Luca Cabibbo

Architettura dei Sistemi Software

Gestione di ambienti

dispensa asw630

ottobre 2024

Provisioning new servers is a manual, repetitive, resource-intensive, and error-prone process. Exactly the kind of problem that can be solved with automation.

Jez Humble and David Farley

- Riferimenti

- Luca Cabibbo. **Architettura del Software**: Strutture e Qualità. Edizioni Efesto, 2021.
  - Capitolo 36, Gestione di ambienti
- Humble, J. and Farley, D. **Continuous Delivery**: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley, 2010.
- Bass, L., Weber, I., and Zhu, L. **DevOps**: A Software Architect's Perspective. Addison-Wesley, 2015.
- Nygard, M. **Release It!**: Design and Deploy Production-Ready Software, second edition. Pragmatic Bookshelf, 2018.

- Obiettivi e argomenti

- Obiettivi
  - presentare gli ambienti di esecuzione e le attività legate alla loro gestione
  - introdurre la gestione automatizzata degli ambienti e delle configurazioni
- Argomenti
  - introduzione
  - ambienti
  - deployment e provisioning
  - gestione automatizzata di ambienti
  - gestione automatizzata di server e ambienti fisici
  - gestione automatizzata di server e ambienti virtuali
  - pets vs cattle
  - discussione

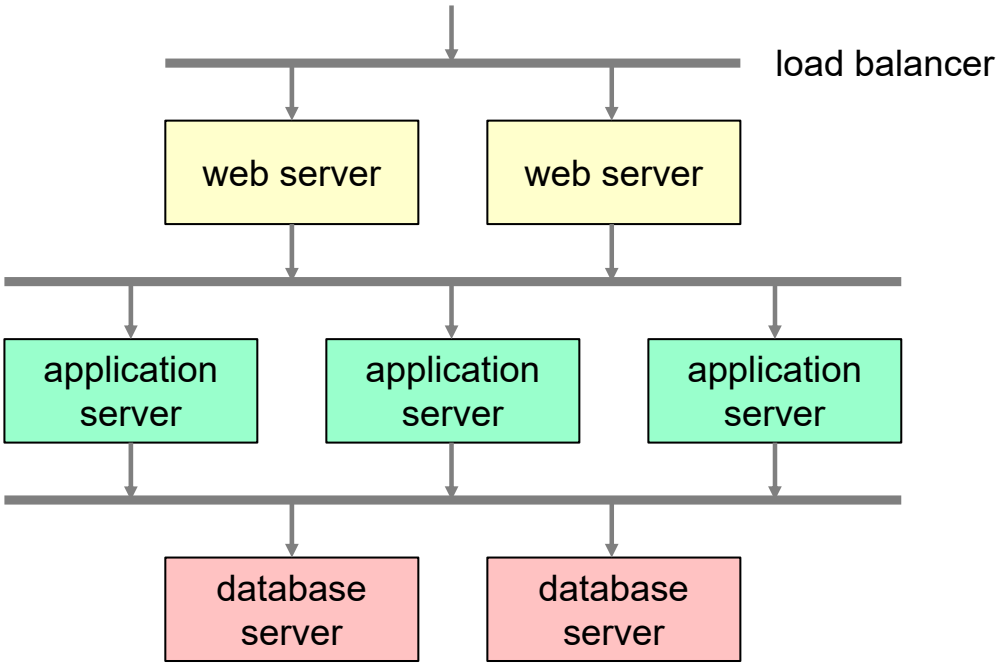
* Introduzione

- La delivery del software ha lo scopo di rilasciare il software di interesse (o una sua nuova versione) agli utenti finali
  - il processo di delivery comprende diverse attività – build, deployment, test e release
    - il deployment comprende il provisioning (la preparazione) dell'ambiente di esecuzione
    - l'ambiente di esecuzione è composto dalle risorse di calcolo (hardware e software) necessarie per poter eseguire il software
  - presentiamo gli ambienti di esecuzione e la gestione di ambienti – nel contesto del deployment e del provisioning

