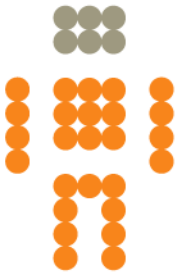
CFEngine

Distributed
Configuration
Management

Practical implementation of promise theory in CFEngine

Mikhail Gusarov
CFEngine AS
<mikhail.gusarov@cfengine.com>

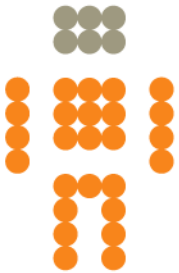


CFEngine

Distributed
Configuration
Management

Contents

- Promise theory
- "reports"
- "processes"
- "packages"
- "variables"
- Summary

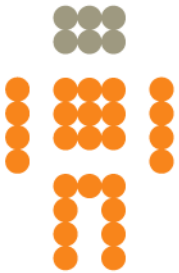


CFEngine

Distributed
Configuration
Management

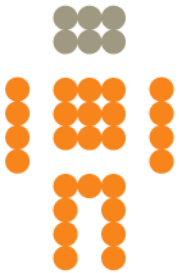
Promise theory

«Agent A promises
to agent B to fulfill C»



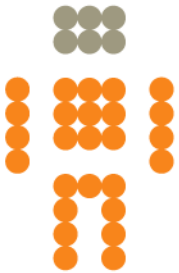
Software Implementation

- Convergency
  - Promise essence is the «desired state»
- Embracing errors
  - Promises may be kept occasionally
- Autonomy
  - Unable to rely on another agents



Real-world example: e-mail

- Convergency
  - Delivery to address
- Embracing errors
  - 4xx, 5xx errors
  - Multiple MXes
  - Mail queues
- Autonomy
  - Independent mail servers
  - Local routing decisions

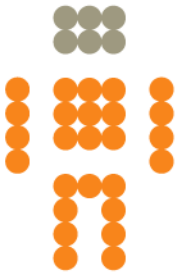


CFEngine

Distributed
Configuration
Management

Reports

```
body common control {
 bundlesequence => {"mybundle"};
}
bundle agent mybundle {
reports:
 linux::
  "Hello world!";
}
```

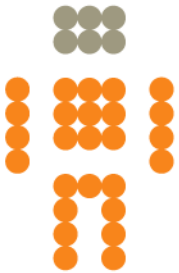


CFEngine

Distributed
Configuration
Management

Reports

```
body common control {
 bundlesequence => {"mybundle"};
}
bundle agent mybundle {
reports:
 linux::
  "Hello world!";
}
```

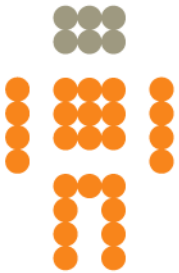


CFEngine

Distributed
Configuration
Management

Reports

```
bundle agent mybundle {
reports:
 linux::
   "Hello world!";
}
```

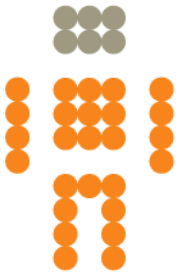


CFEngine

Distributed
Configuration
Management

Reports

```
bundle agent mybundle {
reports:
 linux::
   "Hello world!";
}
```

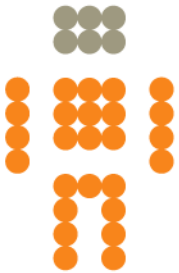


CFEngine

Distributed
Configuration
Management

Reports

```
bundle agent mybundle {
reports:
 linux::
   "Hello world!";
}
```

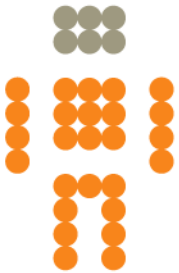


CFEngine

Distributed
Configuration
Management

Reports

```
bundle agent mybundle {
reports:
 linux::
   "Hello world!";
}
```

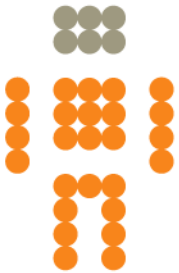


CFEngine

Distributed
Configuration
Management

Reports (translated)

```
def mybundle() {
  if "linux" in
          global_contexts {
    report("Hello world!");
  }
}
mybundle()
```

