Ideas para una administración de sistemas 2.0

Javier García - jgarcia@um.es

UNIVERSIDAD DE MURCIA

Sobre la presentación

UNIVERSIDAD DE MURCIA

2004 empezamos a trabajar con cfengine

Algunas máquinas y servicios

2008 presentación sobre cfengine

Más máquinas y más servicios

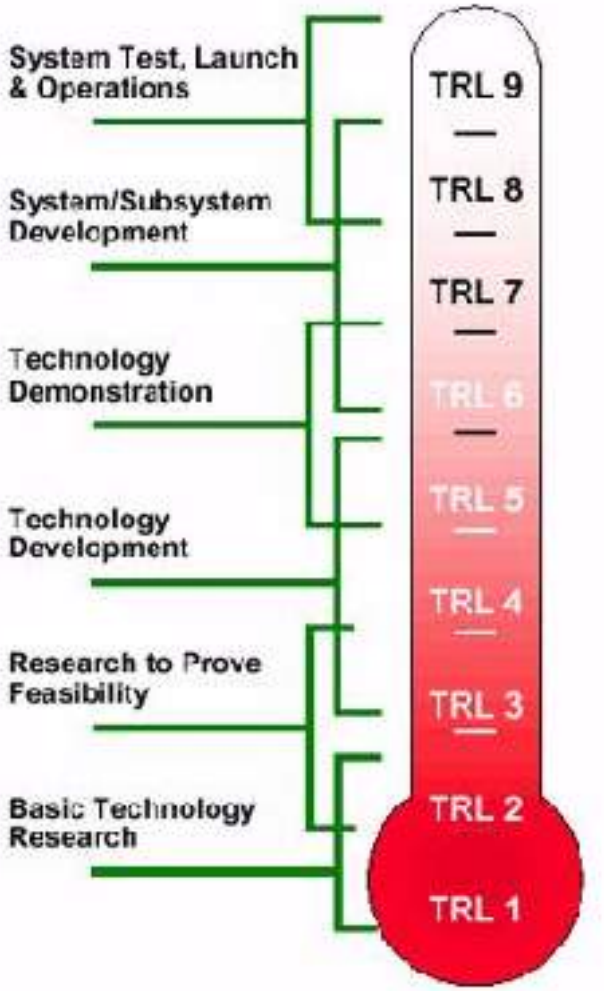
2010 migrando a puppet

Muchas más máquinas y más servicios

Otras herramientas de interés

Cómo puede evolucionar

Qué podemos hacer



¿Qué ha traído la Web 2.0?

UNIVERSIDAD DE MURCIA

- Nuevos servicios
- Nuevos niveles de servicio
- Nuevas espectativas de servicio
- Nuevos problemas
- Facebook: ¿500 millones de usuarios?
- ¿Nuevas soluciones?

Cloud computing

UNIVERSIDAD DE MURCIA

Diferentes tipos:

Software as a Service (SaaS)

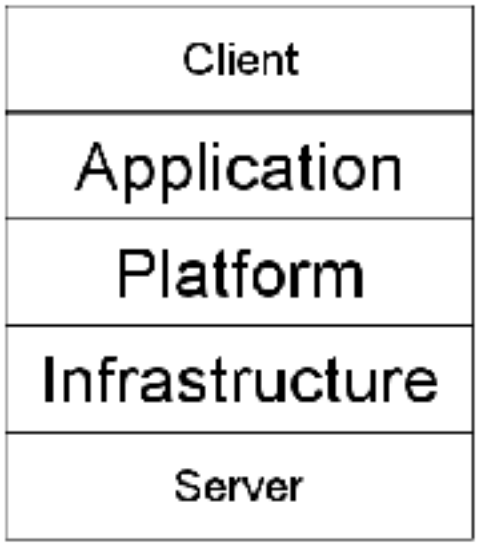
Salesforce, ...

Platform as a Service (PaaS)

Google apps,...

Infrastructure as a Service (IaaS)

AWS,...



Algunas características de IaaS

UNIVERSIDAD DE MURCIA

- Escalable
- Virtualizado
- Pago por uso
- Multi-tenancy
- Elasticidad de recursos
- Eficiencia

Características nube

UNIVERSIDAD DE MURCIA

API's para gestión infraestructuras

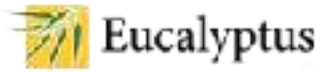
Automatizable

Proyectos como:

http://open.eucalyptus.com/

http://www.fedcloud.com

http://auser.github.com/poolparty/



A todo el mundo no le gusta

UNIVERSIDAD DE MURCIA

"One reason you should not use web applications to do your computing is that you lose control,"

"Somebody is saying this is inevitable -- and whenever you hear somebody saying that, it's very likely to be a set of businesses campaigning to make it true,"

"It's stupidity. It's worse than stupidity: it's a marketing hype campaign,..”