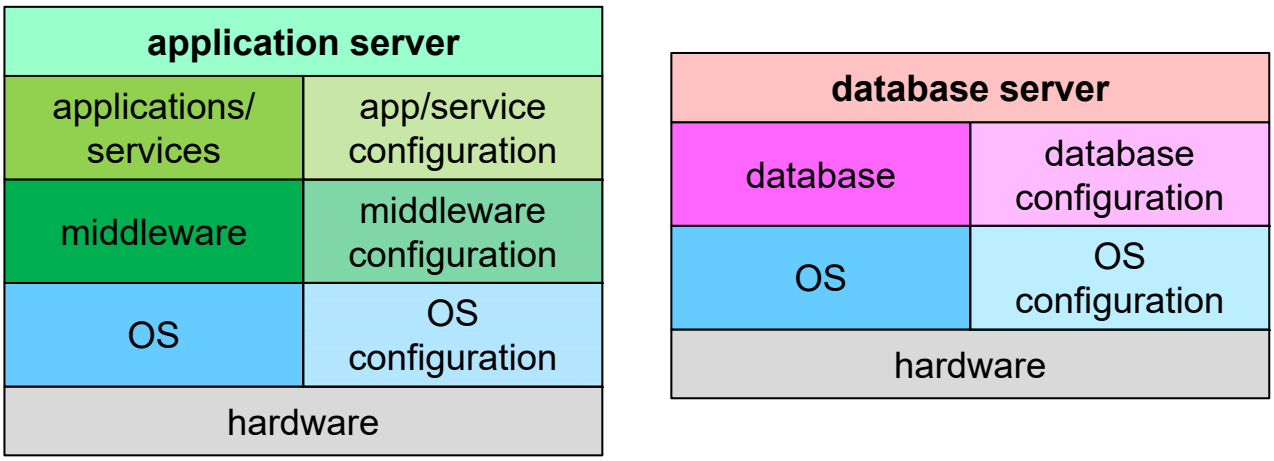
* Ambienti

- Un **ambiente** (*environment*) di esecuzione per un sistema software comprende tutte le risorse di calcolo (hardware e software) necessarie per poter eseguire il sistema software, insieme alle loro configurazioni
  - le risorse hardware (fisiche o virtuali, on premises o nel cloud) comprendono i nodi, lo storage e l'infrastruttura di rete – con le loro configurazioni
  - le risorse software comprendono (per ciascun nodo) l'OS, il middleware e tutto il software di supporto – con le loro configurazioni (ed eventuali dati)
  - in pratica, un ambiente deve essere in grado di eseguire il software di interesse in modo auto-contenuto
    - tranne le dipendenze da specifiche entità e servizi esterni – ad es., Google Maps

