

Prevedere il comportamento delle applicazioni Web in produzione

Serafina Rocca
SPE Engineer
s.rocca@k-tech.it

Simone Federici
APM Specialist
s.federici@k-tech.it



Il ciclo di seminari K-Tech a Roma3 su APM

Il 12 marzo 2009

• **APM: WWWWW**

(What, Why, Where, Who, When)

- Una notte in ufficio: ore 08:00 a.m. risolto!
- APM: Metodologia e strumenti
- Performance (anti)patterns in enterprise architectures
- Troubleshooting methodologies in distributed systems
- Agile APM, an heretic's approach to SPE





Presentare il metodo di certificazione, basato sulle discipline SPE/APM e ideato da K-Tech, per prevedere e controllare le performance dei sistemi in produzione.

Il metodo può essere applicato sia con l'ausilio di software di monitoraggio e di stress test commerciali, sia con software open source.

Il target:

- I Manager delle Operation
- Gli Architetti
- I Responsabili delle linee di Business (LOB Owner)
- Aspiranti Software Performance Engineers



Le motivazioni:

- Condividere la nostra esperienza in un contesto di professionisti e accademici.

Che cosa si intende per performance?

Le possiamo misurare?

Una Ferrari che si spegne a ogni semaforo, ha buone performance?

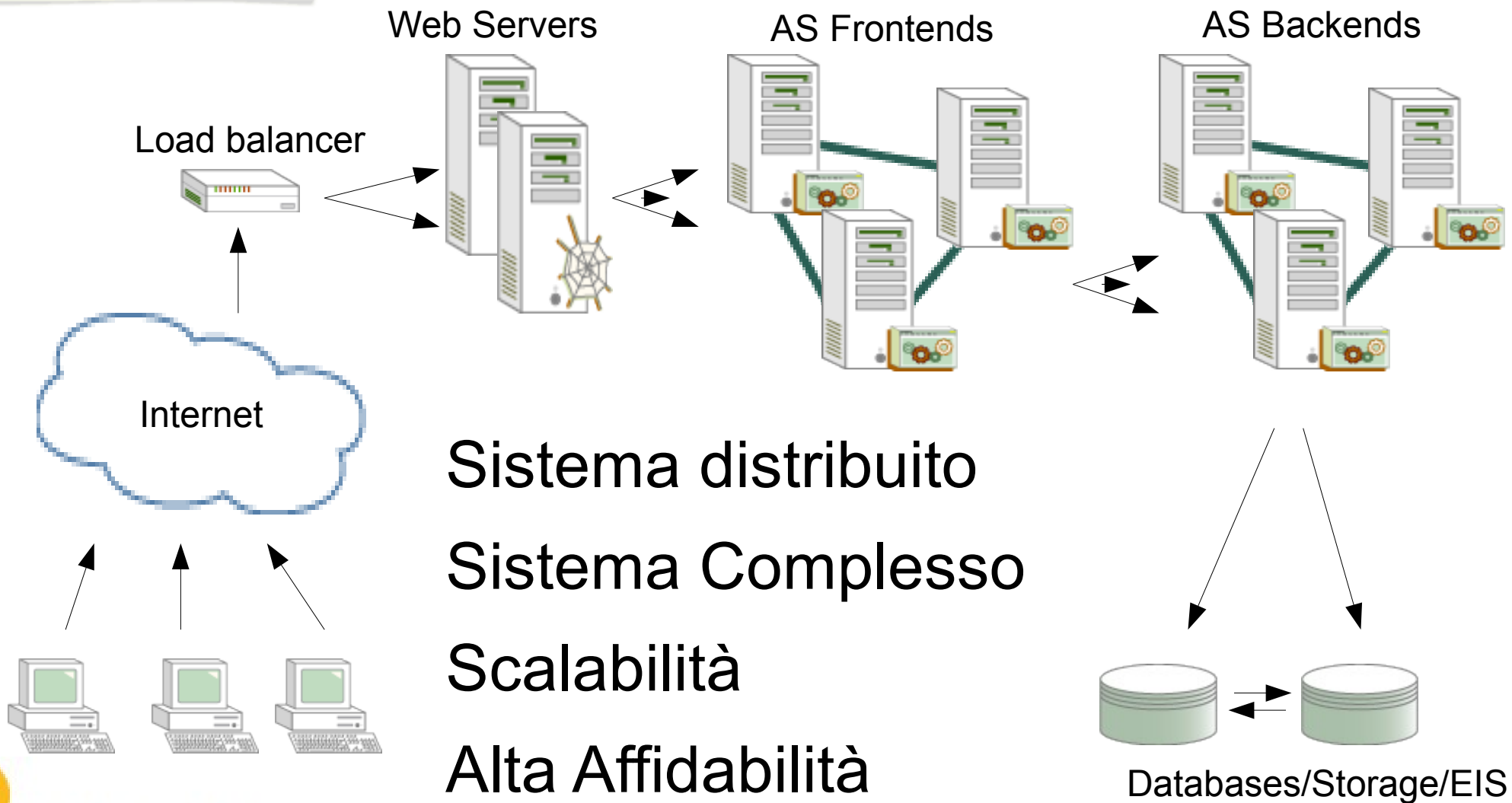
E se l'aspettativa del cliente è il basso consumo di carburante?





