Metodologie e strumenti a supporto dello sviluppo di soluzioni enterprise

Bologna, 05/06/2015

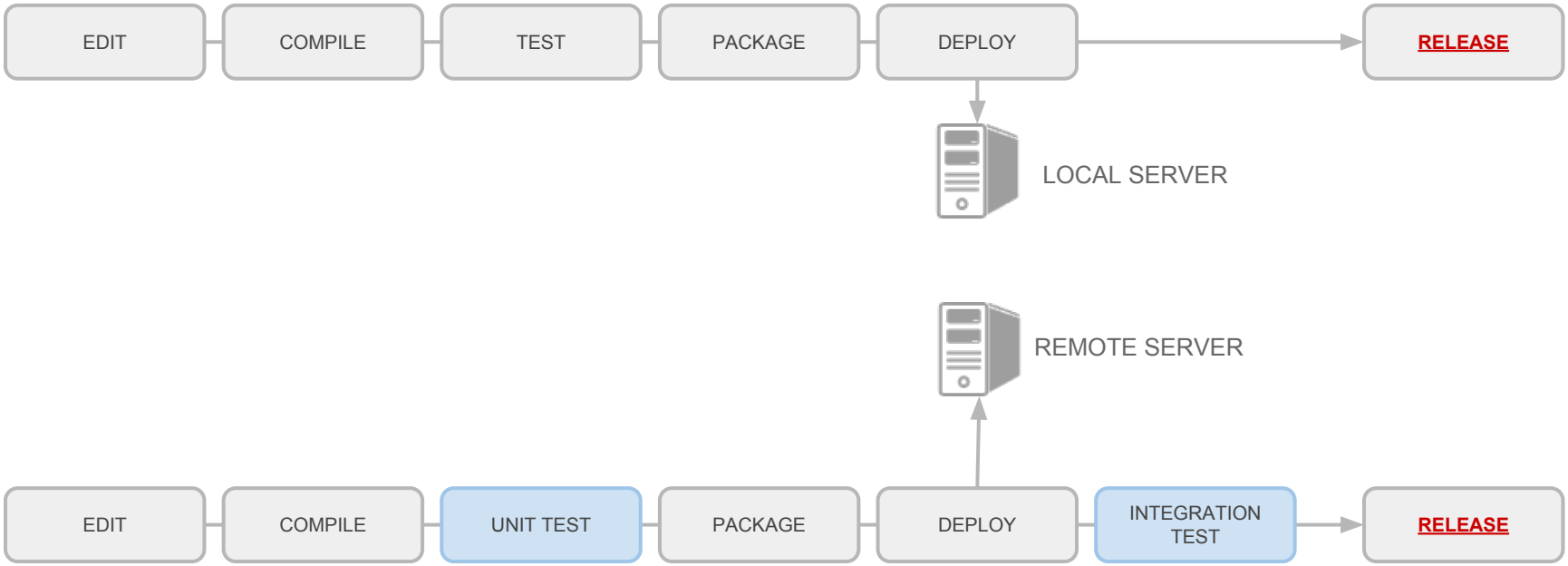
Stefano Monti
stefano.monti@epocaricerca.it
www.epocaricerca.it