Ambiente: un esempio



Ambiente: un esempio

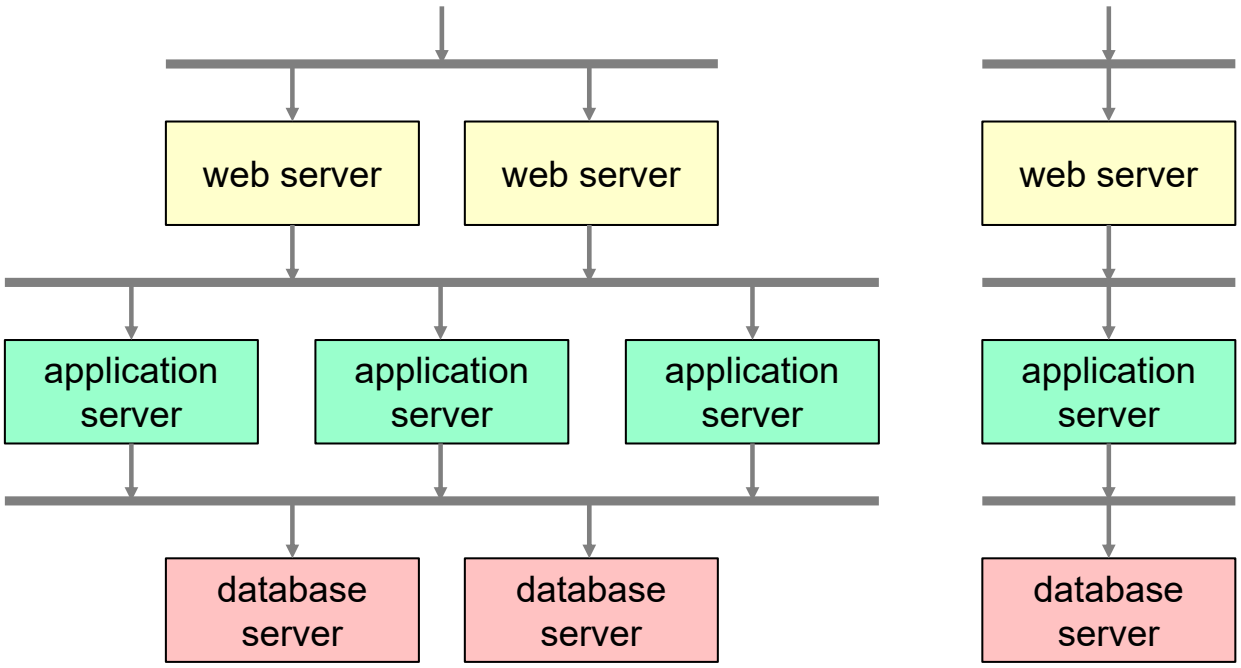


Un sistema software, tanti ambienti

- Un sistema software richiede in genere più ambienti
  - l'**ambiente di produzione**
  - l'ambiente di sviluppo
  - uno o più ambienti di test
- I diversi ambienti per un sistema software sono simili tra loro (tranne quello di sviluppo) – e hanno la struttura richiesta dall'ambiente di produzione
  - per semplicità non consideriamo l'ambiente di sviluppo – e discutiamo la definizione e la gestione di un solo ambiente

Un sistema software, tanti ambienti

- L'ambiente di produzione
- Un ambiente di test



Un sistema software, tanti ambienti

- I diversi ambienti per un sistema software vanno tenuti separati
  - l'ambiente di produzione deve essere isolato rispetto agli ambienti di sviluppo e test
  - l'assenza di risorse condivise modificabili è necessaria per l'isolamento (ma non sempre è sufficiente)
  - eventuali entità esterne modificabili vanno accedute solo dall'ambiente di produzione

Opzioni per le risorse di calcolo

- Ci sono numerose opzioni per le risorse di calcolo che costituiscono un ambiente
  - opzioni per i nodi
    - server fisici
    - macchine virtuali
    - container
  - opzioni di localizzazione
    - in un proprio data center (on premises)
    - nel cloud (cloud computing)
  - ulteriori opzioni
    - piattaforme
    - elaborazione serverless (“senza server”)

Infrastruttura

- L’**infrastruttura** (*infrastructure*) di un’organizzazione comprende tutti gli ambienti gestiti dall’organizzazione, insieme ai servizi infrastrutturali per sostenerli – ad es., server DNS, firewall, repository per il controllo delle versioni, …
  - ci concentriamo soprattutto sulla gestione degli ambienti

* Deployment e provisioning

- La gestione degli ambienti è di interesse soprattutto nel contesto del deployment di un sistema software
  - il *deployment* riguarda l'installazione del software di interesse in un opportuno ambiente di esecuzione
    - il deployment può richiedere la preparazione (provisioning) dell'ambiente
  - il *provisioning* riguarda l'acquisizione e la configurazione di un ambiente
    - viene acquisito l'hardware (fisico o virtuale), in ciascun server viene installato e configurato lo stack software richiesto (OS e middleware), viene configurata la rete, ecc.

Provisioning

- In generale, il termine *provisioning* (approvvigionamento, fornitura) indica il processo di preparazione ed equipaggiamento di un insieme di risorse o di un intero sistema per consentirne l'accesso da parte dei suoi utenti
- Alcune accezioni specifiche
  - hardware provisioning
  - server provisioning
  - software provisioning
  - environment provisioning