- **Analista:** studia i pattern di utilizzo del sistema utilizzando strumenti statistici, file di log, etc.
- **SPE Engineer:** ha la conoscenza di cosa fare durante la certificazione
- **APM Specialist:** conosce il metodo e gli strumenti per il monitoraggio
- **DBA, System Administrator, Architect**
Ogni figura ha il proprio set di strumenti

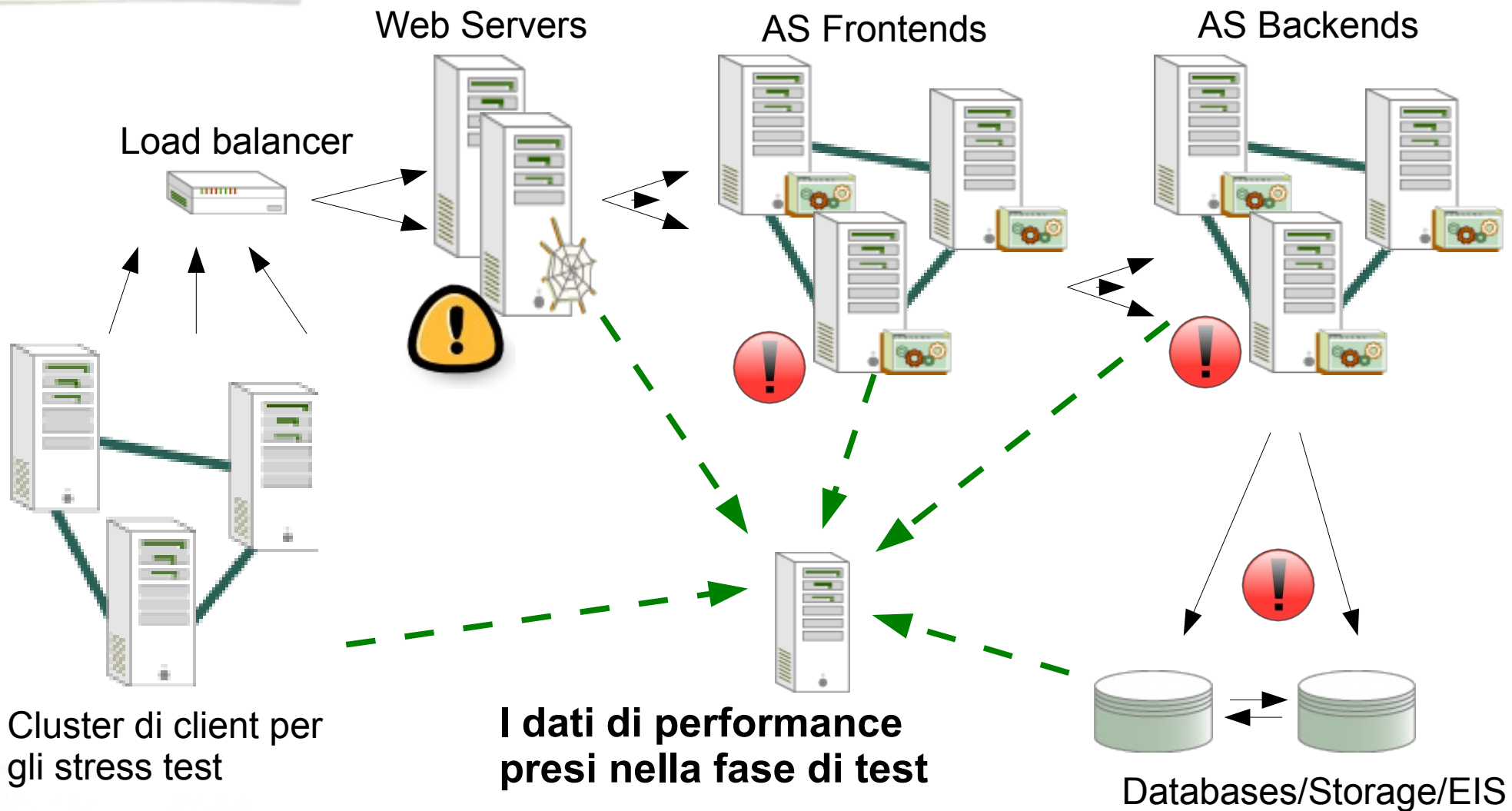
Architettura ambiente di produzione



Monitoraggio Produzione: approccio 'System Thinking'



Monitoraggio: Ambiente di Test



Necessità del cliente

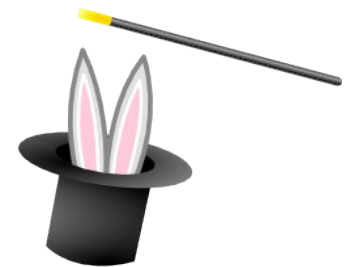
- Servizi online con **alta visibilità** (siti istituzionali..)
- Servizi online **critici per il business** (e-banking, e-commerce..)
- Servizi interni (email, LDAP, intranet, ...)
- Dimensionamento ottimale delle architetture
- **Rispetto SLA / Requisiti non funzionali**
 - tempi di risposta medi (On line)
 - 'finestre' temporali (Batch)
 - 99.9999% di affidabilità



Cos'è il Software Performance Engineering?

