



Seminario sullo sviluppo software in Consoft sistemi

3 giugno 2021

*Speakers: Alessandro Borello, Francesco Mocci
Interventi: Giacomo Russo, Homero Escobar Torres*



**“Ogni giorno
qualcosa di nuovo.**

alessandro.borello@consoft.it



Introduzione

Chi siamo e cosa facciamo in Consoft



Consoft Sistemi - Mission, Vision & Value



Il nostro claim aziendale recita "**Ogni giorno qualcosa di nuovo**" e per supportarlo abbiamo ben chiaro, sin dalla nostra fondazione avvenuta nel 1986, la Mission e la Vision aziendale, così come siamo fieri dei nostri Valori, che fanno leva sulle persone e sulle loro idee.



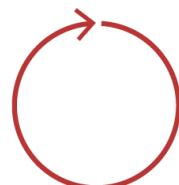
Talento

Crediamo nei **giovani**, nella freschezza delle loro idee e nei nuovi **progetti imprenditoriali**.



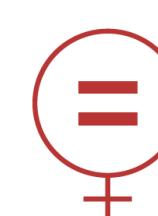
Formazione

Crediamo nell'**aggiornamento** continuo, nella **specializzazione** delle competenze e nella coltivazione della **professionalità**.



Cambiamento

Crediamo nella necessità di un'**evoluzione costante**.



Pari opportunità

Crediamo nel valore strategico della **diversità** che è sempre motivo di arricchimento.



BU Application - Efesto – I nostri Team di sviluppo applicativo



Exremely
Fast
Enterprise
Software
Taskforce
On-demand

Da sempre in **Consoft** abbiamo messo al centro le **persone** considerandole il nostro valore più prezioso.

Grazie a queste persone, alle loro competenze e al loro spirito di gruppo, siamo riusciti in questi anni ad aiutare numerosi Clienti a realizzare i loro progetti con successo. Efesto nasce all'interno della **BU Application**, come naturale evoluzione di questa vocazione a fare “Team” .

Efesto è prima di tutto un gruppo di persone che è riuscito a mettere in comune le proprie competenze, con l'obiettivo di creare sinergie indispensabili per trasformare le idee dei nostri Clienti in software di qualità.

Grazie alle forti **competenze tecniche**, acquisite sul campo in questi anni, i **Team di Efesto** sono in grado di supportare i Clienti durante tutto il ciclo del software, dalla prototipazione al rilascio in produzione, utilizzando metodologie Agili e strumenti che permettono di ottenere il miglior risultato in termini di qualità ed efficienza.

I **Team di Efesto** sono composti da esperti di Software Design, da sviluppatori senior full-stack, da tester e da esperti di Code Review e Continuous Integration. I nostri team sono in grado di confezionare soluzioni software altamente personalizzate ed efficienti per qualsiasi esigenza utilizzando le tecnologie e i pattern di programmazione più moderni.