- Deployment manuale

- Il deployment – compreso il provisioning – viene spesso svolto in modo manuale
  - i nodi o server dell'ambiente di esecuzione vengono preparati individualmente e "a mano"
    - in ciascun nodo o server viene installato "a mano" l'OS e il software di supporto richiesto, copiando o creando i relativi file di configurazione, e copiando eventuali dati di interesse
    - questo può richiedere decine o centinaia di passi da eseguire per ciascuno dei server
  - questo è un antipattern
    - è un'attività lunga, complessa e soggetta a errori – perché ci sono molte cose che possono andare male
    - se viene commesso un errore, il software non funzionerà – e sarà difficile capire quale errore è stato commesso

Il deployment manuale è un antipattern

- Alcuni segni e inconvenienti del deployment manuale
  - c'è un documento che descrive in modo dettagliato i passi da eseguire – scritto "a mano" e difficile da comprendere (anche da chi l'ha scritto)
  - non scala con molti server
  - richiede dei test manuali per confermare che l'applicazione è in esecuzione
  - in caso di problemi, la diagnosi è difficile
  - il rilascio del software richiede spesso più di pochi minuti – anzi, può durare qualche ora o anche diversi giorni
  - è difficile ripristinare una versione precedente funzionante del sistema
  - il rilascio di una nuova versione del software viene vissuto come un incubo – e viene effettuato il meno possibile

- Deployment automatizzato

- Il rilascio del software viene sempre più comunemente svolto in modo automatizzato – anche grazie al supporto fornito da opportuni strumenti software di automatizzazione
  - è possibile automatizzare provisioning e deployment – nonché l'intero processo di delivery del software
  - l'automazione è basata sulla modellazione degli ambienti, mediante un approccio di tipo *infrastructure-as-code*
  - l'automazione è possibile sia per server e ambienti fisici che per server e ambienti virtuali, nonché nel cloud
  - in pratica, il deployment del software può essere effettuato selezionando la versione del software da utilizzare nonché l'ambiente di esecuzione richiesto – e premendo un pulsante "deploy"

Deployment automatizzato

- Vantaggi del deployment automatizzato del software
  - il deployment del software è un processo ripetibile
  - la ripetibilità sostiene la consistenza e l'affidabilità
  - di solito può essere svolto in pochi minuti
  - il rilascio di nuove versioni del software non viene più vissuto come un incubo
  - sono possibili rilasci frequenti – sono possibili anche migliaia di rilasci ogni giorno
  - il rilascio automatizzato del software dovrebbe essere l'unico modo con cui rilasciare sistemi software complessi e/o critici

* Gestione automatizzata di ambienti

- La gestione (provisioning) di un ambiente (o di un'intera infrastruttura) comprende sia la sua creazione che la sua manutenzione ed evoluzione
  - la gestione automatizzata degli ambienti può risolvere molti problemi della gestione manuale
- La gestione automatizzata degli ambienti si basa su tre principi
  - uso di una configurazione eseguibile (*infrastructure-as-code*) per specificare lo stato desiderato di un ambiente
  - l'ambiente deve essere *autonomico* (*autonomic*) – deve essere in grado di aggiornarsi automaticamente allo stato desiderato
  - uso di *monitoraggio* e strumentazione per poter conoscere sempre lo stato attuale dell'ambiente

Benefici

- Alcuni benefici della gestione automatizzata degli ambienti
  - la creazione automatizzata degli ambienti e la loro manutenzione autonomica assicura, ad es., che in caso di problemi sia possibile ricostruire un ambiente in modo prevedibile e rapido
  - supporta la creazione di ambienti di test con una struttura fedele a quella dell'ambiente di produzione (nuovo o modifica di quello attuale) – e dunque supporta la verifica degli aggiornamenti del sistema
  - una "configurazione eseguibile" è preferibile a qualunque altra forma di documentazione dell'ambiente
  - rende più semplice ricostruire un ambiente che non "ripararlo"

Modellazione di ambienti

- Una caratteristica fondamentale della gestione automatizzata degli ambienti è la modellazione e specifica dettagliata di tutte le informazioni di configurazione di un ambiente e dei suoi elementi
  - nella gestione di un ambiente (e di ciascun suo nodo) bisogna considerare molti aspetti
    - l'OS e la sua configurazione
    - lo stack software del middleware – librerie, application server, database server, message broker, … – e la loro configurazione
    - software di supporto dell'infrastruttura – sistema per il controllo delle versioni, repository per altri elaborati, servizi di directory, sistemi di monitoraggio, …
    - punti esterni di integrazione – sistemi e servizi esterni
    - infrastruttura di rete – router, switch, firewall, DNS, DHCP, …