Già dalla fase di disegno del software, si dà una forte importanza alla progettazione, implementazione dei requisiti non funzionali, o meglio alle performance.

Ci dà le basi teoriche e gli strumenti per identificare i classici problemi architetturali.



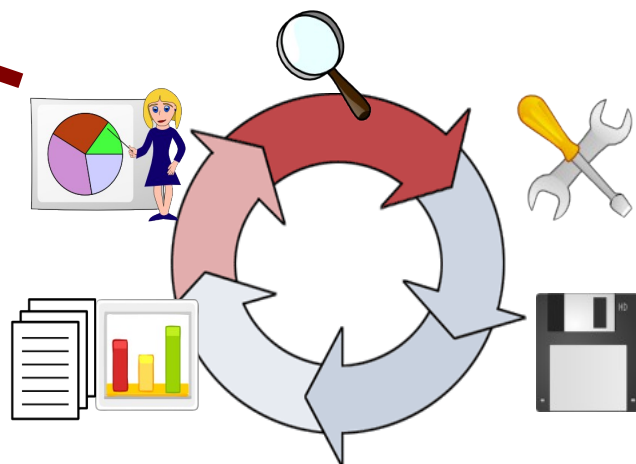
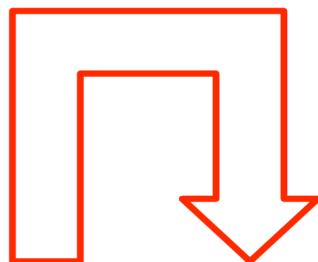
Processo APM