Sviluppo software

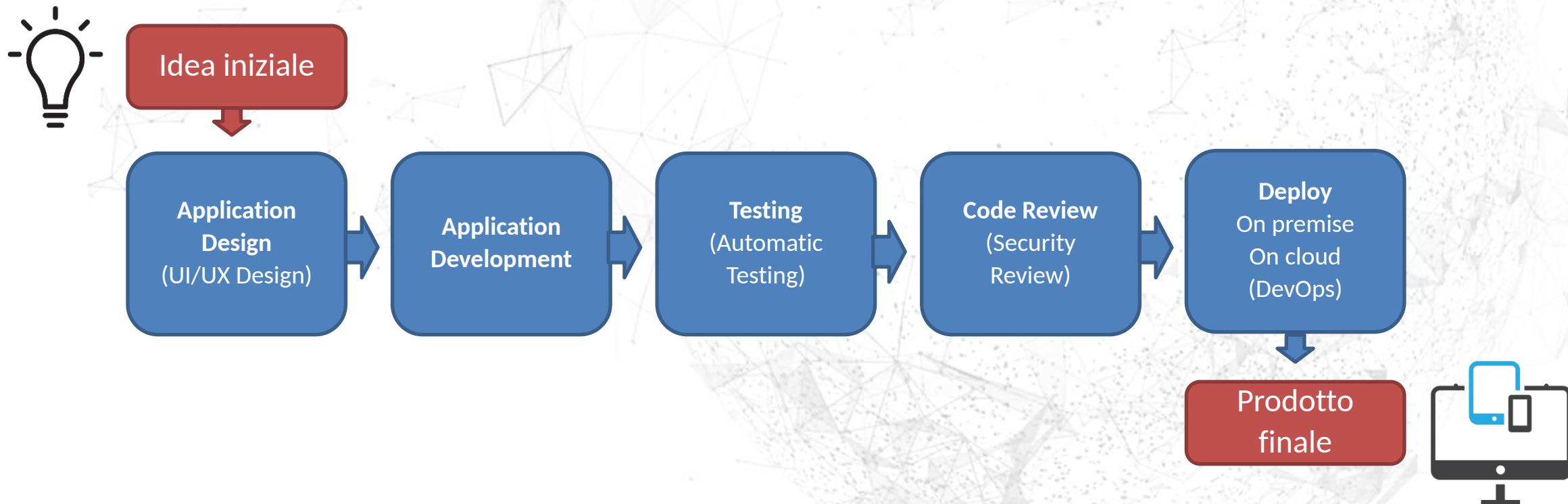
Come si passa dall'idea iniziale al prodotto finito



Il ciclo di vita del software



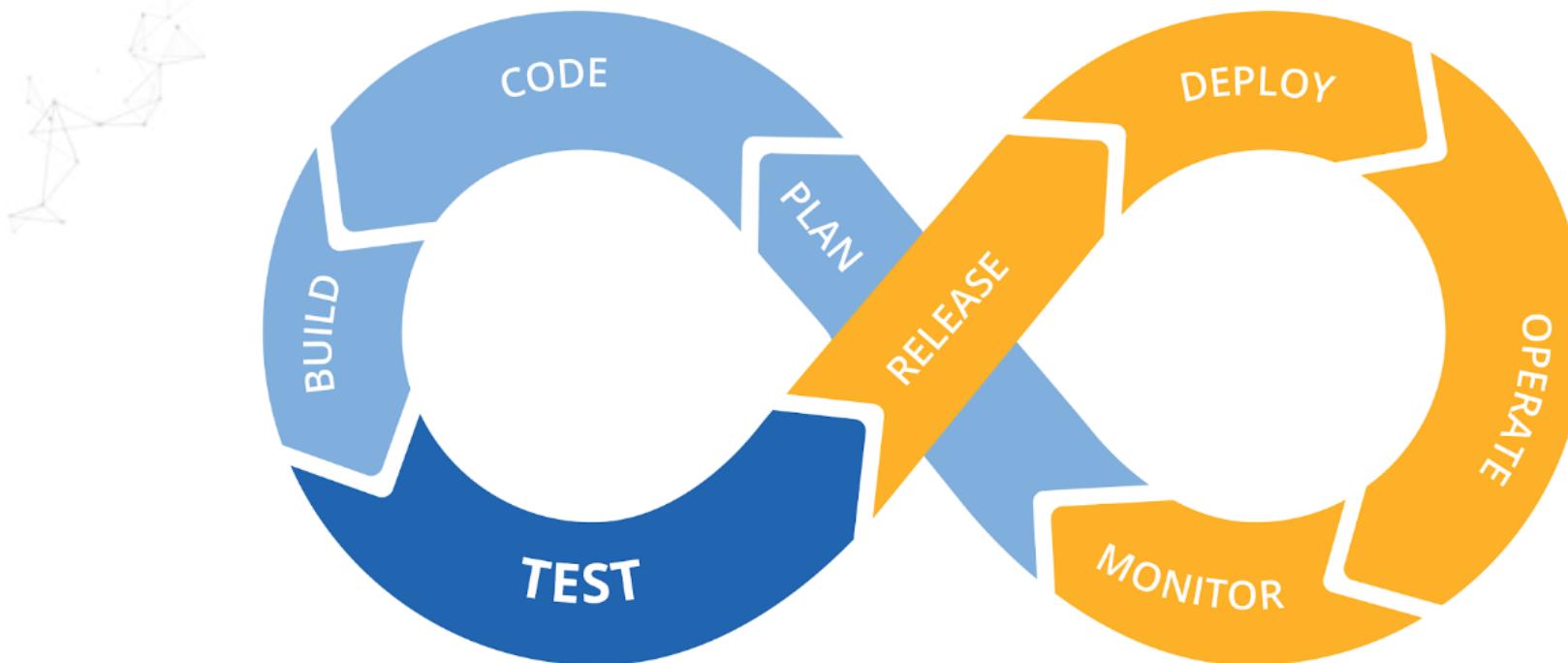
Ogni volta che un cliente ci chiede di realizzare un'applicazione web, i nostri Team si attivano per mettere in pratica una serie di step che ci permettono di realizzare un prodotto finale capace di soddisfare appieno le esigenze degli utenti che lo utilizzeranno.





