



# Insecure Design

...tales





# Chi sono io



# Agile Coach Software architect Software engineer

**Andrea Torino Rodriguez**

[andrea.torino.rodriguez@gmail.com](mailto:andrea.torino.rodriguez@gmail.com)

<https://www.linkedin.com/in/andrearodriguez/>

@agilerod



# Di cosa parliamo?



insecure design tales





# Open Web Application Security Project



9 settembre  
**2001**



1 dicembre  
**2001**



21 aprile  
**2004**



giugno  
**2011**





insecure design tales

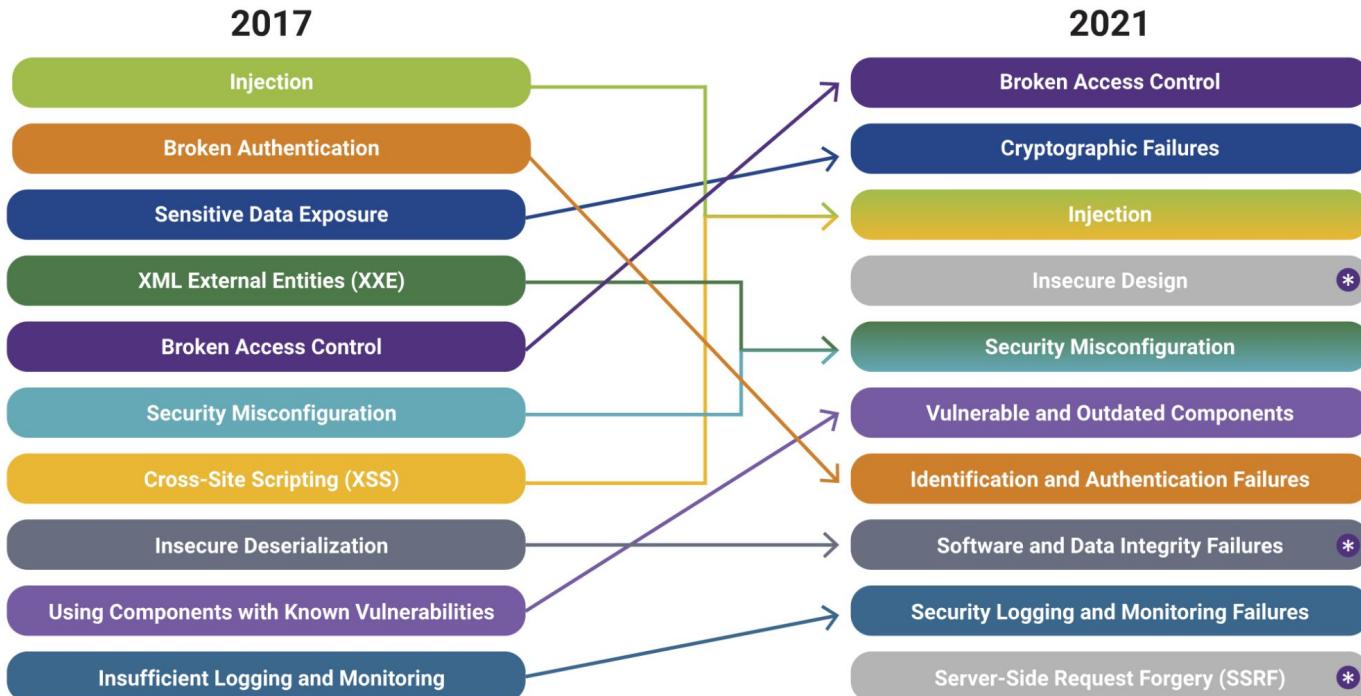


# TOP10

---



# OWASP Top10 2024 (2021)





# OWASP Top10 2024 (2021)





# Cos'è?



# Insecure Design by OWASP

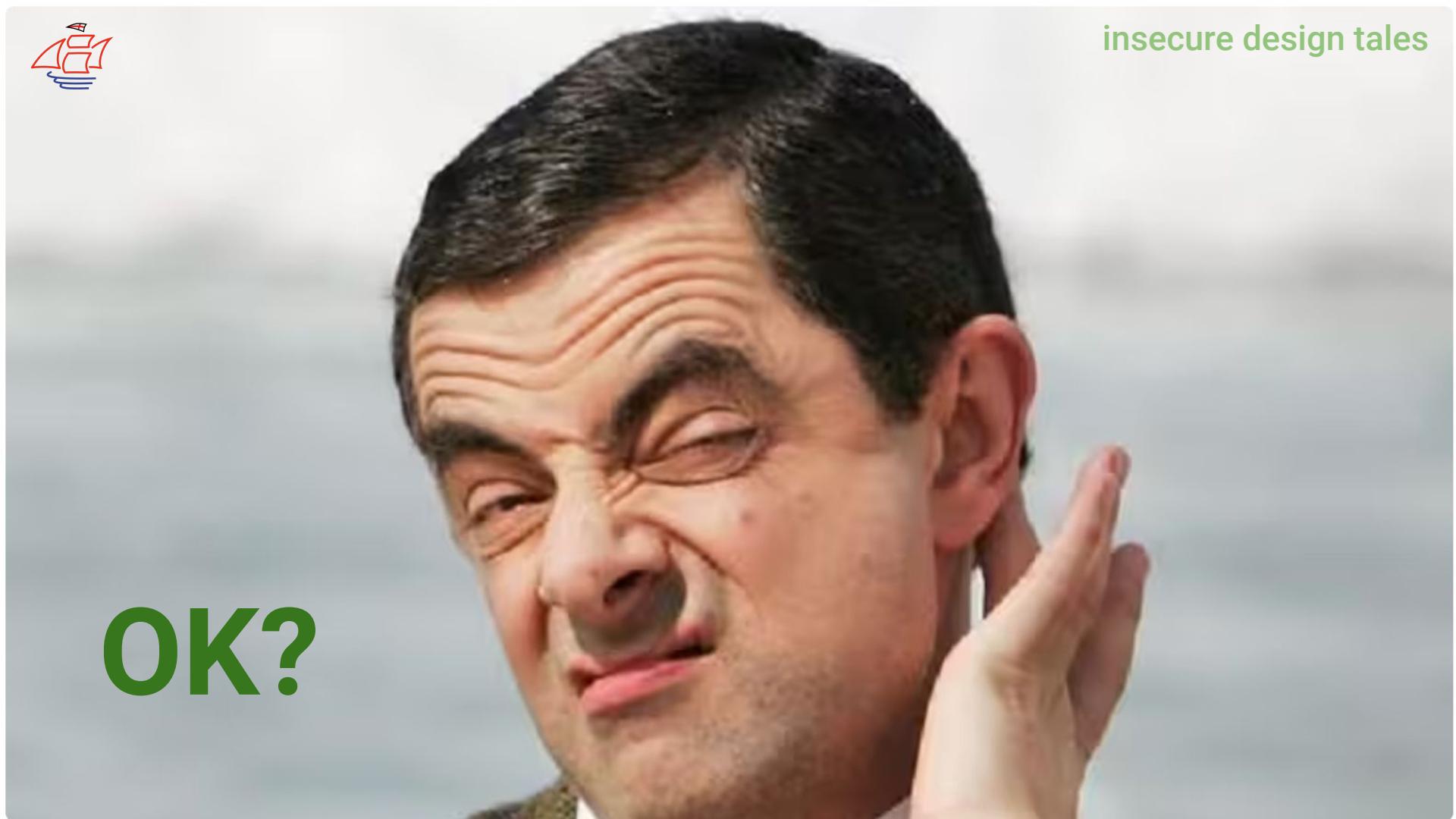
*“Si concentra sui rischi correlati a difetti di progettazione e architettura.”*

*“Invita a un maggiore utilizzo della modellazione delle minacce, di modelli di progettazione sicuri e di architetture di riferimento.”*



# Insecure Design by OWASP

*“Una progettazione insicura non può essere risolta da un’implementazione perfetta.”*

A close-up photograph of Mr. Bean's face. He has his signature weary expression with deeply wrinkled brows and a slightly downturned mouth. He is holding a white mobile phone to his right ear with his index finger pointing towards the receiver. The background is a soft-focus outdoor scene.

OK?



# Cosa NON è



***Insecure Design***

**NON E'**

***Insecure Implementation***



# Insecure Implementation



# Insecure Design



## Insecure Design

## vs Insecure Implementation

- **Cultura**
- **Processo (di sviluppo)**
- **Progettazione**
- **Architettura**

- **Metodologia**
- **Tecnicismi**
- **Capacità/lacune dei Team**
- **Testing**



# Meglio?



## Insecure Design

## vs Insecure Implementation

- **Cultura**
- **Processo (di sviluppo)**
- **Progettazione**
- **Architettura**

