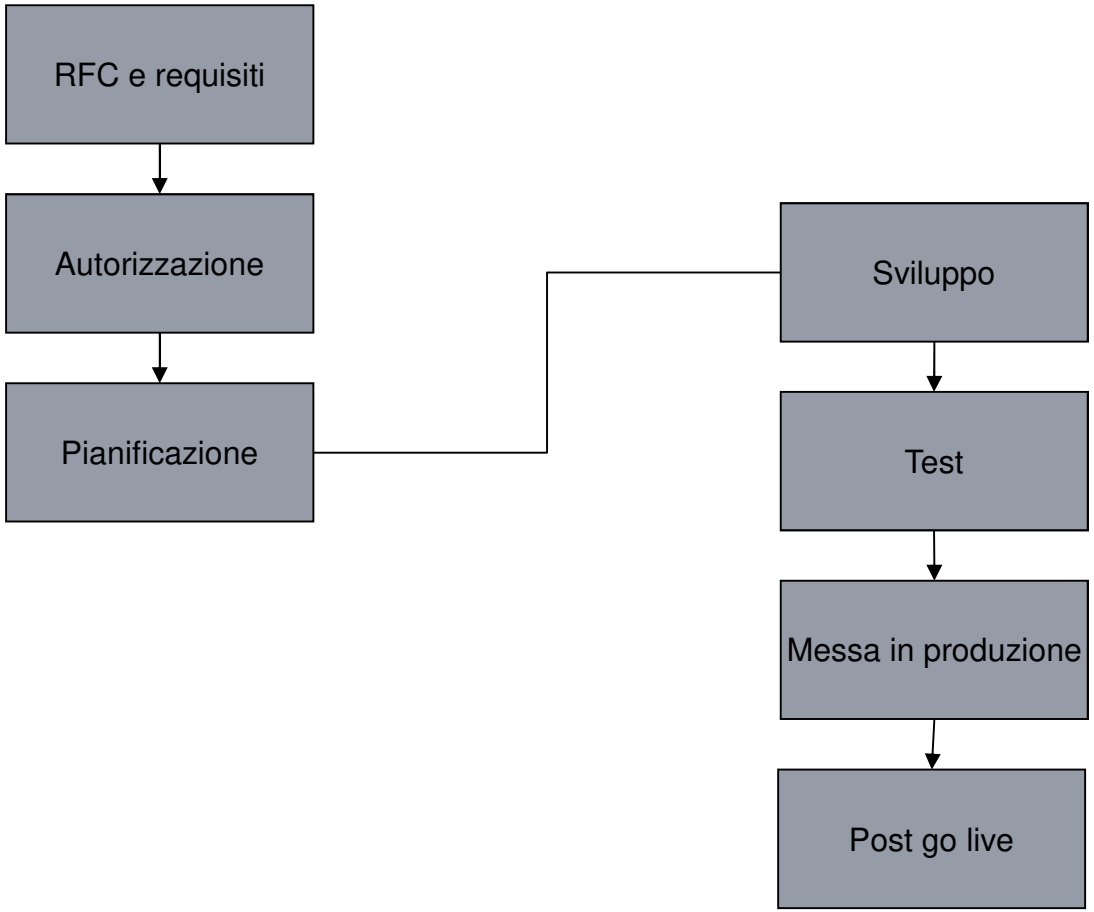
# Spunti per lo sviluppo sicuro del software

Cesare Gallotti

Mestre, 7 ottobre 2022

# Sviluppo software – Come lo conosciamo

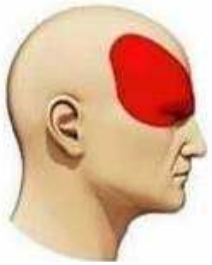


# Qualcosa però non si fa mai

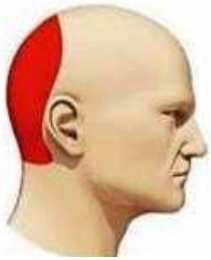
# Requisiti (mal documentati)