Richard Stallman

¿Qué podemos hacer para?

UNIVERSIDAD DE MURCIA

No rehacer trabajo

Reaprovechar lo hecho

Reaprovechar conocimiento

Buenas prácticas

Autodocumentación

Mejorar ratio sistemas-administrador

Sirva en todas las fases del ciclo de vida

Estabilidad, reducción incidencias, puesta en explotación,...

Espera fallos...

UNIVERSIDAD DE MURCIA

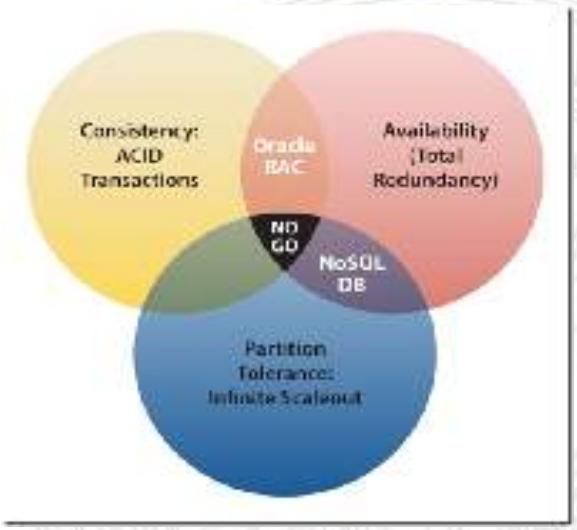
Teorema de CAP (Wikipedia)

The CAP theorem, also known as Brewer's theorem, states that it is impossible for a distributed computer system to simultaneously provide all three of the following guarantees:

* Consistency (all nodes see the same data at the same time)

* Availability (node failures do not prevent survivors from continuing to operate)

* Partition Tolerance (the system continues to operate despite arbitrary message loss)



Aprendiendo de la experiencia ...

UNIVERSIDAD DE MURCIA

Hardware

Muchos y baratos que pocos y caros

Potencia vs I/O

Consumo vs potencia

Simplicidad

Evitar dependencias innecesarias

Instalaciones simples

Aprendiendo de la experiencia ...

UNIVERSIDAD DE MURCIA

Implicar a desarrollo

Mayoría (¿80%?) problemas por cuestiones de diseño y desarrollo

"you built it, you manage it" Amazon

Bajo coste de administración relacionado con cómo de cerca se trabaja con equipos de desarrollo

Diseñar para fallos

Stateless, interfaces Rest, ...

Aprendiendo de la experiencia ...

UNIVERSIDAD DE MURCIA

- Implicar a desarrollo
- Logueo configurable
- Test funcionales
- Versionado de todo
- Estilo aplicaciones Rails
- Borrado soft
- Evitar puntos únicos de fallo
- Ruby cada vez más protagonismo

Aprendiendo de la experiencia ...

UNIVERSIDAD DE MURCIA

- Automatización de todo
- 24x7 caro
- El trabajar sobre presión suele introducir errores (20% o más) de las veces
- Provisión automatizada
  - A mano propensa a errores
  - Crea discrepancias
- Manejar roles y no servidores
- Apagado periódico, servidores, racks, ...

Aprendiendo de la experiencia ...