- **Metodologia**
- **Tecnicismi**
- **Capacità/lacune dei Team**
- **Testing**



# Cultura





# Cultura della sicurezza

**Coinvolgimento del business (requisiti)**

**Assegnare un budget specifico**

**Promuovere la conoscenza in azienda**

**Formare e individuare figure specifiche**

**OWASP SAMM (Software Assurance Maturity Model)**

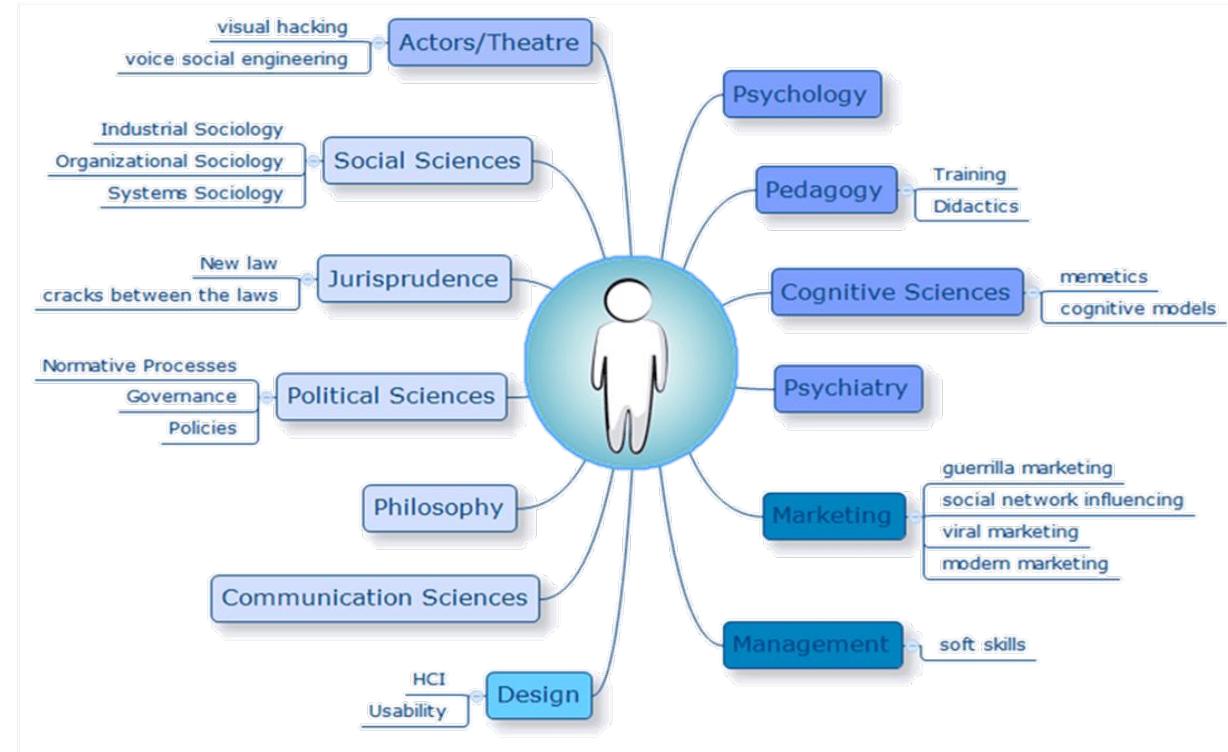


# Cultura della sicurezza

Formare  
figure  
specifiche

(c.d. cybersecurity advocates)

Non banale...