Agenda

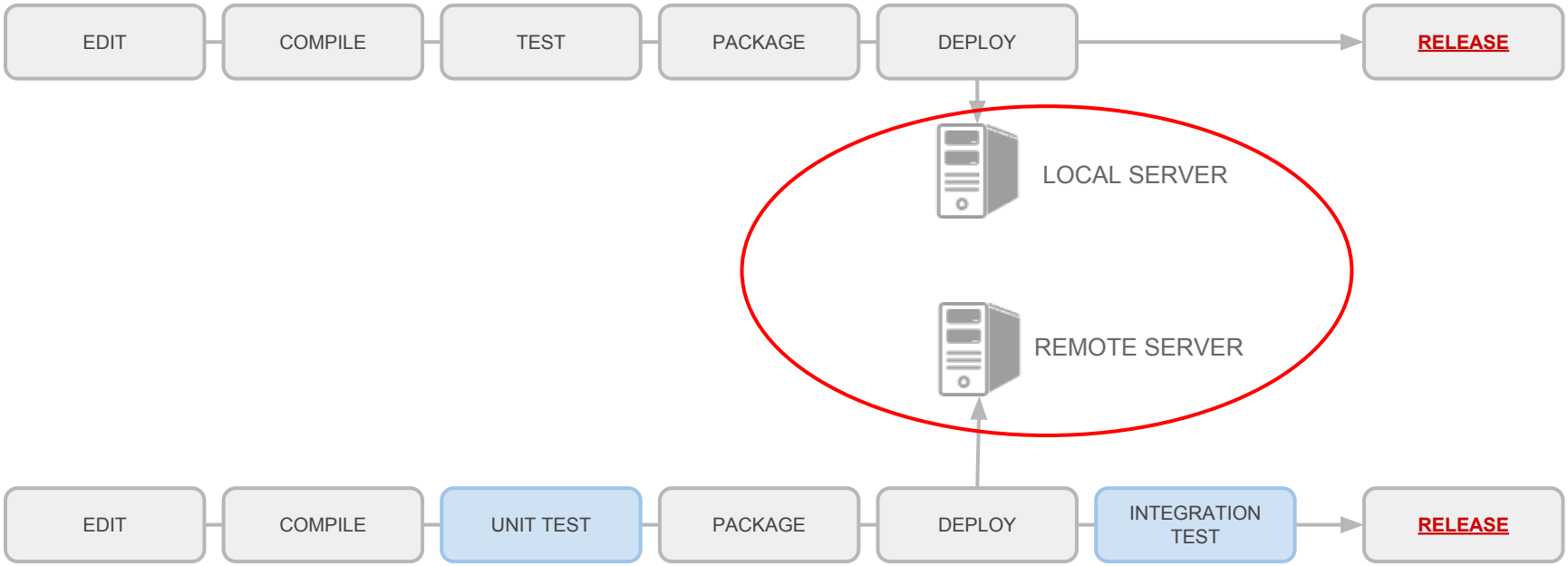
1. Continuous Integration & Continuous Delivery
2. Seamless local/cloud development
3. The mobile (r-)evolution