## Tipi di mal di testa

Requisiti funzionali



Requisiti di prestazioni



Requisiti architetturali

Requisiti di sicurezza

# Quali sono i requisiti di sicurezza?

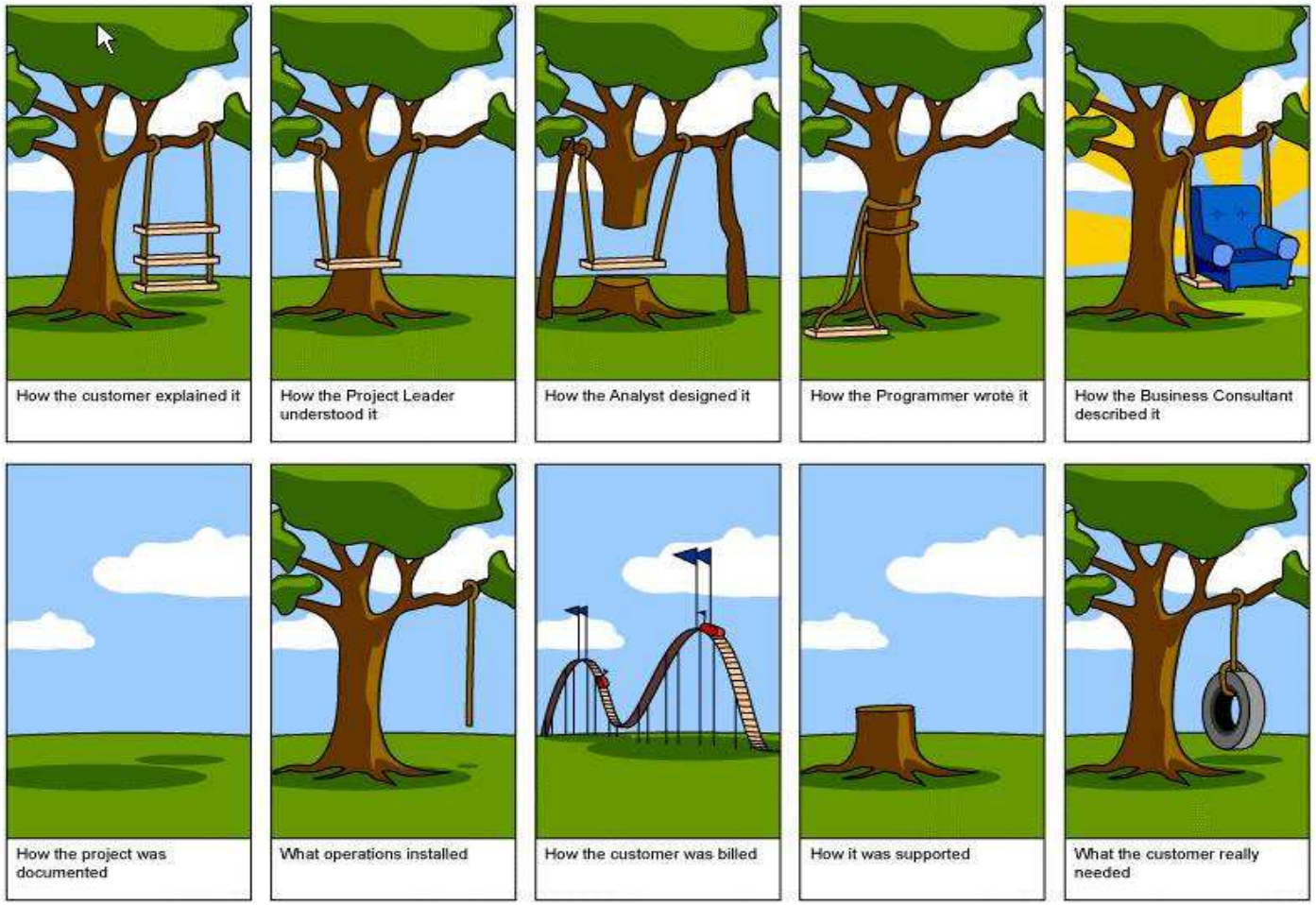
- Funzionali (come gestire le credenziali, le autorizzazioni, i canali cifrati, fail securely, ecc.)
- Architetturali (separazione dell'interfaccia di amministrazione, principi di atomicità e semplicità, ecc.)
- Di codifica (validazione dei dati, dichiarazione delle variabili, ecc.)
- Di connessione con altri sistemi (backup, monitoraggio, ecc.)
- Di manutenzione (aggiornamento librerie, compilatore, ecc.)