# Processo





# Processo di sviluppo (sicuro)

**Uso di pattern “sicuri” (OWASP ...)**

**Metodologie di sviluppo solide, automazione**

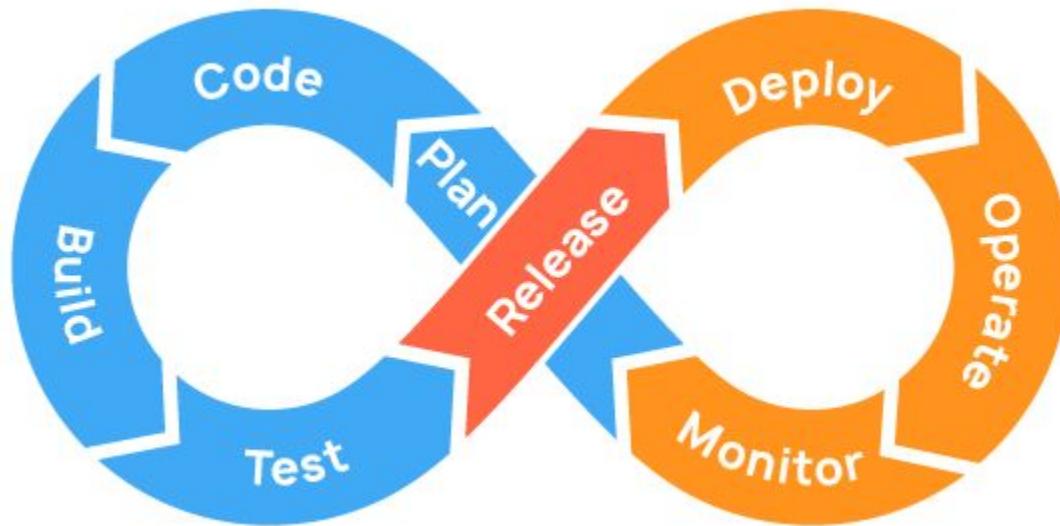
**Componenti e librerie “sicure”**

**Specialisti della sicurezza nel Team**

**Seguire un SAMM (e aggiornarlo)**



# Processo di sviluppo (sicuro)

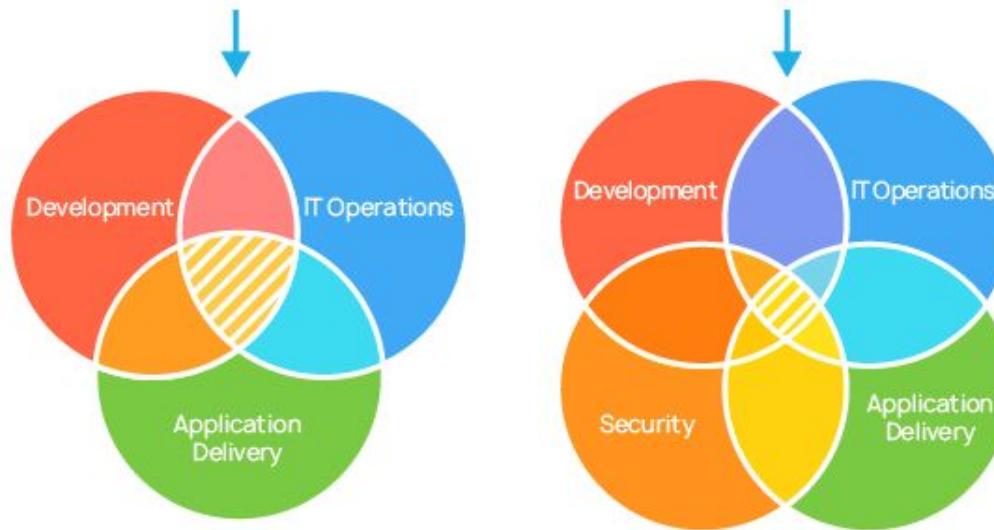


Automazione e monitoring



# Processo di sviluppo (sicuro)

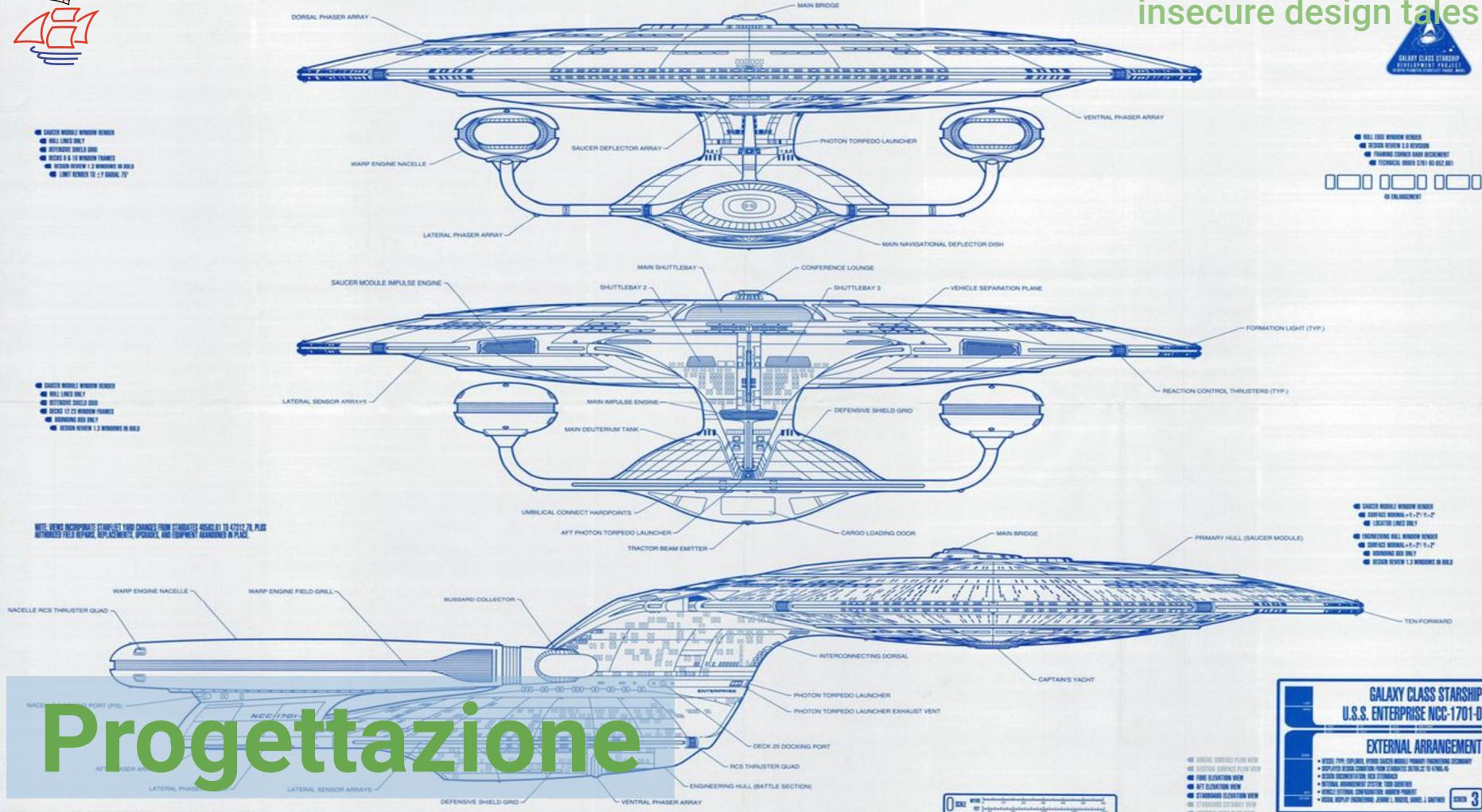
## DevOps VS DevSecOps



Esplicitare l'ambito di Sicurezza



# Progettazione





# Progettazione (della sicurezza)

**Applicazione principi “Security by Design”**

**Applicazione “Separation of concerns”**

**Pianificazione strategia di prevenzione**

**Progettazione della manutenzione**

**Pianificazione verifiche della vulnerabilità**



# Progettazione (della sicurezza)

***“Security by Design”***  
implicazioni sulla qualità



# Progettazione (della sicurezza)

## Software sicuro; definizione

*La pratica di scrivere, sviluppare e mantenere codice sorgente in modo tale da ridurre al minimo il rischio di esposizione a vulnerabilità e minacce di sicurezza.*