Il ciclo di vita del software

Per ridurre le tempistiche di rilascio del prodotto finito, i nostri Team hanno iniziato ad applicare gli strumenti e le metodologie Agile che consentono di mettere subito a disposizione degli utenti le funzionalità man mano che vengono sviluppate. I rilasci frequenti di nuove funzionalità portano notevoli vantaggi ma rendono critiche le fasi di **Design, Coding e Testing** sulle quali è necessario focalizzare la nostra attenzione.





Design UI/UX

L'importanza della progettazione



User Experience: cosa è

User Experience: cosa è

Progettare un prodotto (sito web/app)



Facile
da usare



Rispecchia
aspettative
utente



Raggiunge
business goal

Come ottenere questo risultato

Discovery

Si parla con gli
utenti finali e
stakeholder





Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Analisi

Mappatura di come funziona il servizio e che tipi di utenti ci sono



Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Analisi

Mappatura di come funziona il servizio e che tipi di utenti ci sono

Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

An

Mappa funzionali che

Concept

Come sono strutturate le informazioni. Prima bozza



Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Analisi

Mappatura di come funziona il servizio e che tipi di utenti ci sono

Concept

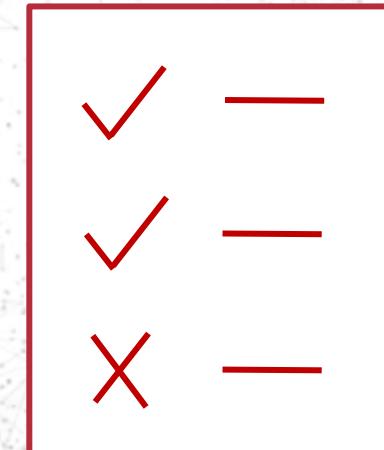
Come sono strutturate le informazioni. Prima bozza