UNIVERSIDAD DE MURCIA

Puestas parciales en explotación con infraestructura real

Rollback

Reducir ciclos de empaquetamiento

Evitamos grandes cambios

Reducción de falsos positivos

Recolección de datos de producción

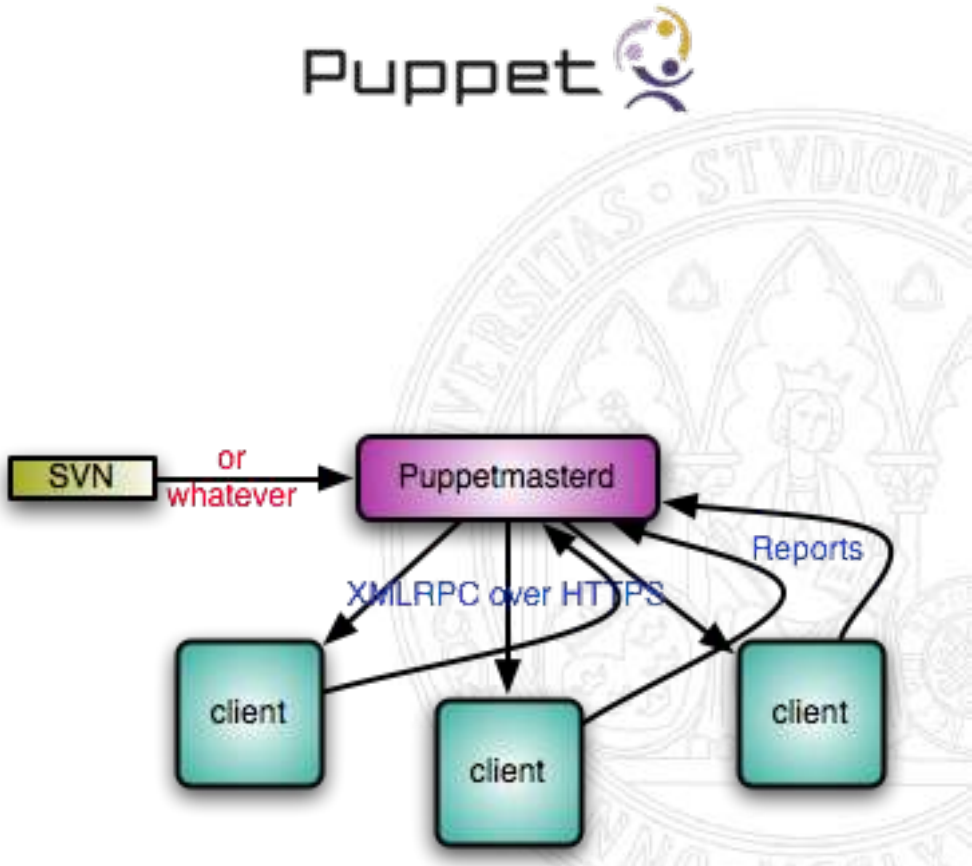
métricas

Ver tendencias

Herramientas: puppet

UNIVERSIDAD DE MURCIA

- Código abierto
- Lenguaje de configuración desarrollado en Ruby
- Alternativa de cfengine, e inspirado en él
- Fácil de extender, con una gran comunidad detrás
- Multiplataforma
- Un servidor central controla la configuración del resto de los equipos



Herramientas: puppet

UNIVERSIDAD DE MURCIA

Idempotencia

elementos que Puppet sabe como configurar

Fichero (contenido, permisos, pertenencia)

Paquete (asegurar que esté instalado o no)

Servicio (habilitado, deshabilitado, corriendo, o detenido)

Nagios

Otros:

cron, exec, group, host, mount, notify, user, sshkey.....

Puppet

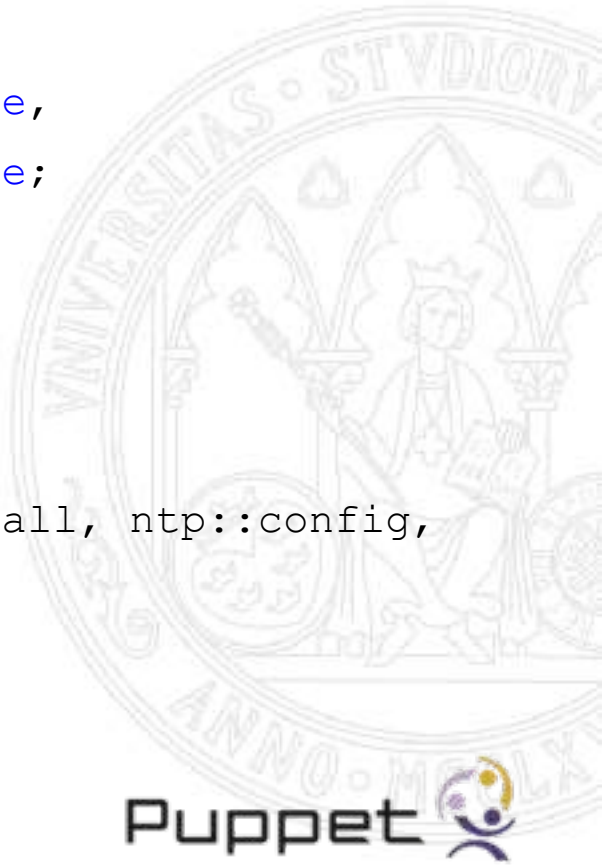
Ejemplo:clase para servicio NTP

UNIVERSIDAD DE MURCIA

```
class ntp::install {
    package { "ntp": ensure =>
   present}
}
class ntp::config {
    file { "/etc/ntp.conf":
        owner => root,
        group => root,
        mode => 440,
        content =>
   template("ntp/etc/ntp.conf.erb"),
        require =>
   Class["ntp::install"],
        notify  =>
   Class["ntp::service"];
    }
   }
```

```
class ntp::service {
service { "ntp":
enable => true,
ensure => true,
hasrestart => true,
hasstatus  => true;
   }
}

class ntp {
include ntp::install, ntp::config,
   ntp::service
}
```