Service Desk



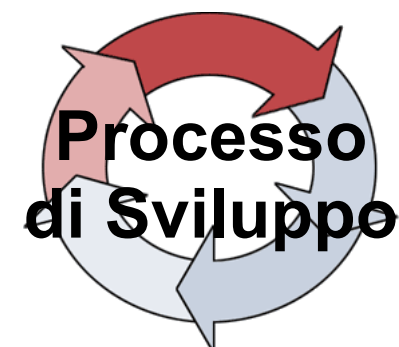
Manager
k-tech
creating excellence



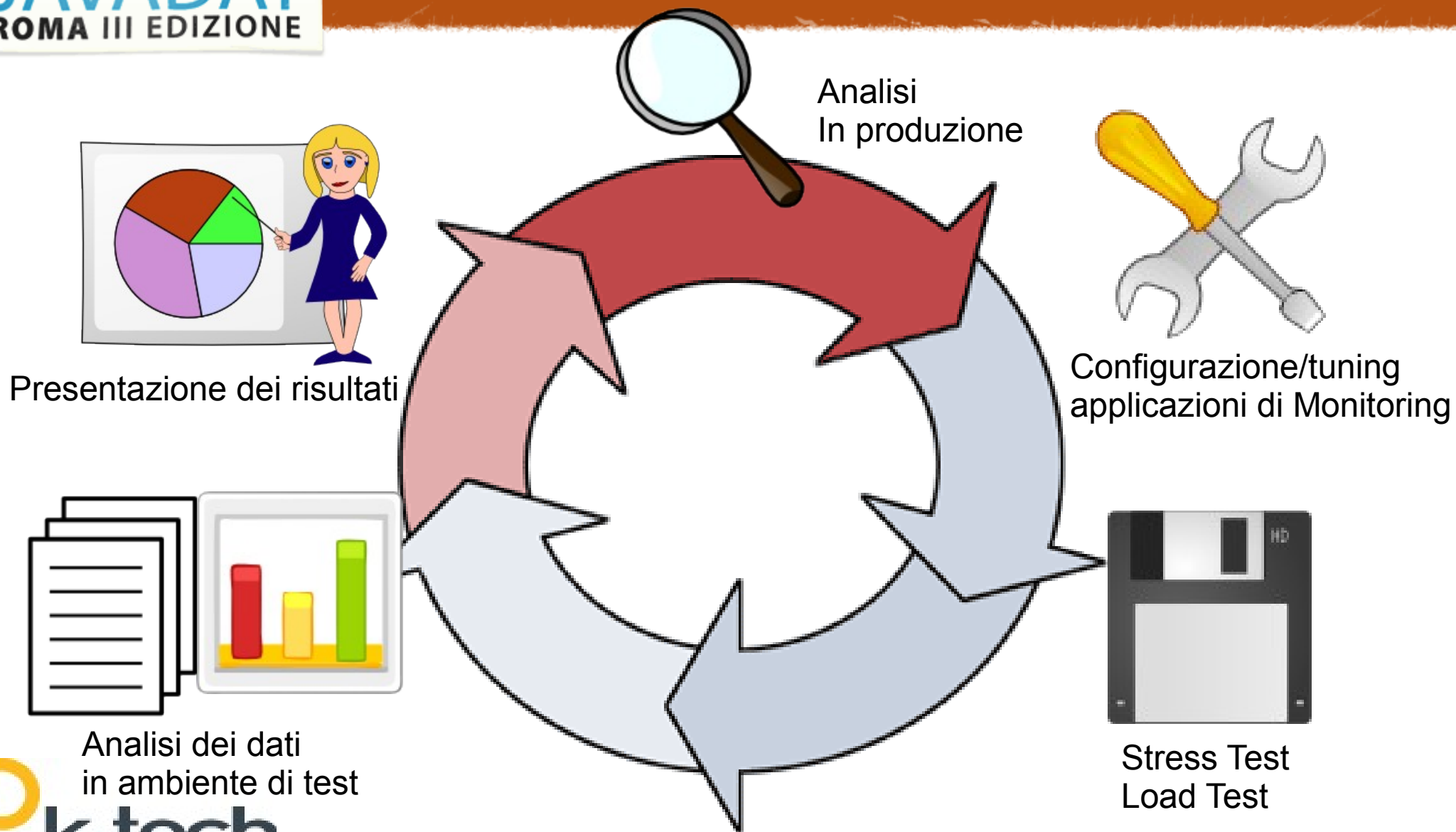
Sysadmin



DBA

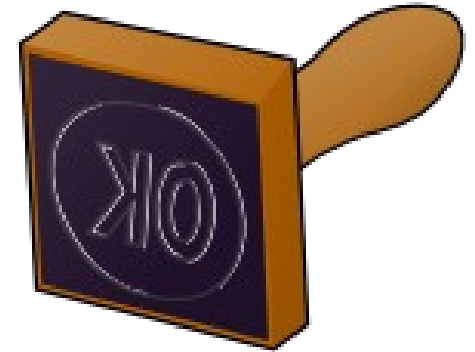


Processo APM



Processo di Certificazione

Cliente
+ Applicazione/i
+ SPE
+ APM
+ **Esperienza**
+ **Metodo**
=



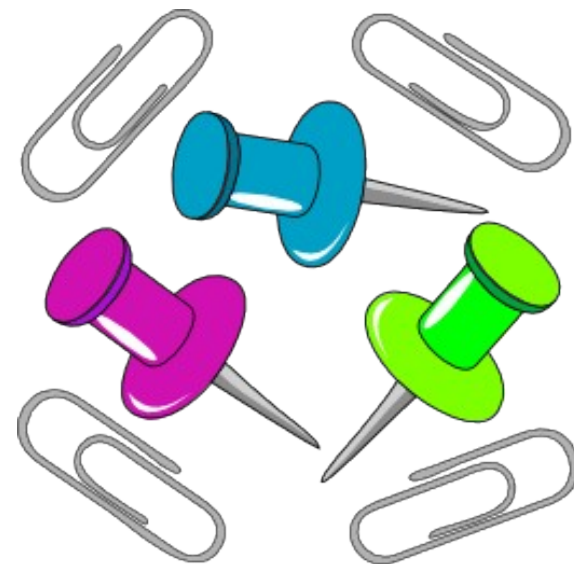
Processo di certificazione

Garantire la qualità di una applicazione prevedendo come si comporterà sotto un carico stimato.

Espressi in funzione delle aspettative

Esempio:

- Utenti connessi
- Processi eseguiti
- Tempi di risposta
- ...



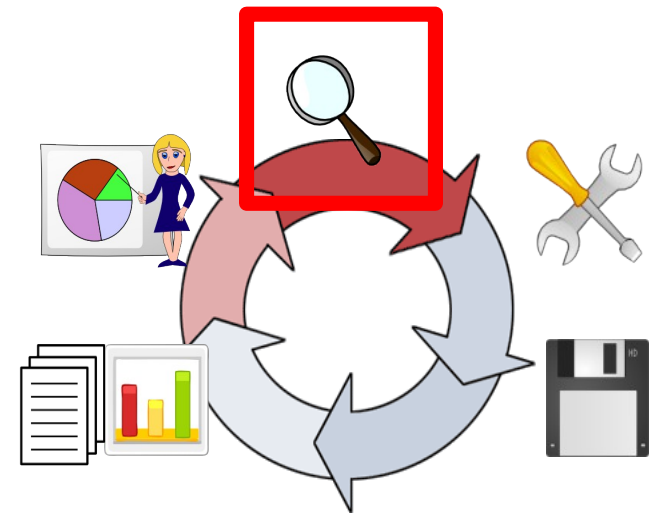
Ossia tutte le possibili metriche **osservabili**
dall'utilizzatore del servizio/software

Cosa?