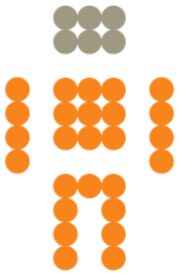


CFEngine

Distributed
Configuration
Management

Reports (run)

```
$ cf-agent
R: Hello world!
$
```

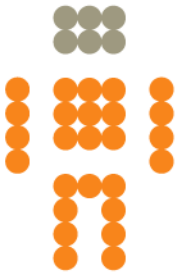


CFEngine

Distributed
Configuration
Management

Reports (run)

```
$ cf-agent
R: Hello world!
$ cf-agent
$
```

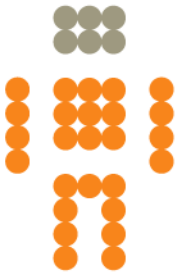


CFEngine

Distributed
Configuration
Management

Reports (run)

```
$ cf-agent
R: Hello world!
$ cf-agent
$ cf-agent
$
```

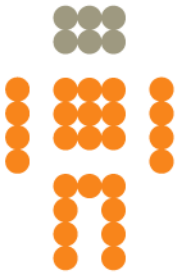


CFEngine

Distributed
Configuration
Management

Reports

```
bundle agent mybundle {
reports:
  linux::
    "Hello world!";
    "That's surprising";
}
```

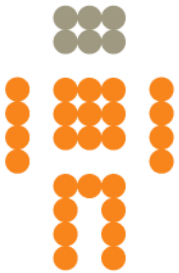


CFEngine

Distributed
Configuration
Management

Reports (run)

```
$ cf-agent
R: Hello world!
$ cf-agent
$ cf-agent
$ cf-agent
R: That's surprising
$
```



CFEngine

Distributed
Configuration
Management

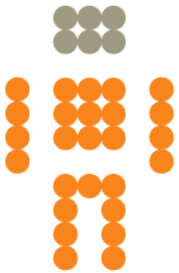
Reports (run)

```
$ cf-agent
R: Hello world!
$ cf-agent
$ cf-agent
$ cf-agent
R: That's surprising
$ cf-agent
R: Hello world!
$
```

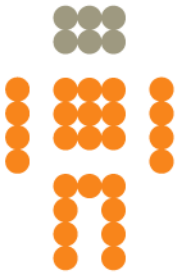
CFEngine

Distributed
Configuration
Management

```
def mybundle() {
  promises.add(
    report("Hello world!",
           activate = λ:
     "linux" in
     global_contexts))
}
mybundle()
run_all_relevant_promises()
```

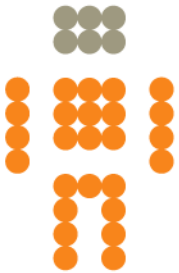


«Ensure that specified message is reported to sysadmin as soon as possible, but no more than once per 5 minutes to the sysadmin using the specified reporting mechanisms»



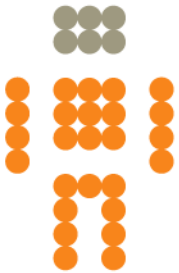
Convergence

«Ensure that specified message is **reported** to sysadmin as soon as possible, but no **more than once per 5 minutes** to the sysadmin using the specified reporting mechanisms»



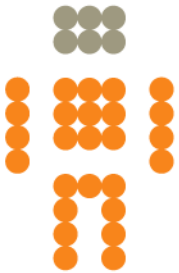
Embracing errors

«Ensure that specified message is reported to sysadmin **as soon as possible**, but no more than once per 5 minutes to the sysadmin using the specified reporting mechanisms»



Autonomy

«Ensure that specified message is reported to sysadmin as soon as possible, but no more than once per 5 minutes to the sysadmin **using the specified reporting mechanisms**»