Modellazione di ambienti

- Una caratteristica fondamentale della gestione automatizzata degli ambienti è la modellazione e specifica dettagliata di tutte le informazioni di configurazione di un ambiente e dei suoi elementi
  - ad es., due installazione di uno stesso OS (o di un servizio di middleware) possono differire in moltissimi modi – le impostazioni e i parametri scelti possono influenzare in modo significativo il modo in cui il sistema software viene eseguito
  - in generale, la scelta delle impostazioni di default (per un OS o un servizio di middleware) è di solito inappropriata
    - ad es., l'OS richiede la configurazione dei servizi abilitati, degli utenti e del controllo degli accessi
    - in un message broker devono essere configurati i canali per messaggi e le sottoscrizioni

Modellazione di ambienti

- In pratica, le informazioni di interesse per sostenere la gestione automatizzata di un ambiente comprendono almeno
  - le definizione dei parametri di installazione dell'OS
  - per ogni software aggiuntivo, la versione scelta e la sua configurazione
  - configurazioni per servizi infrastrutturali – come i file di configurazione del firewall o di un application server
  - ogni script usato per gestire l'ambiente
  - tutte queste informazioni
    - vanno specificate in modo completo e dettagliato (*infrastructure-as-code*)
    - vanno gestite in un sistema di controllo delle versioni

* Gestione automatizzata di server e ambienti fisici

- Il provisioning e la configurazione di ambienti fisici e virtuali presentano molti aspetti in comune – ma anche alcuni aspetti distintivi
  - non consideriamo la gestione manuale
  - iniziamo dalla gestione automatizzata di server e ambienti fisici
  - il caso di server e ambienti virtuali è discusso più avanti