- Dati statistici di accesso all'applicazione
- Traffico di rete
- Dati provenienti dal monitoraggio dei componenti applicativi

Perché?

- La distribuzione temporale del carico sull'applicazione
- Funzionalità più utilizzate
- Eventi critici



Tempi di risposta

Invocazioni concorrenti

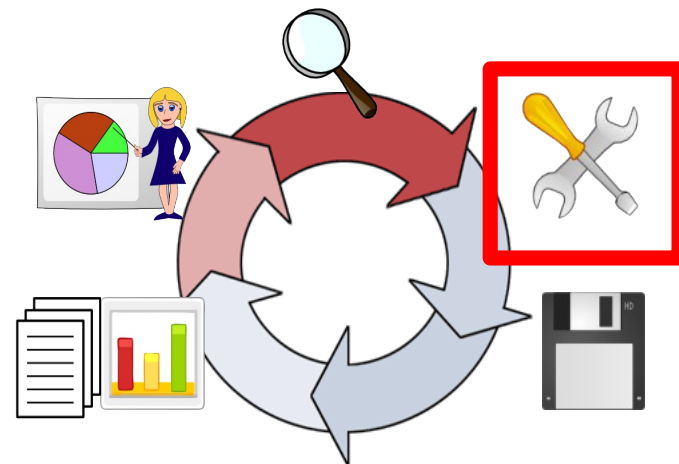
Invocazioni per secondo

Invocazioni andate a buon fine

Numero di invocazioni nella stessa transazione

Risorse in uso

Latenza di rete



Progettazione

Identificazione degli use case critici

Creazione dei test case, schedulazioni e script di automazione

Esecuzione

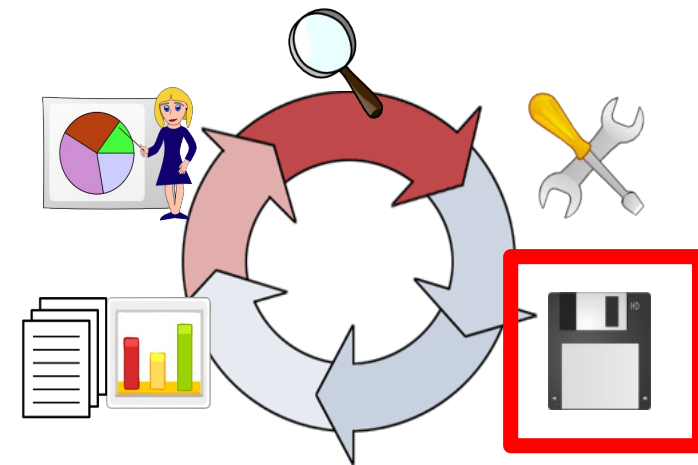
Esecuzione dei test con carichi diversi

Monitoraggio e registrazione dei risultati

Validazione

Stima dell'errore

Ripetibilità dei test



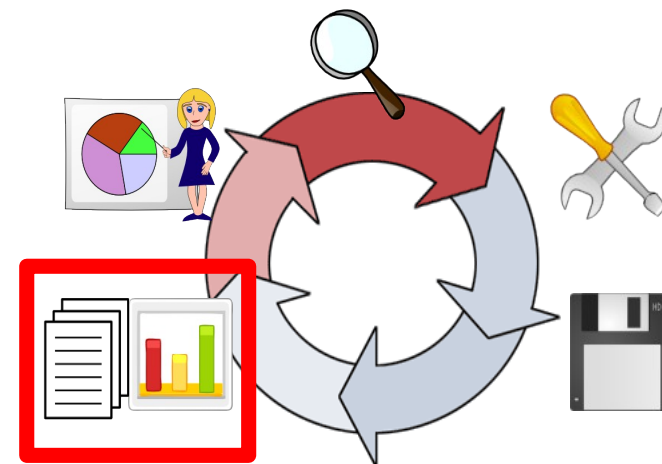
I principali indicatori delle performance sono da individuare nella fase di stress test che emula una situazione di traffico potenzialmente pericolosa per il servizio:

