paperairoplane.net

NETWAYS®

FRAGEN UND ANTWORTEN

NETWAYS®

Question & Answer

NETWAYS GmbH

Deutschherrnstrasse 15-19

90429 Nürnberg

Tel: +49 911 92885-0

Fax: +49 911 92885-77

Email: info@netways.de

Twitter: twitter.com/netways

Blog: blog.netways.de