Ovviamente... tutti derivati da una valutazione del rischio!

# Dove documentarli?

- Nei documenti di requisiti
- Nelle linee guida interne di sviluppo
- Nelle storie o nei task (negli sviluppi agili)
- Nelle definition of done (negli sviluppi agili)

# Pianificazione

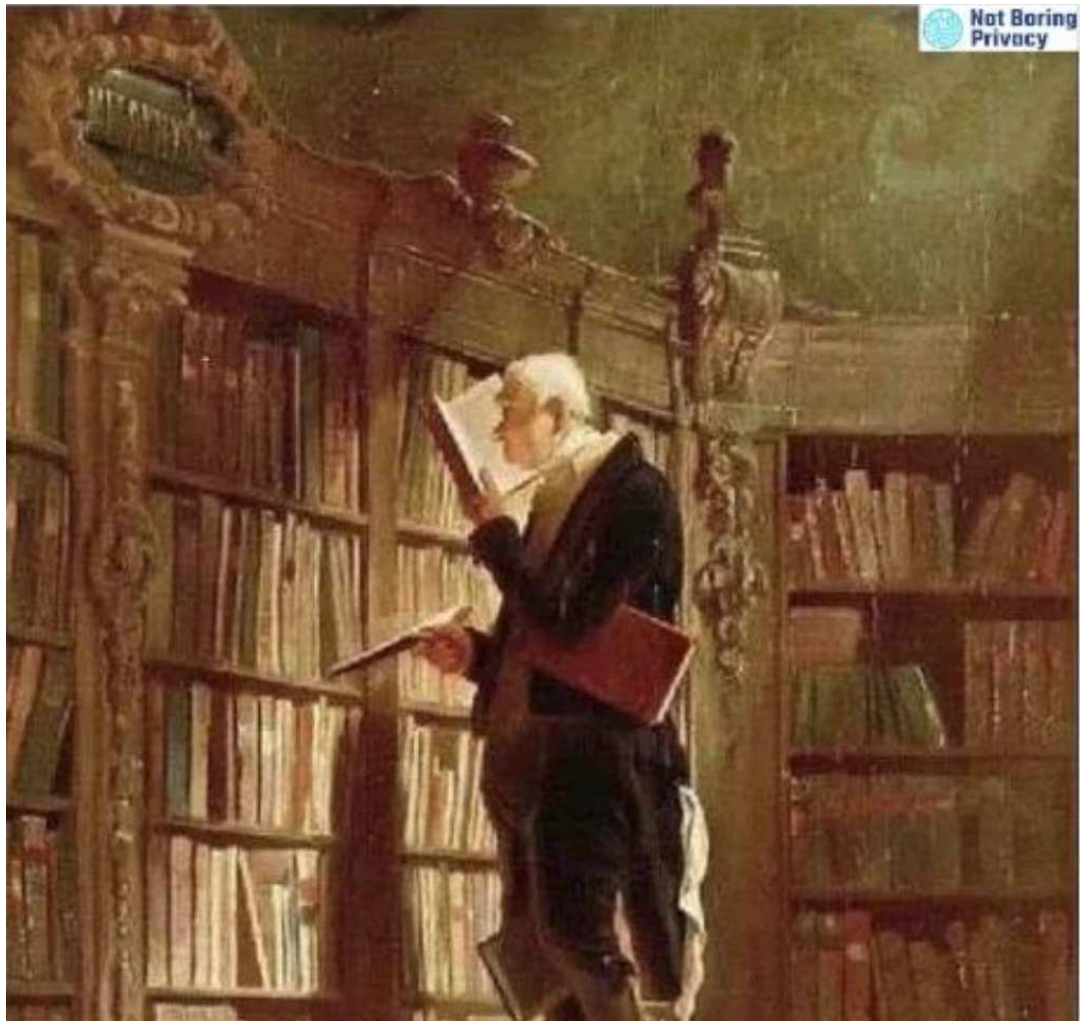



La pianificazione (anche dei test) non si trova neanche nelle vignette!

# Dove pianificare?

- Nel GANTT
- Prevedendo vari stati (inclusi quelli di test) nei programmi di change management
- Nelle definition of done (negli sviluppi agili)

# I test (questi sconosciuti)




I test sicuramente sono fatti, ma poco documentati

# Quali test di sicurezza?

- Funzionali e dis-funzionali (provare a usare password sbagliate, ecc.)
- Regressione
- SAST e DAST (spesso automatizzati)
- Vulnerability assessment e penetration test
- Code review

Ovviamente... tutti derivati da una valutazione del rischio e hanno costi, anche significativi.

# Dove documentarli?

- Nei documenti di test (ricavati da quelli di requisiti; secondo il principio «sviluppo guidato dai test»)
- Negli strumenti automatizzati (per SAST e VA)