Throughput dei componenti

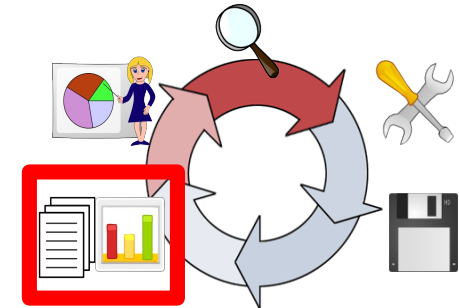
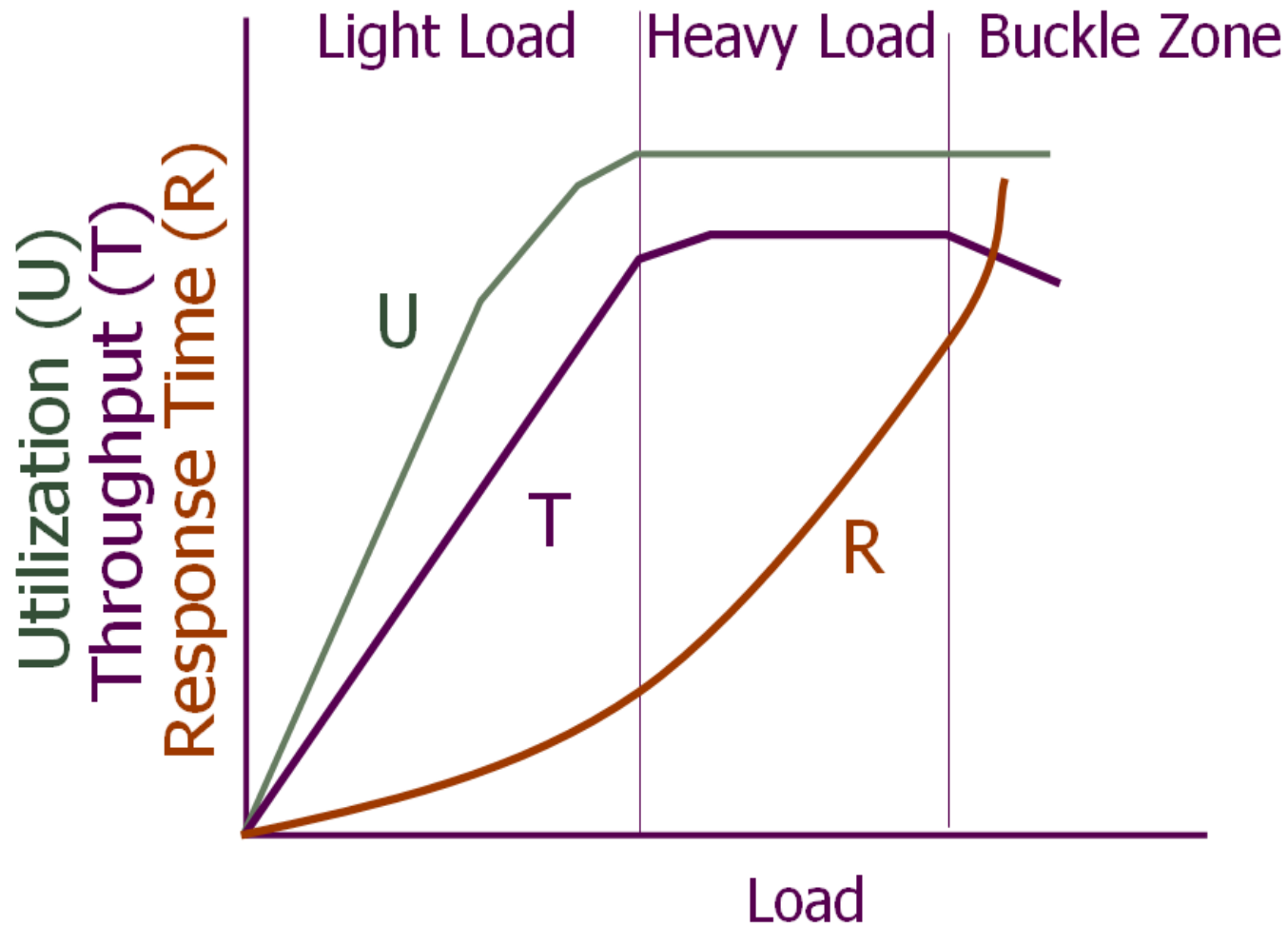
(Servlet, Web Services, ...)

Response Time (EJB, SQL, ...)

Risorse Usate (CPU, Memoria, Banda ...)

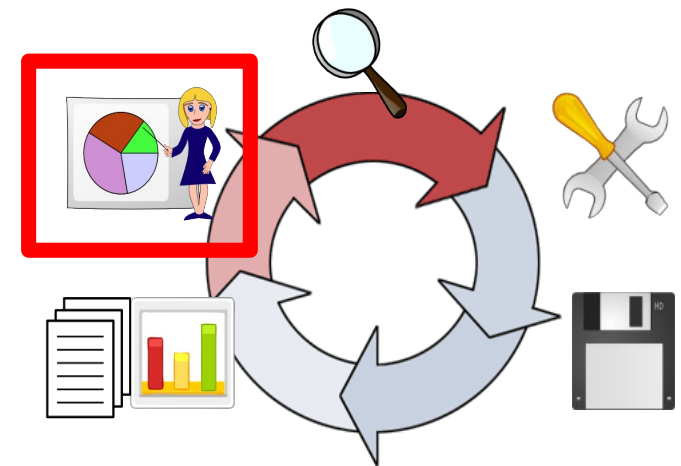


Output



Gli esiti della certificazione devono essere compatibili con le osservazioni registrate in produzione

Devono essere presentati in un documento che certifica i limiti del sistema, documenta le azioni fatte e giustifica le conclusioni.