1. Continuous Integration & Continuous Delivery

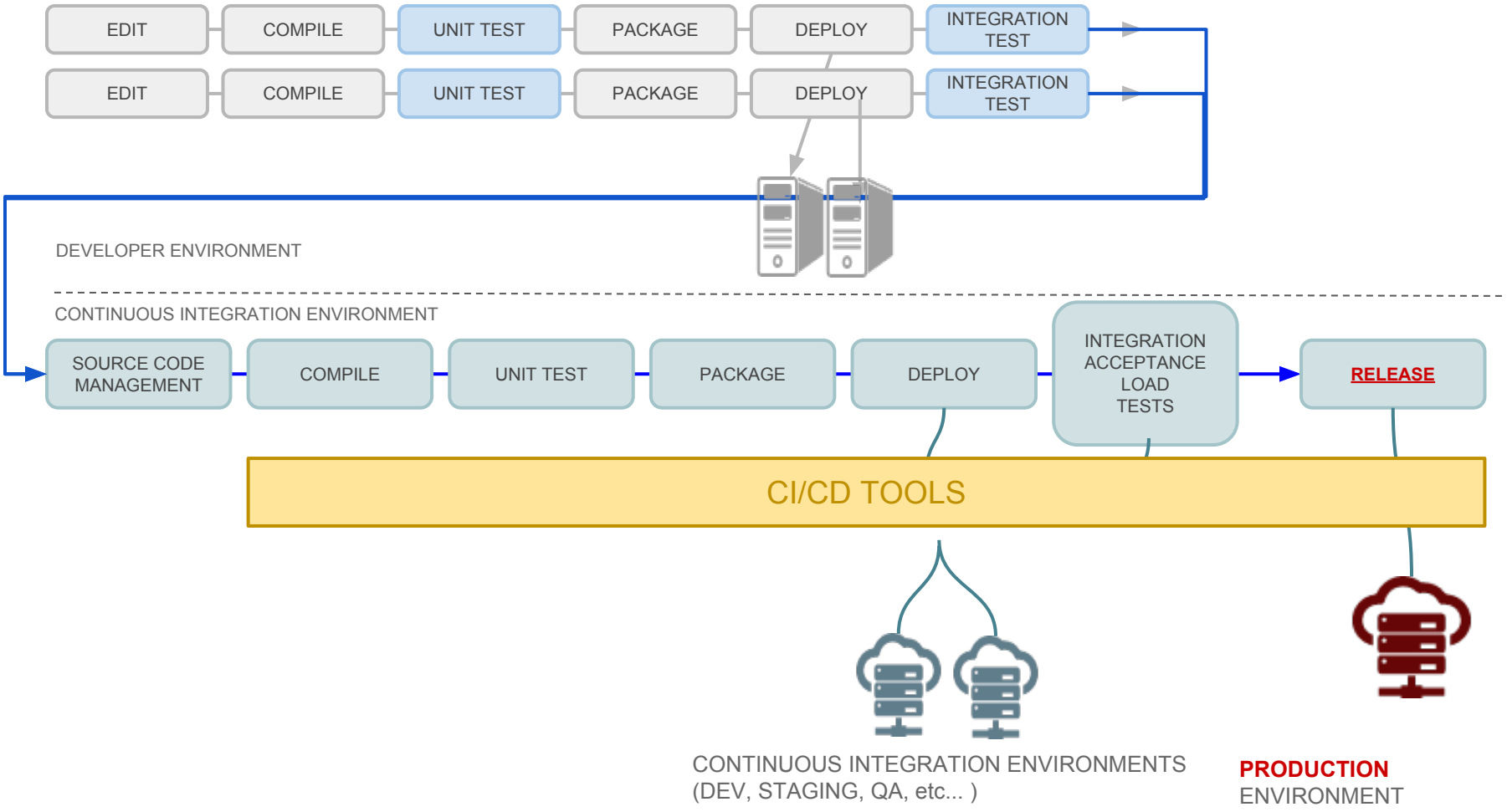
Traditional development pipeline



Traditional development pipeline



CI/CD pipeline (simplified)



Continuous Integration & Continuous Delivery

- adozione di strumenti di Source Code Management (es Mercurial, Git, SVN, ...)
- adozione di strumenti di automazione di **test distribuiti**
- ambienti di test **uguali** a quelli di produzione
- cloud (IaaS, PaaS) come ambiente ideale per
  - scalare risorse
  - replicare ambienti
- ogni build è teoricamente “production-ready”

CI/CD - benefici

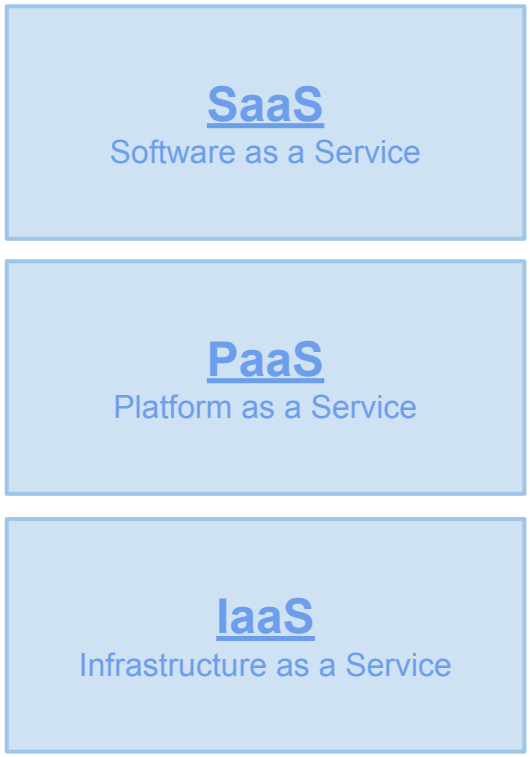
- **immediata** integrazione sorgenti e **risoluzione conflitti**

- **automatizzazione test di alto livello**
  - integration
  - performance & load
  - user acceptance

- **replicabilità e garanzia** risultati tramite procedure di build e test in **ambiente server standard** (tipicamente **cloud**)

2. Seamless local/cloud development

Cloud computing



Cloud computing - soluzione o problema?