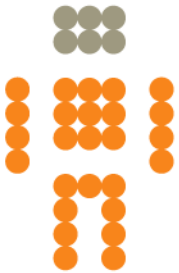


CFEngine

Distributed
Configuration
Management

Processes

```
bundle agent myprocesses {
processes:
 "nginx"
  match_range => irange("1","20");
  restart_class => "restart_nginx";
commands:
 restart_nginx::
  "/etc/init.d/nginx start";
}
```

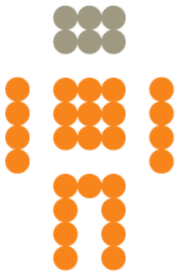


CFEngine

Distributed
Configuration
Management

Processes

```
bundle agent myprocesses {
processes:
 "nginx"
   match_range => irange("1","20");
   restart_class => "restart_nginx";
commands:
 restart_nginx::
   "/etc/init.d/nginx start";
}
```

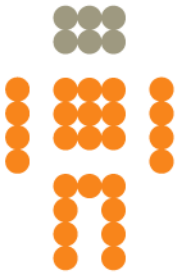


CFEngine

Distributed
Configuration
Management

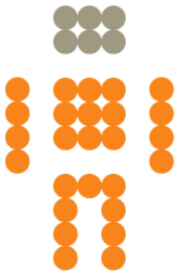
Processes

```
bundle agent myprocesses {
processes:
 "nginx"
  match_range => irange("1","inf");
  restart_class => "restart_nginx";
commands:
 restart_nginx::
  "/etc/init.d/nginx start";
}
```



CFEngine

Distributed
Configuration
Management

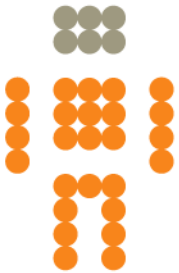


```
def myprocesses() {
 nginx_cnt = count_proc("nginx")
 if nginx_cnt == 0 {
  promises.add(
   command(
     "/etc/init.d/nginx start"))
 } elif nginx_cnt > 20  {
    Promises.add(kill("nginx",
                      nginx_cnt – 20))
 }
}
myprocesses()
run_all_relevant_promises()
```

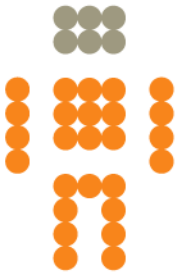
CFEngine

Distributed
Configuration
Management

```
def myprocesses() {
 promises.add(
  processes("nginx",
   λ count: if count == 0 {
    define_context("restart_nginx")})
 promises.add(
  commands(
   "/etc/init.d/nginx start",
  activate_if = λ:
   "restart_nginx" in global_contexts))
}
myprocesses()
run_all_relevant_promises()
```



«Ensure that amount of specified processes is kept in specified range, killing extra processes and communicating the need to restart if necessary»

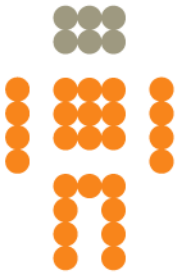


CFEngine

Distributed
Configuration
Management

Convergence

«Ensure that amount of specified processes is kept **in specified range**, **killing extra processes and communicating the need to restart** if necessary»

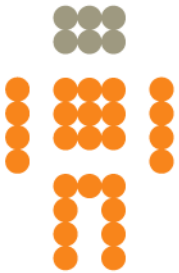


CFEngine

Distributed
Configuration
Management

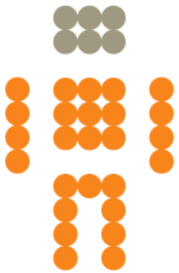
Embracing errors

«Ensure that amount of specified processes is kept in specified range, killing extra processes and communicating the need to restart if necessary»



Autonomy

«Ensure that amount of specified processes is kept in specified range, killing extra processes and communicating the need to restart if necessary»

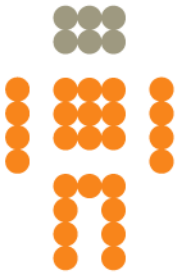


CFEngine

Distributed
Configuration
Management

Packages

```
bundle agent mypackages {
packages:
 "libapache2-mod-wsgi"
  package_method => apt,
  package_policy => "addupdate",
  package_version => "3.3-4",
  package_select => ">=";
}
```