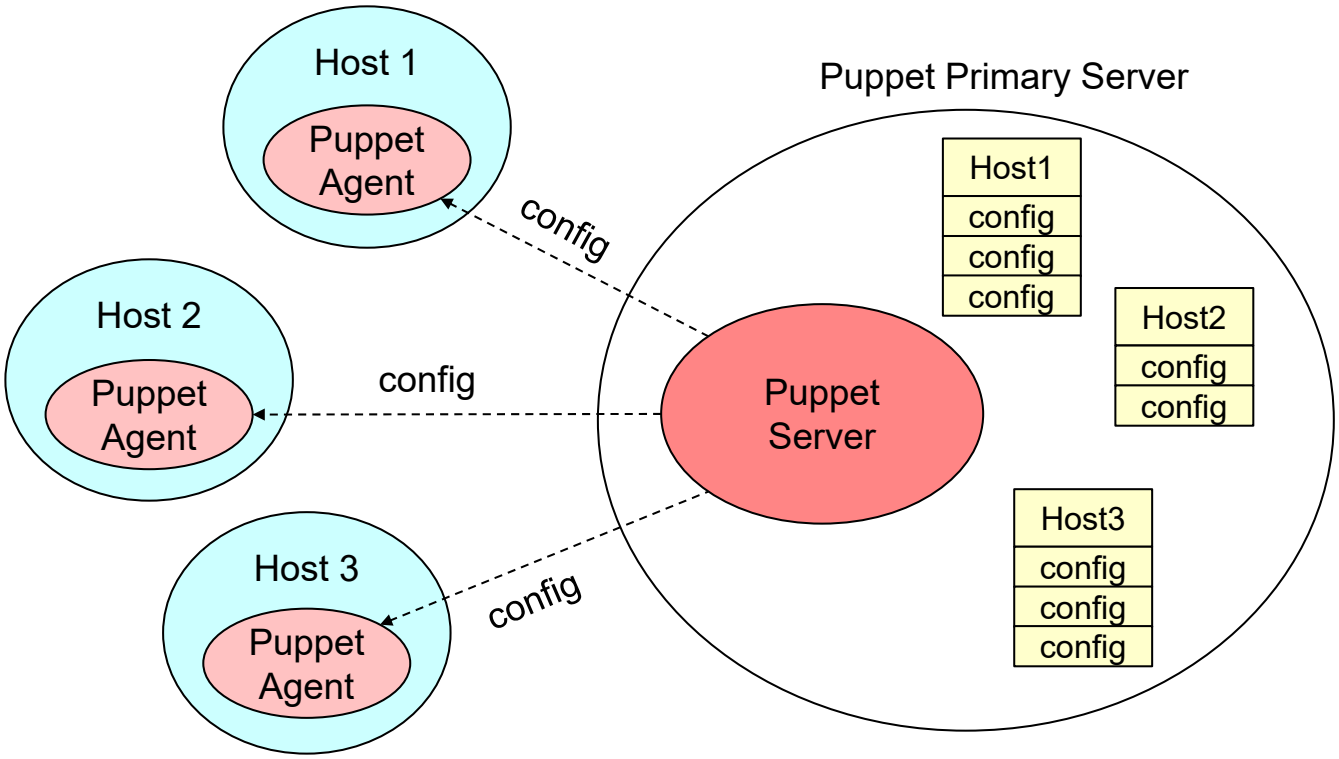
- Provisioning iniziale

- Il provisioning di un nodo (server) inizia con l'acquisizione e l'installazione fisica dell'hardware (che non può essere automatizzata) – ma come configurare il software del nodo?
  - un modo comune è iniziare con un boot remoto
    - ad es., con PXE (Preboot eXecution Environment) o Windows Deployment Services
    - il boot ha inizio da un'immagine di installazione di un OS, scaricata da un repository di immagini e poi avviata
    - le immagini nel repository possono essere predefinite – oppure potrebbero essere state personalizzate
  - va quindi completata l'installazione e la configurazione dell'OS
    - un modo comune è usare un processo di installazione "unattended" ("senza sorveglianza")
    - ad es., Kickstart (RedHat), Preseed (Debian) e Unattended Windows Setup (Windows)

- Gestione delle configurazioni

- Il passo successivo riguarda l'installazione e la configurazione del middleware e dello stack software necessario in ogni nodo
  - si può automatizzare usando uno strumento software per la *gestione delle configurazioni* (*configuration management software*) – ad es., Puppet, Chef o Ansible
    - l'amministratore specifica lo stato desiderato di un ambiente mediante un linguaggio di configurazione – per configurare ambienti, nodi (server) e altre risorse (ad es., package, middleware e servizi)
    - lo strumento garantisce che l'ambiente e i suoi nodi siano nello stato specificato
  - un'architettura comune è quella agent-server
    - il nodo server (centralizzato) gestisce le configurazioni di una flotta di nodi gestiti – in ogni nodo gestito, la configurazione dell'ambiente viene applicata da un agente (locale)

Gestione delle configurazioni (Puppet)



Gestione delle configurazioni

- Le configurazioni (di ambienti, nodi e risorse) sono specificate mediante dei file di testo (*infrastructure-as-code*)
  - viene usato un Domain-Specific Language (DSL) specifico per le informazioni di configurazione – la sintassi è spesso basata su linguaggi come YAML, JSON o Ruby
  - la configurazione di un ambiente comprende
    - la descrizione dei suoi nodi
    - per ciascun nodo, la specifica dei package e servizi da installare in quel nodo (con i relativi parametri di configurazione) e la definizione delle variabili d'ambiente
    - altri file e template – ad es., file di configurazione personalizzati e script da eseguire nei nodi
  - i linguaggi di configurazione sono di solito dichiarativi (e non imperativi) – ovvero, la configurazione desiderata viene specificata indipendentemente dal modo in cui verrà ottenuta

Gestione delle configurazioni

- I sistemi per la gestione delle configurazioni sostengono l'autonomicità (la capacità di autoripararsi) dei nodi e degli ambienti
  - le configurazioni vengano applicate in modo idempotente
  - se nel repository viene modificata una configurazione, allora il sistema provvede ad aggiornare tutti i nodi che adottano quella configurazione