# Progettazione (della sicurezza)

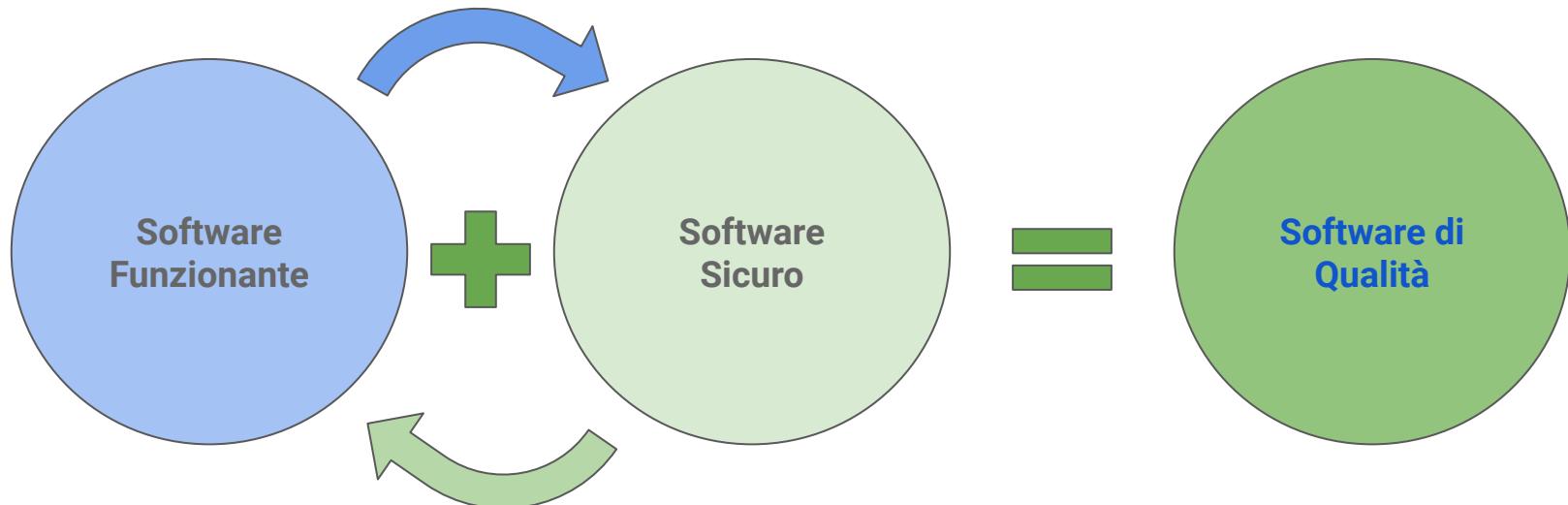
Software funzionante; definizione

**Il software funzionante (*Working Software*) è completamente integrato, testato e pronto per essere fornito al cliente o messo in produzione.**

*(Ciò non significa che è stato provato un paio di volte  
e ha funzionato senza rompersi...)*



# Progettazione (della sicurezza)





# Architettura



# Architettura

**“Threat Modeling” (STRIDE, PASTA, VAST, TRIKE ...)**

**Implementazione “Separation of concerns”**

**Esplicitare il contesto di sicurezza**

**Verifica scenari vulnerabilità**

**Dimensionamento delle risorse (preventivo)**



# Architettura

The screenshot shows a web browser displaying the Threat Modeling Manifesto website at [threatmodelingmanifesto.org](https://threatmodelingmanifesto.org). The page features a large green header with the word "Architettura". Below the header is a navigation bar with links for Threat Modeling Manifesto, Values, Principles, About, and Threat Modeling Capabilities (marked as "New"). The main content area includes a logo consisting of a globe with arrows, followed by the text "THREAT MODELING MANIFESTO". A large green call-to-action button at the bottom contains the URL "https://www.threatmodelingmanifesto.org". The page also contains sections for "What is threat modeling?", "Who should threat model?", and "How should I use the Threat Modeling Manifesto?", each with associated text and a "Read more" link.

What is threat modeling?

Threat modeling is analyzing representations of a system to highlight concerns about security and privacy characteristics.

At the highest levels, when we threat model, we ask four key questions:

1. What are we working on?
2. What can go wrong?
3. What are we going to do about it?
4. Did we do a good enough job?

<https://www.threatmodelingmanifesto.org>

When you perform threat modeling, you begin to recognize what can go wrong in a system. It also allows you to pinpoint design and implementation issues that require mitigation, whether it is early in or throughout the lifetime of the system. The output of the threat model, which are known as threats, informs decisions that you might make in subsequent design, development, testing, and post-deployment phases.

Who should threat model?

You. Everyone. Anyone who is concerned about the privacy, safety, and security of their system.

How should I use the Threat Modeling Manifesto?

Use the Manifesto as a guide to develop or refine a methodology that best fits your needs. We believe that following the guidance in the Manifesto will result in more effective and more productive threat modeling. In turn, this will help you to successfully develop more secure applications, systems, and organizations and protect them from threats to your data and services. The Manifesto contains ideas, but is not a how-to, and is methodology-agnostic.



# Architettura

Cos'è?  
breve approfondimento



# Architettura

# 1990'

**Stili (patterns)**  
**Linguaggi (ADLs)**  
**Documentazione**  
**Metodi formali**



# Architettura

1996

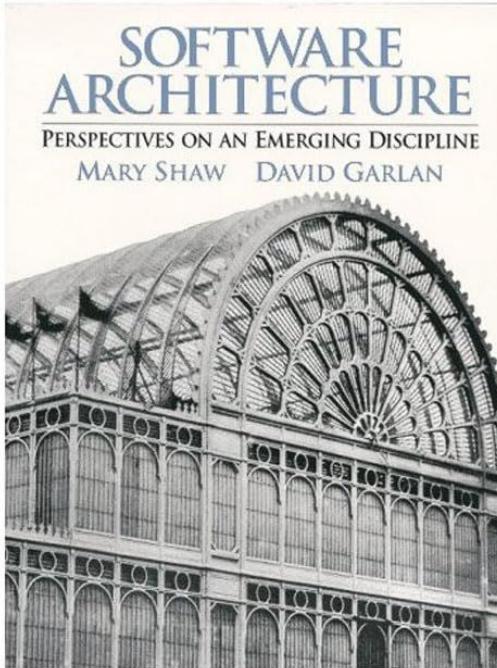
Mary Shaw  
David Garlan



Carnegie Mellon University



# Architettura



## Principali concetti

Stili,  
Connettori,  
Componenti...