- Nelle storie o nei task (negli sviluppi agili)
- Nelle definition of done (negli sviluppi agili)
- In report specifici

Va poi previsto un piano di rientro.

# Perché no?

- Mancanza di persone
- Mancanza di strumenti adatti
- Pressione dai clienti o dalla direzione
- Mancanza di competenze degli sviluppatori

# Mancanza di risorse

- La motivazione più diffusa.
- Lo sviluppo software non è ritenuto critico.
- Molti hanno la percezione che lo sviluppo software non richieda sforzo:
  - intangibilità;
  - disponibilità di software facilmente reperibile anche gratuitamente;
  - immagine informale (giocosa) degli sviluppatori;
  - effetto Dunning-Kruger («individui poco esperti e poco competenti in un campo tendono a sopravvalutare le proprie abilità») sulla Direzione.

# Mancanza di strumenti adatti

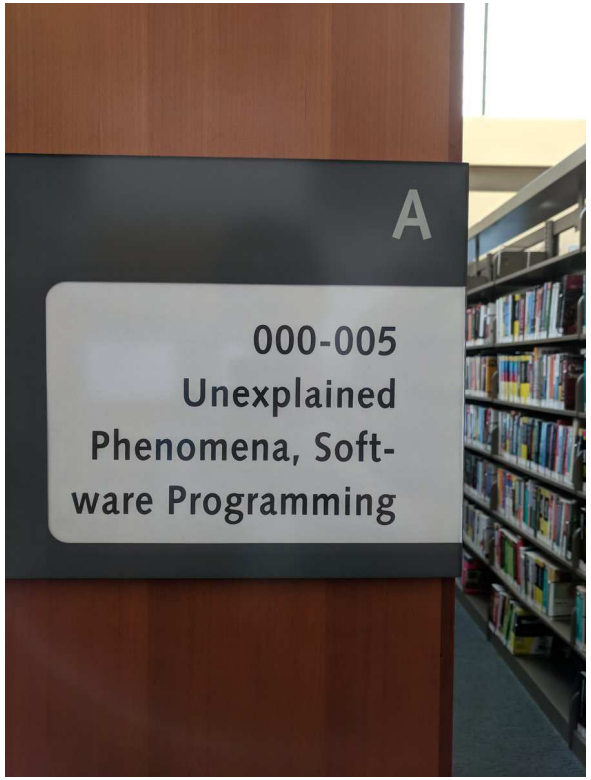
- Gli strumenti costano (anche avere diversi ambienti).
- Bisogna conoscere gli strumenti:
  - gestori del codice;
  - SAST;
  - DAST;
  - strumenti per il passaggio di ambiente (sviluppo, test, produzione).
- Bisogna conoscere gli strumenti non tecnologici:
  - processi;
  - approcci.