Batch

Identificazione dei colli di bottiglia

Determinazione del livello di parallelismo minimo

Online

Determinazione delle risorse minime necessarie
(dimensione del cluster, numero di connessione al DB)

Identificazione dei pattern problematici

Impatto disattenzione Best Practices





Scenario: Contratti da fatturare aumentano di un ordine di grandezza. Hardware maggiorato.

Come assicurare che le performance del batch permettano di rispettare la finestra di tempo a disposizione?

Stabiliti gli obiettivi di performance KPI: fatture/ora

Punto di partenza: 16 giorni

Monitorati e individuati i colli di bottiglia:

Opencursor su Mainframe, Attività Query (commits, tempi, invocazioni), Eccessivo Log, ecc..

Aumentato il parallelismo (sino al minimo necessario).

Tempi: 2 settimane (elapsed)

Punto di arrivo: 5 ore

Successo: Previsti con precisione i tempi del batch in produzione.



Scenario: Picchi di carico massimo ad opera degli utenti. Sistema instabile, crash sistematici all'aumentare delle richieste. (vedi pagina seguente)

Hardware potenziato.

Punto di partenza: 10.218 unique browser/h, 124.000 page views/h

Obiettivo: Architettura che permettesse un throughput doppio rispetto al limite attuale.

Individuata una migliore configurazione FE – BE per non dare disservizio.

Cambiamento architetturale database e Application Servers.

Aumentata la cache sul FE (cosa registrare e per quanto tempo)

Indicazione sulle modifiche da effettuare sul BE (quali moduli sono problematici)

Tempi: 3 settimane (elapsed)

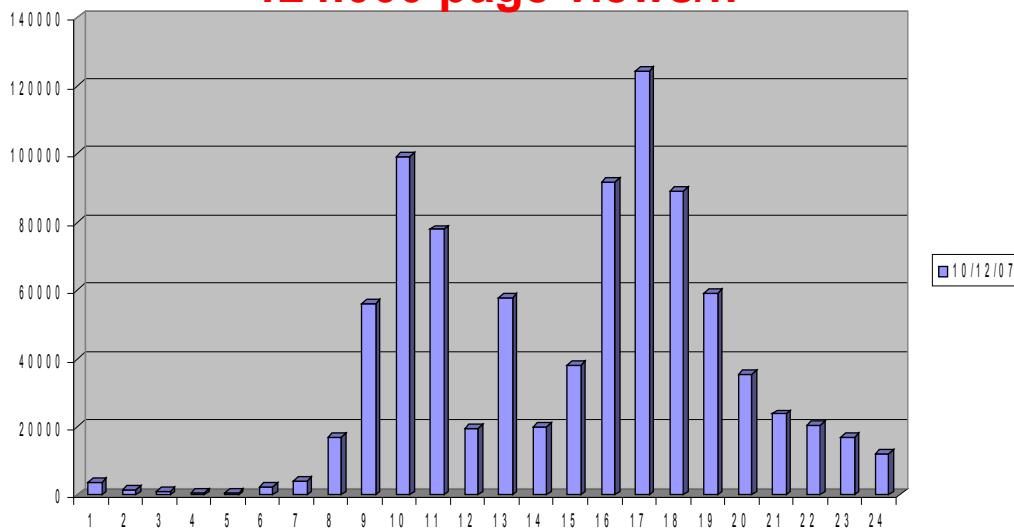
Punto di arrivo: 7 gennaio, 18.000 unique browser/h e 207.000 page view/h

Successo: registrati 273 Gb di banda giornalieri (record contro i 109 precedenti)