Puppet

Ejemplo:Nodos

UNIVERSIDAD DE MURCIA

```
node basenode {

# Configuración básica de puppet
   $puppet_server = 'puppet.um.es'

# Locale por defecto
   $defLocale = "es_ES.UTF-8"

# Servidores NTP
   $ntpServers = [ "hora.um.es" ]

# Array con los procesos que se
   quiere monitorizar
   $snmpProcs = [ "rsyslogd",
   "puppetd" ]

}

node default inherits basenode {

    include baseclass

}
```

```
node 'puppetmaster.um.es' inherits
   basenode {

$snmpProcs += ["puppetmasterd"]
   include baseclass
   include rol_puppetserver

}
```

Puppet

Ejemplo:plantilla para servicio NTP

UNIVERSIDAD DE MURCIA

```
# ntp/etc/ntp.conf.erb
# Servidores NTP: Esta opción se configura mediante puppet. Para
# añadir más servidores, no hay más que añadirlos a la variable ntpServers
# del nodes.pp
#
<% ntpServers.each do |server| -%>
server <%= server %>
<%end -%>
```

Herramientas: Puppet

UNIVERSIDAD DE MURCIA

Pros

- Inspirado en cfengine
- Mejorando cosas
  - Herencia de clases
  - Comunidad de desarrolladores
- Repositorios de módulos
- ERB muy potente

Contras

- Hecho en falta comandos de edición en línea
  - Ámplia discusión
  - Se pueden crear
- Tipos de datos más complejos
  - Hash,
  - arrays n dim

Puppet

Herramientas: capistrano para sys-admins

¿Qué es Capistrano?

Según la Wikipedia: Capistrano is an open source tool for running scripts on multiple servers; its main use is deploying web applications. It automates the process of making a new version of an application available on one or more web servers, including supporting tasks such as changing databases. ... http://en.wikipedia.org/wiki/Capistrano

Herramientas: capistrano para sys-admins

UNIVERSIDAD DE MURCIA

```
#desc "Ejemplo uptime"
task:uptime1, :hosts => 'usuario@maquina1.um.es,
   usuario@maquina2.um.es' do
 run 'uptime'
end
$cap uptime1
# role :webs, "root@maquina1.um.es", "root@maquina2.um.es", "
   root@maquina3.um.es"
desc "Ejemplo2 uptime"
   task:uptime2, :role => :webs do
    run 'uptime'
   end
$ cap uptime2
```

Herramientas: capistrano para sys-admins

UNIVERSIDAD DE MURCIA

Shell paralelo

```
$ cap HOSTS='maquina1.um.es, maquina2.um.es' shell * executing
   'invoke'
```

Roles dinámicos

```
def getrole(role)
   f = open('/etc/manage/roles/' + role)
   f.readlines
end
role(:web) { getrole('web')
role(:db) { getrole('db') }
```

Herramientas: capistrano para sys-admins

UNIVERSIDAD DE MURCIA

¿Dónde corre?

Cliente

¿Qué puede hacer?

Lanzar un comando

Capturar salida a variables

Capturar salida tipo tail -f

Pedir interacción usuario

Subir ficheros

Bajar ficheros

Limitar por roles, host, excluir

Herramientas: capistrano para sys-admins

UNIVERSIDAD DE MURCIA

Pros

Potencia ruby (Rake – ruby make)

Excelente para actuaciones puntuales

Reproducción de tareas

Documentación de las mismas

Pero ...

Cada cosa para lo suyo

Monitorización

UNIVERSIDAD DE MURCIA

Nagios, u otras

Zabbix, Cacti, Munin,...