Soluzione

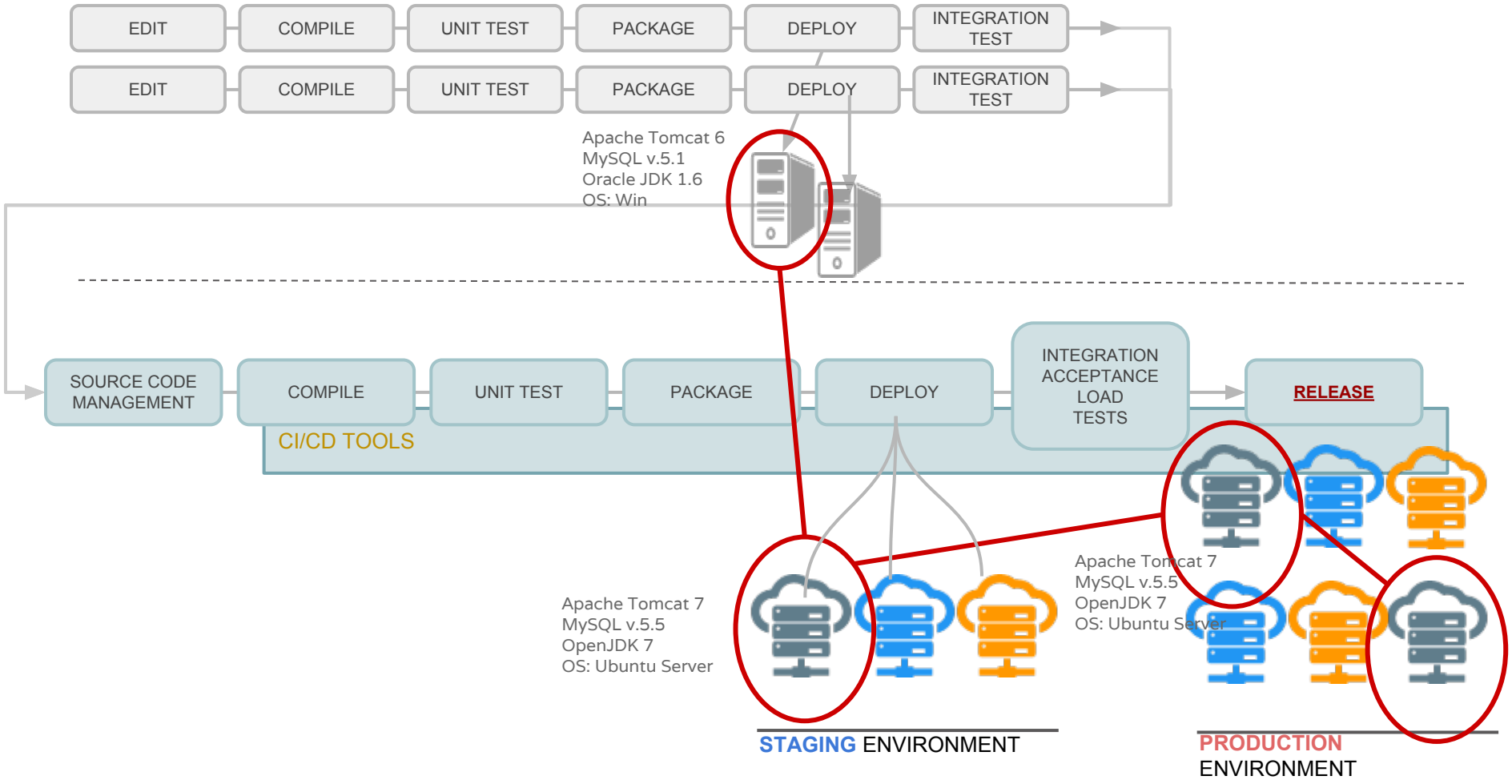
1. disponibilità **on demand**
2. **scalabilità** e flessibilità
3. **virtualizzazione**, **affidabilità** e replica

Cloud computing - soluzione o problema?

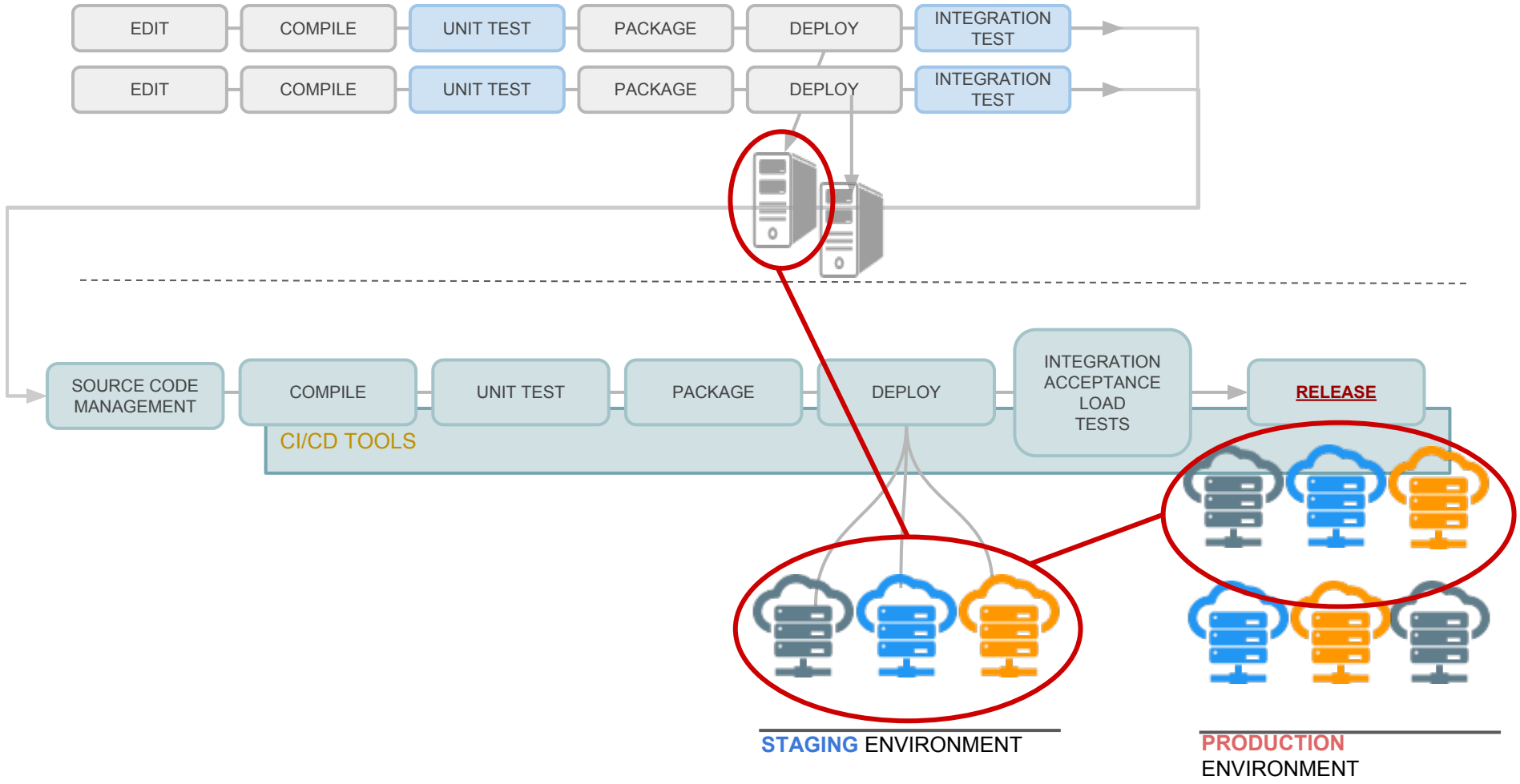
Problema: come gestire *scale up* dell'infrastruttura?