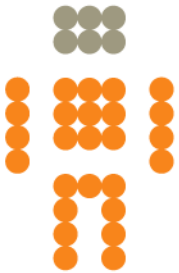


CFEngine

Distributed
Configuration
Management

Packages

```
bundle agent mypackages {
packages:
 "libapache2-mod-wsgi"
   package_method => apt,
   package_policy => "addupdate",
   package_version => "3.3-4",
   package_select => ">=";
}
```

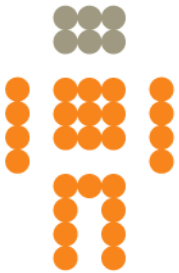


CFEngine

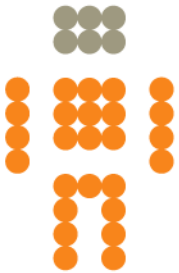
Distributed
Configuration
Management

Packages (translated)

(Implementation is too large to fit in the ~~margin~~slide)

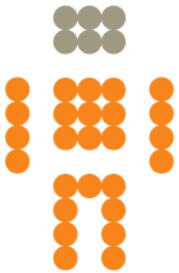


«Ensure that specified package is installed or not installed according to the constraints specified. Do this by instructing local package manager to add, upgrade or remove package»



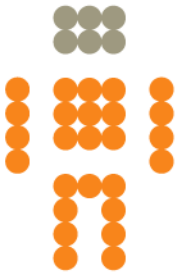
Convergent

«Ensure that specified package is installed or not installed according to the constraints specified. Do this by instructing local package manager to add, upgrade or remove package»



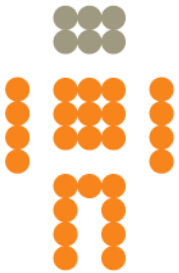
Embracing errors

«Ensure that specified package is installed or not installed according to the constraints specified. Do this by instructing local package manager to add, upgrade or remove package»



Autonomous

«Ensure that specified package is installed or not installed according to the constraints specified. Do this by instructing local package manager to add, upgrade or remove package»

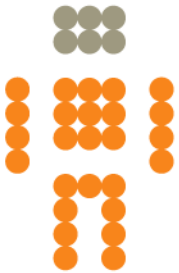


CFEngine

Distributed
Configuration
Management

Variables

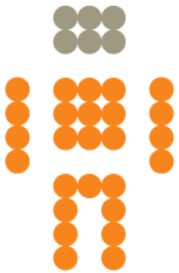
```
bundle agent myvars {
 vars:
   "a" string =>
          "My $(string)";
}
```



CFEngine

Distributed
Configuration
Management

«Maintain in-agent binding of left-side name to the value of right-side expression»

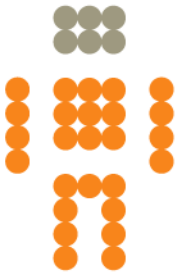


CFEngine

Distributed
Configuration
Management

Convergent, autonomous

«Maintain in-agent binding of left-side name to the value of right-side expression»

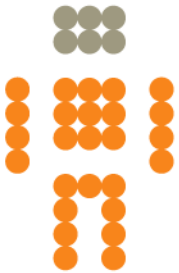


CFEngine

Distributed
Configuration
Management

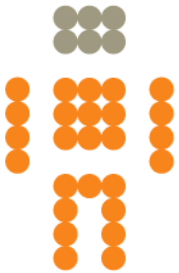
Embracing errors

```
bundle agent myvars {
vars:
 "myvar" string => execresult(
    "/usr/bin/testparm
     /etc/smb/smb.conf",
    "noshell");
}
```

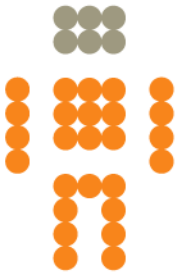


Summary

- CFEngine building blocks are promises
  - Higher-level
  - Convergent
  - Error-embracing
  - Autonomous
  - Not the things from conventional languages



Questions?



DSL vs. eDSL

- Pro eDSL:
  - Familiar host language
- Contra eDSL:
  - Radically different building blocks
  - Different control structure and evaluation

→ DSL