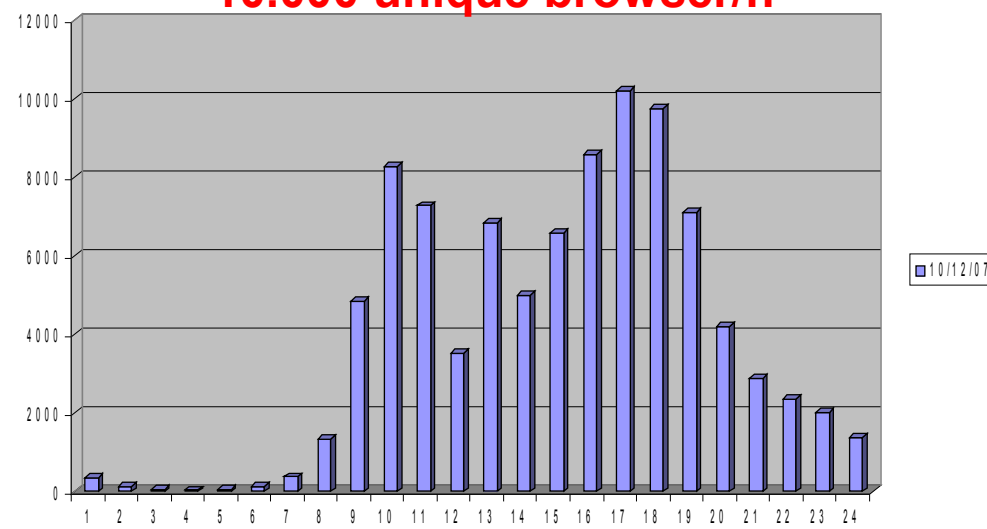


Throughput: Situazione Iniziale

Pagine Visualizzate
124.000 page views/h

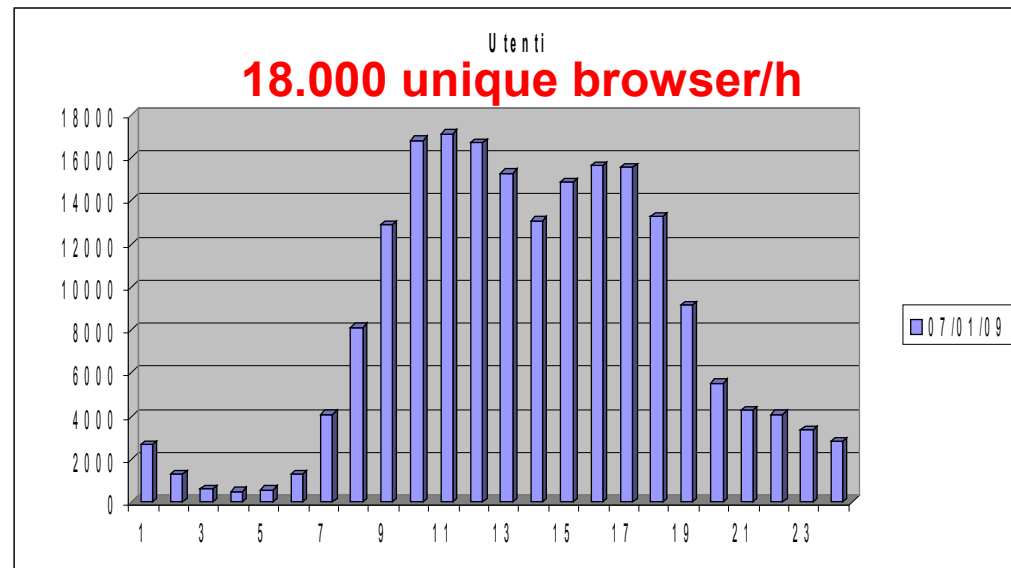
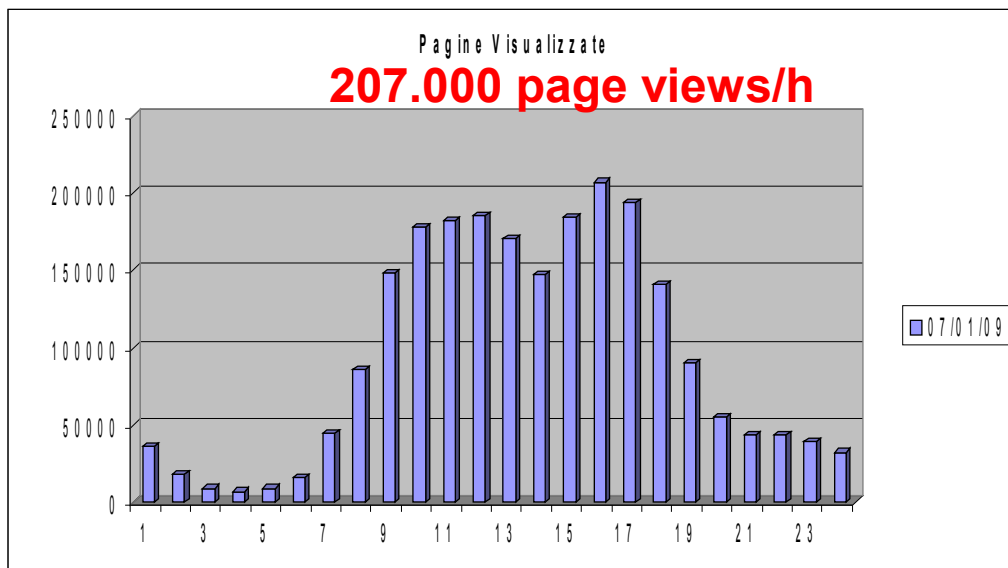


Utenti
10.000 unique browser/h





Throughput: Situazione Attuale (working in progress)



Monitoraggio:

- Introscope Wily CA
- IBM Tivoli
- Mercury Interactive
- Quest
- Veritas
- JXInsight
- Awstats
- Nagios

Test:

- Jmeter
- Grinder
- Load Runner

Un ringraziamento speciale a Giuseppe Galli, principale esperto di performance in tutta EMEA

<http://www.k-tech.it>

<http://www.javaportal.it>

<http://www.perfeng.com/>

<http://www.systems-thinking.org/>

Connie U. Smith & Lloyd G. Williams (2005), ***Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*** (Addison-Wesley Object Technology Series)

Connie U. Smith, (1990) ***Performance Engineering of Software Systems, 1st Edition***