Problema: come gestire *controllo* dell'infrastruttura?

Garanzia di configurazione



Automazione setup infrastruttura



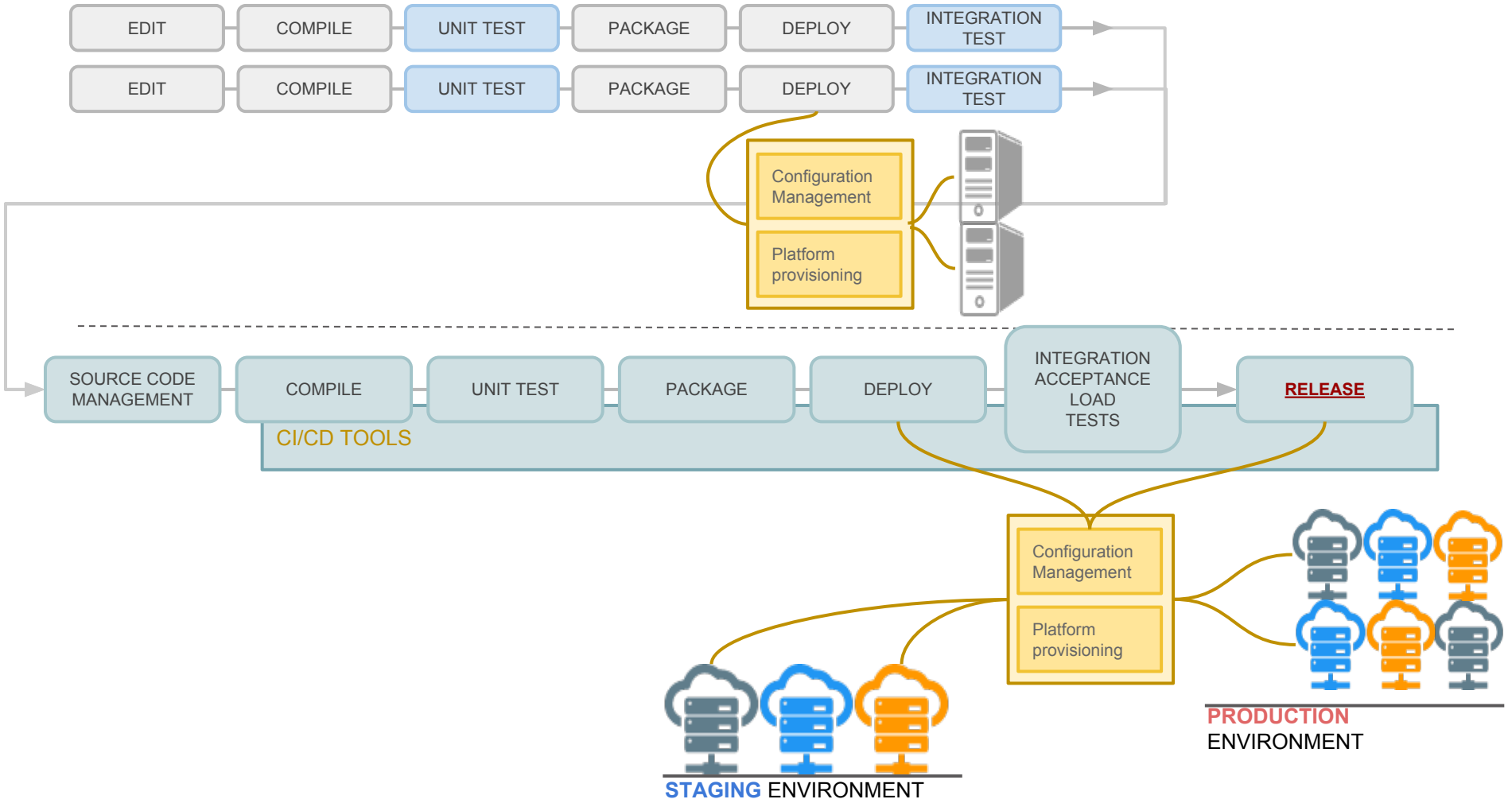
Cloud computing - soluzione o problema?