# Architettura

2000-2011+

IEEE 1471-2000

ISO/IEC 42010

ISO/IEC 15288

ISO/IEC 12207 (\*)



# Architettura

## Definizione IEEE/ANSI 1471-2000

*“L’architettura software è l’organizzazione di base di un sistema, espressa dalle sue componenti, dalle relazioni tra di loro e con l’ambiente, e i principi che ne guidano il progetto e l’evoluzione.”*



# Architettura

2017

ISO/IEC  
12207:2017

Systems and software engineering  
Software life cycle processes

*"humans, processes, procedures,  
facilities, materials and naturally  
occurring entities"*



# Architettura

## Caratteristiche (buone)

**Astratta (metafora)**

**Quality driven**

**Emergente**

**Soddisfa più parti**



# Architettura

## Problemi

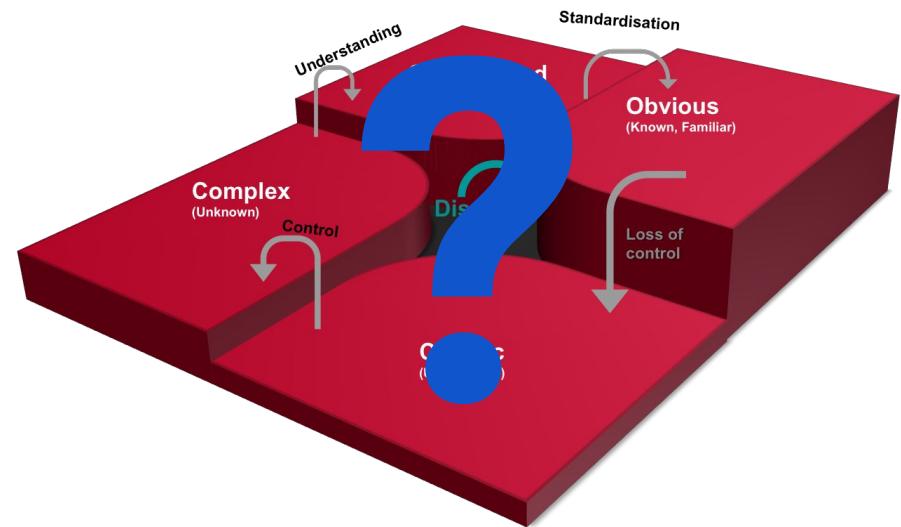
2 problemi principali...



# Architettura

## Complessità

non eliminabile  
non eludibile  
non lineare  
affrontabile empiricamente



*Cynefin framework, Dave Snowden 1999*



# Architettura

Ridurre la complessità:

***“separation of concerns”***

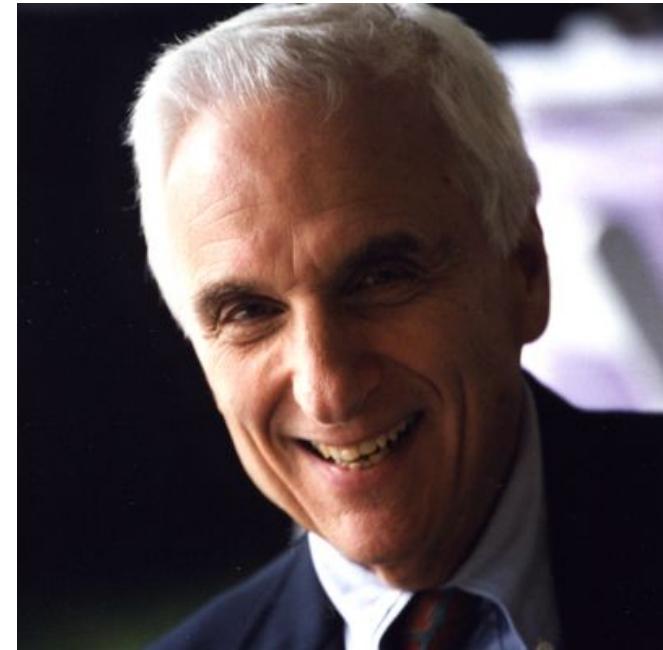
*“Descrivere l’architettura da punti di vista separati associati alle diverse preoccupazioni delle parti interessate. Queste descrizioni separate sono chiamate viste architettoniche.”*



# Architettura

## Conway's Law

Ogni organizzazione che progetti un sistema (sistemi informativi, e non solo) produrrà inevitabilmente un design la cui struttura è una copia della struttura comunicativa dell'organizzazione.