Elemento crucial

Importante

No sólo alertas

Sino métricas que permita ver la evolución

Necesario API's que ataquen a los estados de forma externa

Inventario

UNIVERSIDAD DE MURCIA

Inicialmente manual

wiki

OCS inventory

Tenemos en explotación

Iclassify

Basada en facter de ruby

Nodos se autoregistran

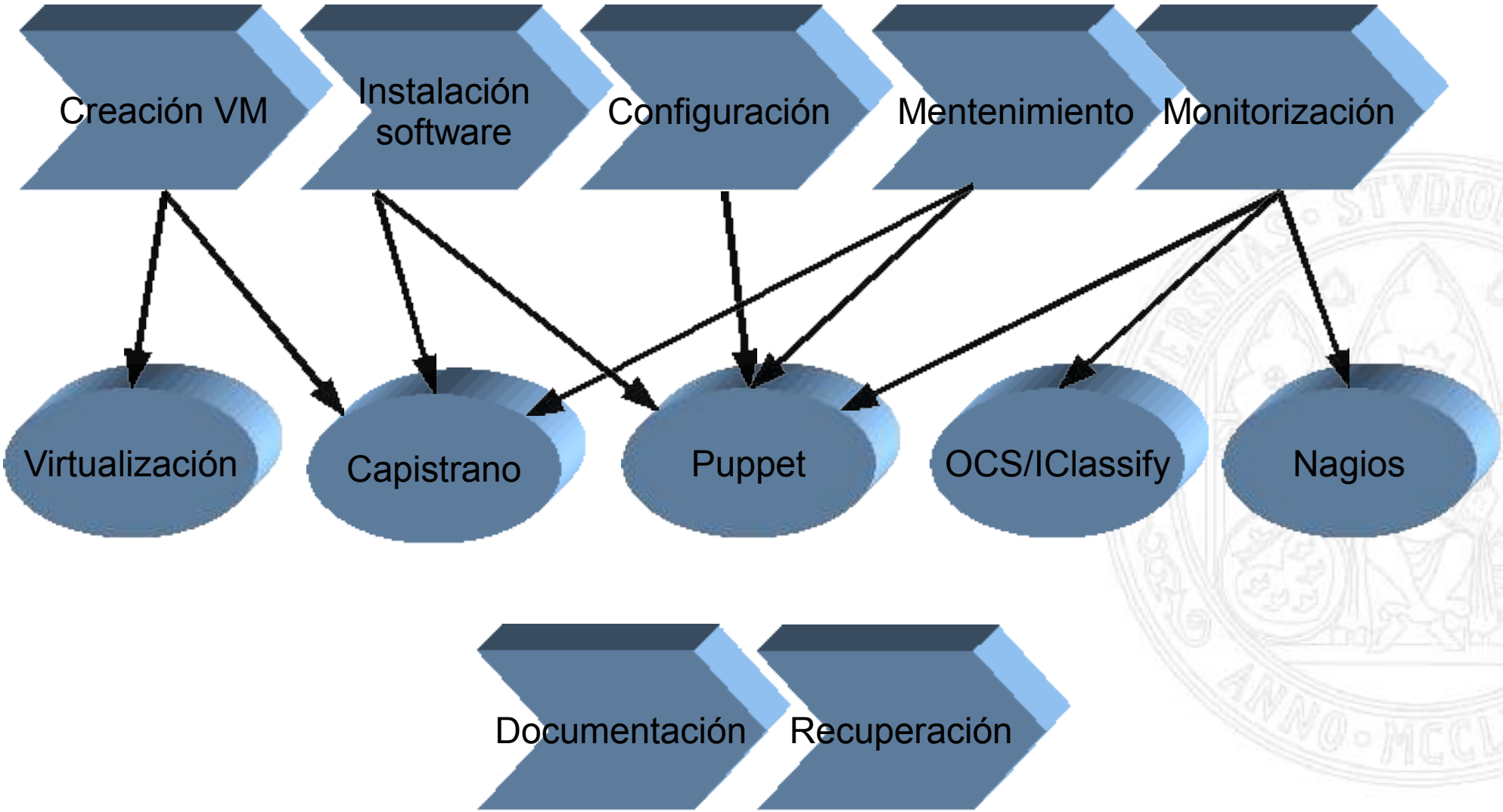
Etiquetado

Atributos manuales

¡Integrable con Capistrano!

¡Integrable con Puppet!

Automatización a lo largo del Ciclo de Vida



Ideas finales ...

UNIVERSIDAD DE MURCIA

Virtualización, nubes, etc han venido para quedarse

Aprendamos lo que hacen bien

La administración 2.0 no sólo depende de nosotros

    Desarrollos, como la propia administración

Automaticemos todo

Auto-documentación

Usa herramientas como Puppet, capistrano, etc,,,

Integrémoslas

De la ingeniería de sistemas a la ingeniería del software

Muchas Gracias
Javier García

UNIVERSIDAD DE
MURCIA