Problema: come gestire *scale up* dell'infrastruttura?

1. gestione infrastrutture complesse e configurazione server
   → Configuration Management (Chef, Puppet, …)

2. controllo e riproducibilità locale/remota
   → Platform Provisioning tools (e.g. Vagrant, Vortex)

Automazione setup infrastruttura



Configuration management tools

Obiettivo: **facilitare la configurazione** ambienti server (locali o in cloud)

1. **automatizzazione** della configurazione server
2. approccio **dichiarativo e/o procedurale**
3. **integrazione** in procedure di build

Configuration management tools

```
class mysql::server {

  package { "mysql-server": ensure => installed }
  package { "mysql": ensure => installed }

  service { "mysqld":
    enable => true,
    ensure => running,
    require => Package["mysql-server"],
  }

  file { "/var/lib/mysql/my.cnf":
    owner => "mysql", group => "mysql",
    source => "puppet:///mysql/my.cnf",
    notify => Service["mysqld"],
    require => Package["mysql-server"],
  }

  file { "/etc/my.cnf":
    require => File["/var/lib/mysql/my.cnf"],
    ensure => "/var/lib/mysql/my.cnf",
  }

  exec { "set-mysql-password":
    unless => "mysqladmin -uroot -p$mysql_password status",
    path => ["/bin", "/usr/bin"],
    command => "mysqladmin -uroot password $mysql_password",
    require => Service["mysqld"],
  }
```

```
package { 'openssh-server':
        ensure => installed,
}
file { '/etc/ssh/sshd_config':
        source => 'puppet:///modules/sshd/
        sshd_config',
        owner => 'root',
        group => 'root',
        mode => '640',
        notify => Service['sshd'],
        require => Package['openssh-server'],
}
service { 'sshd':
        ensure => running,
        enable => true,
        hasstatus => true,
        hasrestart => true,
}
```

Puppet configuration file excerpt
http://docs.puppetlabs.com/

Platform provisioning tools

Obiettivo: **istanziare infrastrutture server complesse**

1. integrazione con Configuration Management tools
2. **astrazione infrastruttura locale/cloud tramite Virtual Machine**

Vagrant

Platform Provisioning tool che astrae:

- **infrastructure provider**: supporto trasparente a provider di Virtual Machine
  - locali - Virtualbox, VMWare
  - remote - Amazon EC2, Openstack

- **infrastructure provisioning**: supporto a Configuration Management tools e/o script a riga di comando

Vagrant

```
$ vagrant init ubuntu/trusty64
$ vagrant up --provider virtualbox
$ vagrant ssh
```

```
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure("2") do |config|
   config.vm.box =”ubuntu/trusty64”
end
```