Come ottenere questo risultato

Discovery
Si parla con gli utenti finali e stakeholder

Ana
Mappare le funzioni e che ti

Test di usabilità
Si testa la bozza sugli utenti, si trovano problemi



Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Analisi

Mappatura di come funziona il servizio e che tipi di utenti ci sono

Concept

Come sono strutturate le informazioni. Prima bozza

Test di usabilità

Si testa la bozza sugli utenti, si trovano problemi

Come ottenere questo risultato

Discovery

Si parla con gli utenti finali e stakeholder

Ana

Mappati funzior
e che ti

Grafica

Si «veste» la bozza
con colori, stile e
look

Test di usabilità

Testa la bozza
agli utenti, si
vano problemi

Come ottenere questo risultato





Sviluppo Applicativo

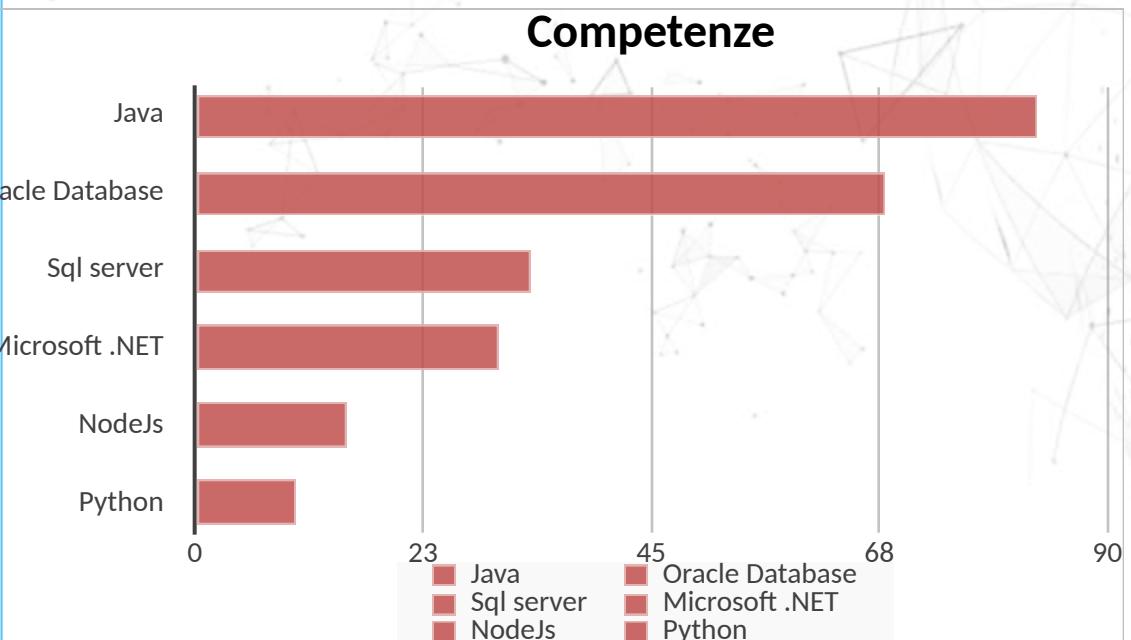
Gli strumenti e i framework che utilizziamo



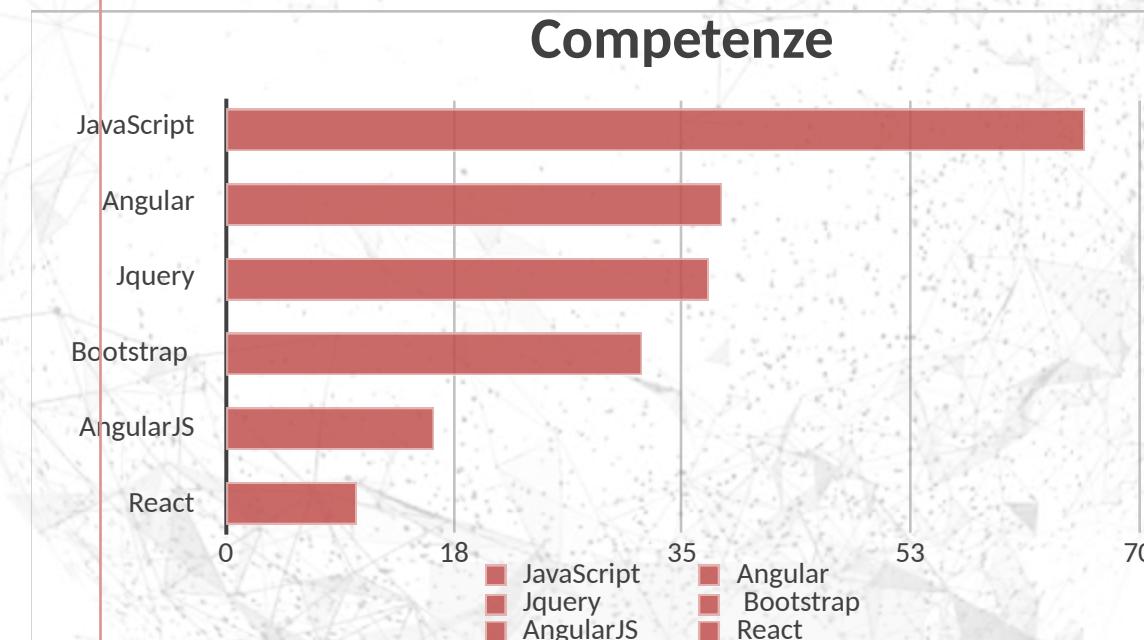
Composizione dei nostri Team, alcuni numeri