- Gestione e configurazione del middleware

- Ogni servizio di middleware (ad es., un server web o un message broker) è di solito composto da file binari, configurazioni e dati
  - nei casi più semplici, l'installazione di un servizio di middleware può essere gestita con uno strumento di gestione delle configurazioni
  - ci sono però anche casi meno semplici da automatizzare
    - ad es., se il middleware non è disponibile nel formato del package manager in uso nell'OS di destinazione – oppure se non è stato pensato per un'installazione silenziosa
    - una possibile soluzione è predisporre un proprio package per il servizio di middleware di interesse

Configurazione del middleware

- Ogni servizio di middleware richiede delle configurazioni (con diverse modalità) – la loro gestione automatizzata è più o meno semplice
  - file di configurazione (spesso in XML)
    - è possibile usare dei template personalizzati
  - configurazione mediante CLI (comandi testuali), operazioni REST o API
    - è possibile usare degli script
  - configurazione binaria, gestita mediante una console grafica
    - è spesso problematica – ma talvolta sono possibili soluzioni ad-hoc
    - in alcuni casi si preferisce usare un servizio di middleware differente

- Gestione di servizi infrastrutturali

- Anche i servizi infrastrutturali – come reti, router, DNS e servizi di directory – vanno configurati opportunamente
  - le configurazioni di rete possono essere complesse – ad es., più reti isolate per diversi tipi di traffico (ad es., public e admin)
  - i problemi relativi a queste configurazioni sono di solito difficili da diagnosticare – ad es., se un'applicazione funziona nell'ambiente di test ma non in quello di produzione
  - alcuni suggerimenti
    - fare il versionamento di tutte le configurazioni dei servizi di rete o infrastrutturali
    - usare un buon sistema di monitoraggio della rete
    - usare il logging per sostenere la diagnosi
    - usare nell'ambiente di test una topologia della rete quanto più possibile simile a quella dell'ambiente di produzione

- Deployment di applicazioni e servizi

- L'installazione e la configurazione di applicazioni, componenti e servizi applicativi può essere effettuata in modo automatizzato con le tecniche viste in precedenza
  - i componenti o servizi vengono compilati e assemblati in unità di rilascio – ad es., file jar oppure war – che comprendono anche i file di configurazione
  - questa unità possono essere spesso installate mediante CLI, operazioni REST o API – oppure mediante una semplice copia
  - il deployment mediante console grafica (o console web) non è sempre automatizzabile

- Creazione di immagini personalizzate

- Dopo aver installato e configurato un server, con una configurazione specifica
  - è possibile creare un'immagine personalizzata e salvarla nel repository delle immagini
  - questo può rendere più semplice e più veloce
    - il provisioning di un nuovo server con quella configurazione
    - il ripristino di quel server

* Gestione automatizzata di server e ambienti virtuali

- Il provisioning automatizzato di server (nodi) e ambienti virtuali (oppure nel cloud) presenta molti aspetti in comune con il caso fisico – ma anche alcuni aspetti distintivi

- Provisioning iniziale

- Il provisioning di un nodo (server) ha inizio dalla configurazione e creazione di una VM
  - è facile configurare in modo automatizzato l'hardware del nodo virtuale – ma anche dello storage e della rete
  - la VM viene poi avviata da un'immagine di VM predefinita – da un repository di immagini, pubblico o privato
    - è anche possibile l'avvio da un'immagine di installazione di un OS – ad es., per iniziare la creazione di una nuova immagine di VM
  - questa attività può richiedere pochi secondi

- Un repository di immagini di VM

- Le immagini di VM (o template o baseline o box) vengono spesso gestite in un repository di immagini (pubblico o privato)
  - ciascuna immagine contiene un OS preinstallato e preconfigurato – ma può anche contenere del software aggiuntivo
  - le immagini sono spesso personalizzate
    - è facile creare immagini di VM
  - il livello di personalizzazione di un'immagine è anche chiamato *baking* ("cottura")
    - un'immagine può essere molto generale (*light baking*) – o anche ottimizzata per un caso specifico (*heavy baking*)

- Gestione delle configurazioni

- L'installazione può proseguire con l'utilizzo di un sistema per la gestione delle configurazioni – per installare e configurare nel nodo tutto il software desiderato
  - questo può richiedere da pochi secondi a diversi minuti
- Il risultato dell'intera installazione e configurazione di una VM può essere poi salvato come nuova immagine di VM e aggiunto al repository di immagini
  - per ridurre ulteriormente il tempo di configurazione di nuove VM