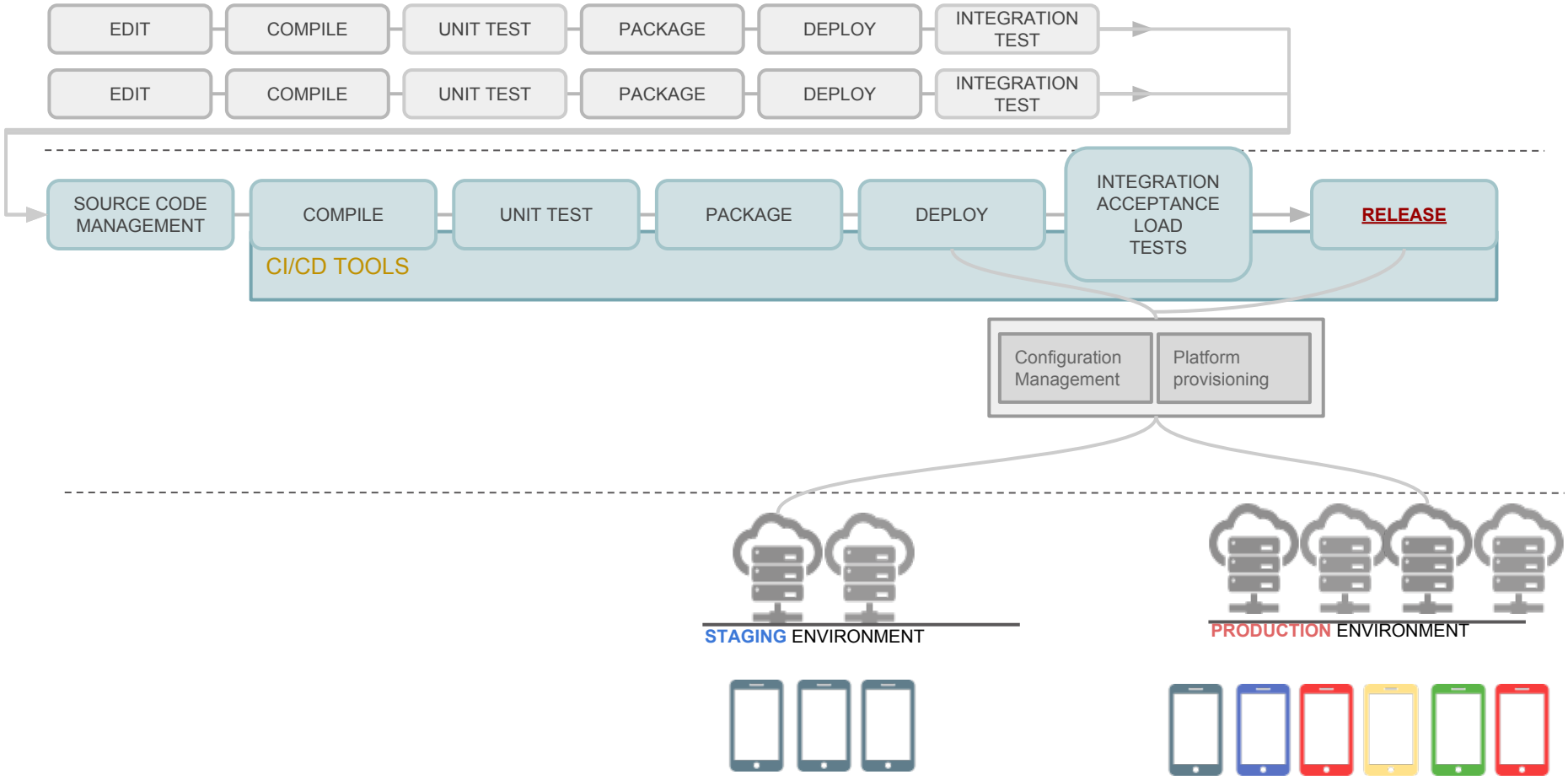
```
$ vagrant init ubuntu/trusty64
$ vagrant up --provider aws
$ vagrant ssh
```

```
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure("2") do |config|
   config.vm.provider :aws do |aws, override|
   aws.access_key_id = "XXX"
   aws.secret_access_key "YYY"
   aws.keypair_name = "key.pem"
   aws.region = "eu-west-1"
   aws.ami = "ami-8e987ef9"
   aws.instance_type="m1.medium"
   aws.tags = {
     'Name' => 'MyFirstVagrantServer',
   }

 end
```

Vagrant examples
http://docs.vagrantup.com/v2/getting-started/index.html

3. The mobile (r)evolution

Mobile landscape



Mobile landscape

- **eterogeneità device**
  - differenti S.O. e differenti versioni
  - differenti produttori e librerie proprietarie
  - differente hardware (es. risoluzione schermo, connettività, ecc…)

- gestione "**deploy**" su mobile

- approccio sempre più **fat client**
  - logica di processamento e interazione su device
  - backend server (relativamente) meno centrali