Back-end



Front-end





I framework front-end che utilizziamo



Angular è un framework sviluppato da Google e rilasciato per la prima volta nel 2010 (all'epoca si chiamava AngularJs). E' basato sull'utilizzo dei linguaggi TypeScript e JavaScript. Nel 2016 è stata rilasciata la versione 2 (chiamata Angular 2+ o semplicemente Angular) che ha rappresentato una evoluzione notevole rispetto alla precedente versione. Attualmente siamo alla versione 12 (19 Maggio 2021)

Vantaggi

- Basato su componenti (alto riutilizzo)
- Permette di implementare i servizi
- Template scritti in HTML

Quando utilizzarlo

Angular è un framework front-end indicato per utilizzi in ambienti enterprise basati su infrastrutture complesse (ad esempio micro-servizi).

Dove l'abbiamo utilizzato



Psd2



Gestione logistica



I framework front-end che utilizziamo



Angular è un framework sviluppato da Google e rilasciato per la prima volta nel 2010 (all'epoca si chiamava AngularJs). E' basato sull'utilizzo dei linguaggi TypeScript e JavaScript. Nel 2016 è stata rilasciata la versione 2 (chiamata Angular 2+ o semplicemente Angular) che ha rappresentato una evoluzione notevole rispetto alla precedente versione. Attualmente siamo alla versione 12 (19 Maggio 2021)

Vantaggi

- Basato su componenti (alto riutilizzo)
- Permette di implementare i servizi
- Template scritti in HTML

Quando utilizzarlo

Angular è un framework front-end indicato per utilizzi in ambienti enterprise basati su infrastrutture complesse (ad esempio micro-servizi).

Dove l'abbiamo utilizzato

Psd2



Gestione logistica



I framework front-end che utilizziamo



React è un insieme di librerie sviluppate da Facebook e inizialmente rilasciate nel 2013. Facebook utilizza ampiamente queste librerie nei suoi prodotti (Facebook, Instagram e Whatsapp). L'ultima versione stabile è la 17.0.2.

Vantaggi

- Basato su JavaScript
- Non necessita di una organizzazione a progetto
- Rappresenta una estensione del DOM Html

Dove l'abbiamo utilizzato



Progetto Canp la casa nel parco

Quando utilizzarlo

React diventa uno strumento molto potente se utilizzato in combinazione con altri tool come Redux o MobX. E' indicato nelle situazioni nelle quali c'è la necessità di utilizzare molti elementi grafici sulle pagine. Le app scritte con React sono facilmente espandibili e possono crescere nel tempo.