- Gestione di ambienti virtuali

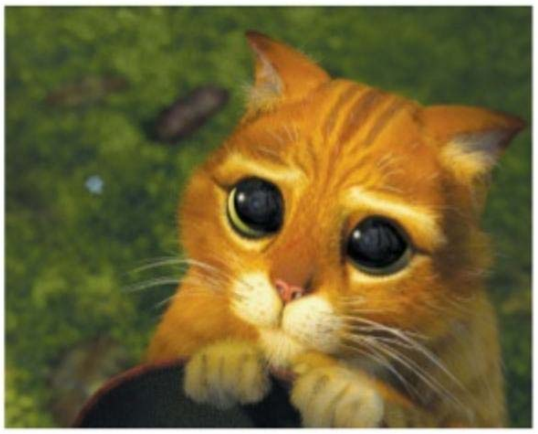
- Ci sono anche strumenti per la configurazione e la gestione automatizzata di interi ambienti virtuali o nel cloud – ad es., VMware vRealize, Terraform, Vagrant e AWS CloudFormation
  - un approccio infrastructure-as-code
  - la configurazione di un ambiente comprende
    - descrizione dei nodi – per ciascun nodo
      - configurazione dell'hardware virtuale e dell'immagine di VM da utilizzare
      - configurazione software del nodo – anche con riferimento a strumenti per la gestione delle configurazioni
    - descrizione e la configurazione dell'infrastruttura
      - ad es., reti virtuali e storage
  - le configurazioni possono essere parametriche
    - per creare ambienti multipli, strutturalmente simili o identici tra loro

Gestione di ambienti virtuali multipli

- Se ci sono sufficienti risorse di calcolo (ad es., nel cloud) è possibile creare ambienti multipli – ciascuno separatamente dagli altri
  - alcuni scenari di utilizzo
    - per creare uno o più ambienti di test – separati dall'ambiente di produzione
    - per effettuare l'aggiornamento di un sistema software in esecuzione nell'ambiente di produzione

* Pets vs Cattle

- I modelli “pets” (“animali da compagnia”) e “cattle” (“bestiame”) – e i meme “pets vs cattle” e “cattle not pets” – rappresentano una dicotomia nella gestione degli ambienti di esecuzione
  - “pets” rappresenta la gestione manuale e antiquata
  - “cattle” rappresenta la gestione (completamente) automatizzata e moderna



Pets vs Cattle

- Modello “pets”
  - i nodi dell’ambiente sono unici e indispensabili
  - hanno nomi carini (ad es., thelma e louise)
  - il software viene installato e aggiornato manualmente
  - non dovrebbero mai andare giù e non possono essere sostituiti – in caso di problemi, si fa di tutto per “ripararli” manualmente
- Modello “cattle”
  - i nodi dell’ambiente sono progettati per il fallimento
  - hanno nomi mnemonici (ad es., web-01 e web-02)
  - il software viene installato e aggiornato in modo automatizzato
  - in caso di problemi, vengono distrutti e sostituiti da altri nodi identici – non si cerca mai di “riparare” i nodi
  - è un modello moderno – la cui adozione richiede un cambiamento non solo tecnologico, ma anche culturale

* Discussione

- La gestione automatizzata degli ambienti è possibile e importante sia nel caso di server e ambienti fisici che, ancora di più, nel caso di server e ambienti virtuali o nel cloud
  - la virtualizzazione amplifica i benefici della gestione automatizzata di server e ambienti
    - supporta un processo di provisioning e rilascio controllato, completamente automatizzato e ripetibile – con una riduzione dei tempi, dei costi e dei rischi
    - il provisioning è favorito dalla possibilità di creare e gestire facilmente un repository di immagini di VM personalizzate
    - l'acquisizione di un ambiente virtuale o nel cloud richiede in genere pochi minuti e un costo iniziale nullo
    - è semplice gestire ambienti multipli e separati simili o identici
    - abilita la possibilità di effettuare il rilascio di un sistema software in un ambiente con un singolo click