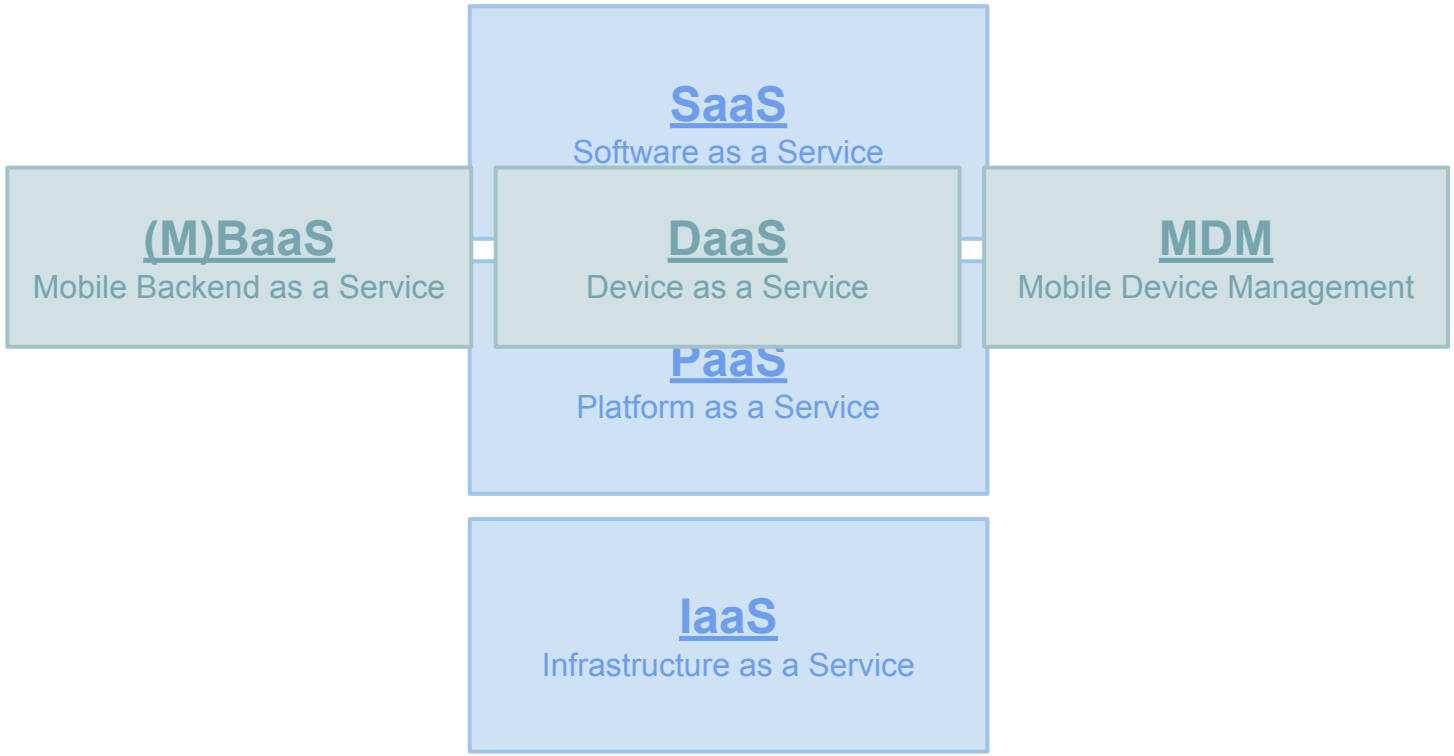
Mobile landscape

- **eterogeneità** device → infrastrutture **Device as a Service** per **testare in cloud su dispositivi fisici**

- gestione "**deploy**" su mobile → infrastrutture **Mobile Device Management** per gestire e coordinare piani di rilascio

- approccio sempre più **fat client** → infrastrutture **Mobile Backend as a Service** per prototipare e realizzare **logica di backend in cloud in maniera semplificata**

Mobile landscape



Mobile Device Management

Strumenti a supporto della **gestione flotte di dispositivi mobile**

- **installazione/rimozione** app (anche **selettivo)**
- **aggiornamento** app (anche **selettivo)**
- **multi-OS (Android, iOS, Windows)**
- gestione **sicurezza dispositivo**
  - **blocco** e/o **wipe** in caso di furto
  - **cifratura** supporti di memoria
  - definizione **profili** d'uso
- modalità **cloud** (SaaS) oppure **on-premise**
- strumenti *open source* (es. WSO2 EMM, OpenMEAP) o commerciali (es. CISCO Meraki, Airwatch, Apple Profile Manager, ecc...)

Mobile Backend as a Service

Strumenti a supporto della **prototipazione e realizzazione rapida di backend**

- ideali per **realizzare app con modelli fortemente fat client** (es. backend solo per memorizzazione remota) con supporto per
  - cloud database
  - push notifications
  - analytics
- modalità **cloud** (SaaS) e/o **on-premise**
- **supporto multi-OS**
- strumenti *open source* (es.Baasbox) o commerciali (es. Kinvey, Parse, Firebase, ecc...)

Device as a Service

Strumenti a supporto del **testing rapido su dispositivi mobile**

- possibilità di testare su numero virtualmente illimitato di dispositivi fisici ed OS
- modalità **cloud**
- **supporto multi-OS**
- strumenti commerciali (es. AppThwack, ecc...)

Questions?

Stefano Monti
stefano.monti@epocaricerca.it
www.epocaricerca.it