I framework front-end che utilizziamo



Vue.js è stato sviluppato da un ex-dipendente di Google, Evan You nel 2014. Negli ultimi anni Vue ha registrato un enorme interesse nelle community degli sviluppatori anche se non è supportato da una grande azienda come i suoi competitor. L'ultima versione disponibile è la 3.0.11

Vantaggi

- Combina HTML con direttive specializzate
- Permette di creare componenti snelli e riutilizzabili
- Versatile e altamente produttivo
- Ha una curva di apprendimento molto rapida

Quando utilizzarlo

Vue.js è stato sviluppato con l'idea di essere immediatamente produttivi. E' l'ideale per applicazioni e progetti che necessitano di un time-to-market molto ristretto.

Dove l'abbiamo utilizzato



Progetto Covid



Portale Open Data



Testing

Come garantire la qualità del codice che scriviamo



Perché testare è importante?



- Il costo di un bug, può essere misurato: dalla gravità del difetto e quando viene rilevato.
- In poche parole prima si trova il bug e minore è il costo.
- Ad esempio, se l'errore viene rilevato nelle specifiche dei requisiti, durante la raccolta e l'analisi dei requisiti, è piuttosto economico ripararlo.
- Mentre se vengono rilevati in fase molto avanzata, a seconda di quanto sia grave il bug, il costo è quasi triplicato.
- Il costo di una regressione in fase di manutenzione è ancora più alto.



Un esempio reale



Il giorno del «Prime» annuale di Amazon, quindi uno dei giorni più attivi dell'anno, i suoi servizi web sono crollati, causando una sbalorditiva perdita di 34 milioni di dollari l'ora.

Questo non è stato uno scherzo per Amazon, considerando l'entità delle sue perdite.



SORRY
something went wrong
on our end

Please go back and try again
or go to [Amazon's home page](#).



Jaja
Meet the dogs of Amazon



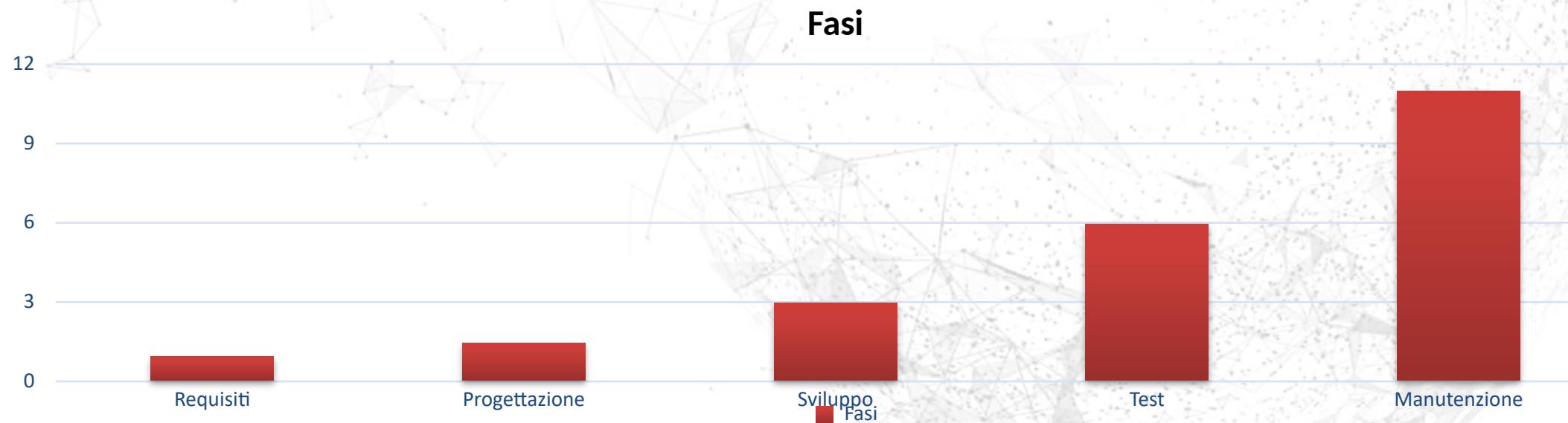
Costo rilevazione e fixing nel tempo



Non è una novità, che i difetti del software, siano molto più costosi, se trovati più tardi nel ciclo di vita dello sviluppo, il momento più critico è quando vengono rilevati nella produzione.