# Pressione dai clienti o dalla direzione

- La pressione spinge a sottovalutare le esigenze di sicurezza (si privilegiano le funzionalità).
- Approccio orientato al cliente che spinge a cambiare sempre i piani.
- Molti hanno la percezione che lo sviluppo software non richieda sforzo (come sopra).
- Mancanza di accordo tra gli sviluppatori e commerciali, clienti e Direzione (vedere le considerazioni da cui è nato l'approccio Agile).

# Mancanza di competenze degli sviluppatori

- Effetto Dunning-Kruger (ancora!) sugli sviluppatori.
- Auto-referenzialità all'interno del proprio gruppo.
- Scarsa conoscenza delle tecniche di sviluppo sicuro (spesso si riducono alle Owasp Top 10) e degli strumenti (ancora!).
- Scarsi investimenti e risorse (ancora!) per la formazione degli sviluppatori (convegni, corsi, affiancamento di consulenti specializzati).
  - Spesso si chiede alle Università di affrontare queste questioni, ma non è questo il loro ruolo.
- Spinta verso lo sviluppo di funzionalità e sottovalutazione della sicurezza (ancora!).



# Post scriptum - Elementi emersi dopo la presentazione (1/2)

- Mio spunto: in alcuni casi, gli sviluppatori non seguono un processo di sviluppo sicuro perché consulenti e auditor lo complicano troppo (p.e. richiedendo documenti eccessivi rispetto al rischio).
- Un partecipante mi ha raccontato che ha avuto molti insuccessi nell'attuare soluzioni di sviluppo sicuro nella sua azienda (con più di 100 sviluppatori), perché soluzioni centralizzate o non centralizzate hanno i loro difetti. Al momento ha:
  - persone di una funzione di sicurezza centrale che intervengono sui progetti (requisiti, regole, test);
  - persone specializzate in sicurezza («champion») in ogni gruppo di progetto;
  - l'impegno dei capi progetto a lasciare il tempo per la sicurezza (requisiti, test, analisi dei risultati di SAST e DAST, VA-PT, piani di rientro), anche se sono completamente occupati.

# Post scriptum - Elementi emersi dopo la presentazione (2/2)

- Un partecipante ha segnalato il fatto che bisogna partire da attività di base (p.e. strutturare un processo di gestione degli incidenti) e solitamente le persone sono disposte a collaborare.
- C'è molto materiale, anche gratuito, per sensibilizzare sulla sicurezza. Più difficile trovare materiale più tecnico. Alcuni erogano corsi di sviluppo sicuro molto tecnici, ma sono poco noti.
- Un partecipante ha segnalato il fatto che è importante assicurare la disponibilità di meccanismi di sicurezza nella piattaforma di sviluppo (attività quindi centralizzata), in modo che le persone siano forzate ad attuare le pratiche di sicurezza.
- Gli sviluppatori perdono tempo anche perché troppo spesso il cliente (interno o esterno) non sa bene cosa vuole e richiede di elaborare e rielaborare i requisiti funzionali (e quindi è difficile seguire con quelli di sicurezza).

# Grazie!


- Cesare Gallotti
  cesaregallotti@cesaregallotti.it
  PEC: cesaregallotti@mailcert.it
  http://www.cesaregallotti.it
  http://blog.cesaregallotti.it