L'ovvia ragione, per l'aumento dei costi è lo sforzo complessivo richiesto dagli sviluppatori per identificare

la causa principale del difetto, risolverlo ed eseguire la quantità di test per garantire che il difetto sia corretto e assicurare che la correzione non introduca una nuova regressione .





Percentuale bug rilevati



Il rilevamento dei difetti, lo stesso giorno in cui sono stati introdotti nella build del software consente agli sviluppatori di dedicare il 47% della loro giornata lavorativa all'innovazione e allo sviluppo di funzionalità, eseguendo al contempo circa il 25% di scenari di test automatizzati.

Aumentare la copertura dell'automazione dei test al 75% lascerebbe il 77% di tempo in cui uno sviluppatore si potrà concentrare sull'innovazione.

Grandezza progetto (num linee codice)	Errori tipici per densità
Meno di 2K	0 – 25 errori per 1000 linee di codice
2k – 16k	0 – 40 errori per 1000 linee di codice
16k – 64	0.5 – 50 errori per 1000 linee di codice
64k – 512k	2 – 70 errori per 1000 linee di codice
Oltre 512 k	4 – 100 errori per 1000 linee di codice



E ora.... Come mettersi all'opera



Per evitare problemi come quello che Amazon ha dovuto affrontare, o ancora peggio.

I team DevOps sono tenuti a sviluppare proattivamente i test, automatizzare e testare questi scenari il più presto possibile.

La best practice sviluppare test comprensibili anche a un non sviluppatore è utilizzare l'approccio bdd con un linguaggio gherkin (Domain Specific Language (DSL)).

Questo fa sì che committenti e team di progettazione e sviluppo siano in grado di scrivere test di accettazione in modo semplice

```
# Language it
Funzionalità: Questo

Ci serve un software che porti l'utente lì e là.

scenario: Lì

Dato che l'"utente" si è registrato
quando l'"utente" clicca "quiricchicchi"
Allora l'"utente" deve arrivare alla pagina "lì"

scenario: Là

Dato che l'"utente" ha nel carrello vari prodotti
quando nel conto ha "abbastanza" "lallaralla"
Allora l'"utente" deve arrivare alla pagina "là"
```



Un esempio pratico



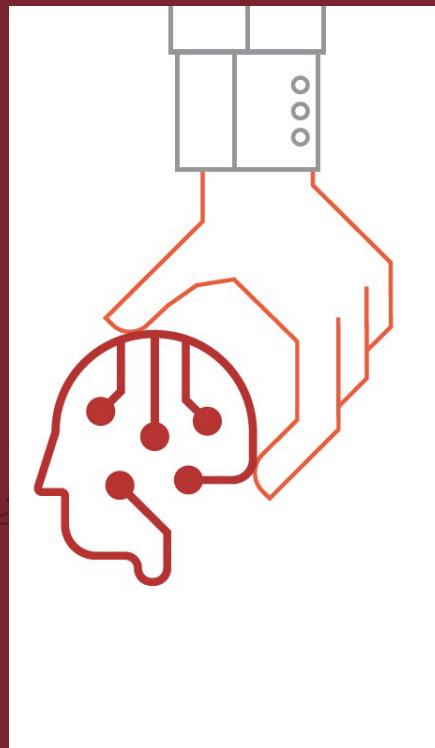


Un esempio pratico



Vediamo come si scrive uno unit test ...





Proposte Tesi e Tirocini

myopportunity@consoft.it

<https://www.consoft.it/it/career>



DOMANDE



GRAZIE!