*Melvin Conway, 1967*



# Architettura

## Strategia inversa alla Conway's Law

*“Eliminare i silos che vincolano la capacità del Team di collaborare in modo efficace”*

*Jonny Leroy, Matt Simons 2010*



# Architettura

Caratteristiche  
1 peculiarità...



# Architettura

## Granularità

Il concetto di architettura del software, può essere applicato - e definire - diversi insiemi di componenti del sistema, dagli strati più “grossolani” fino a quelli più dettagliati (il codice)





# Architettura



## Architettura

**"Threat Modeling"** (STRIDE, PASTA, VAST, TRIKE ...)

**Implementazione "Separation of concerns"**

**Esplicitare il contesto di sicurezza**

**Verifica scenari vulnerabilità**

**Dimensionamento delle risorse (preventivo)**



**Hexagonal**



# Domain Driven Design

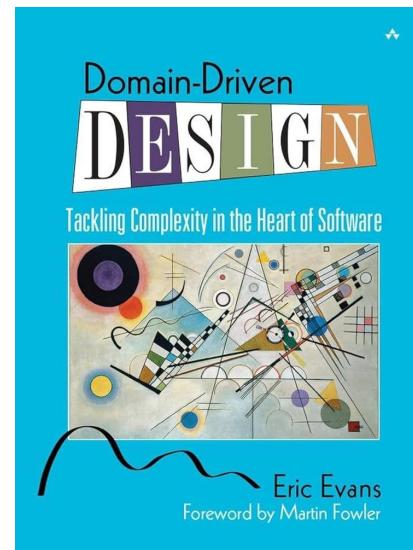


# Architettura

## D.D.D.

Domain Driven Design  
si concentra sulla modellazione del  
dominio in base all'input degli  
esperti di quel dominio.

In DDD la struttura e il linguaggio  
del codice devono corrispondere al  
dominio aziendale.



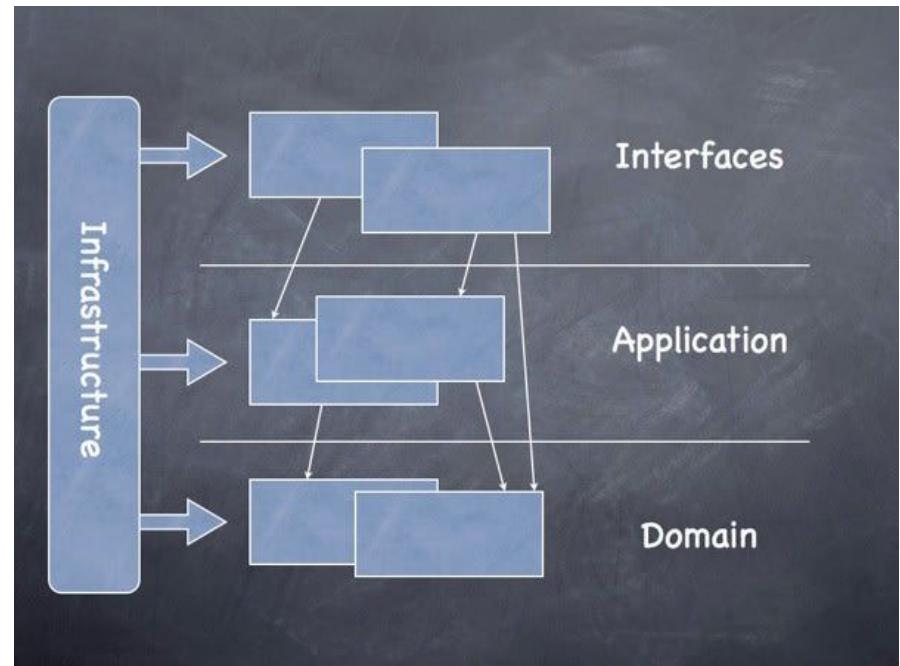
*Eric Evans, 2003*



# Architettura

## Domain Driven Design

Identifica 3 principali layer; Interfaces, contiene ciò che viene esposto dal sistema, Application contiene ciò che necessita al funzionamento, Domain contiene lo stato del sistema. Infrastructure è un service layer trasversale comune ai tre.





# Architettura

## Domain Model: cos'è

*“An object model of the domain that incorporates both behavior and data”*

*Martin Fowler, “Patterns of enterprise application architecture”, 2003*



# Architettura

## Domain Model: cos'è

**Una delle componenti a maggior valore aggiunto dell'applicazione**

**Ha aspettativa di vita indipendente dalla tecnologia circostante**

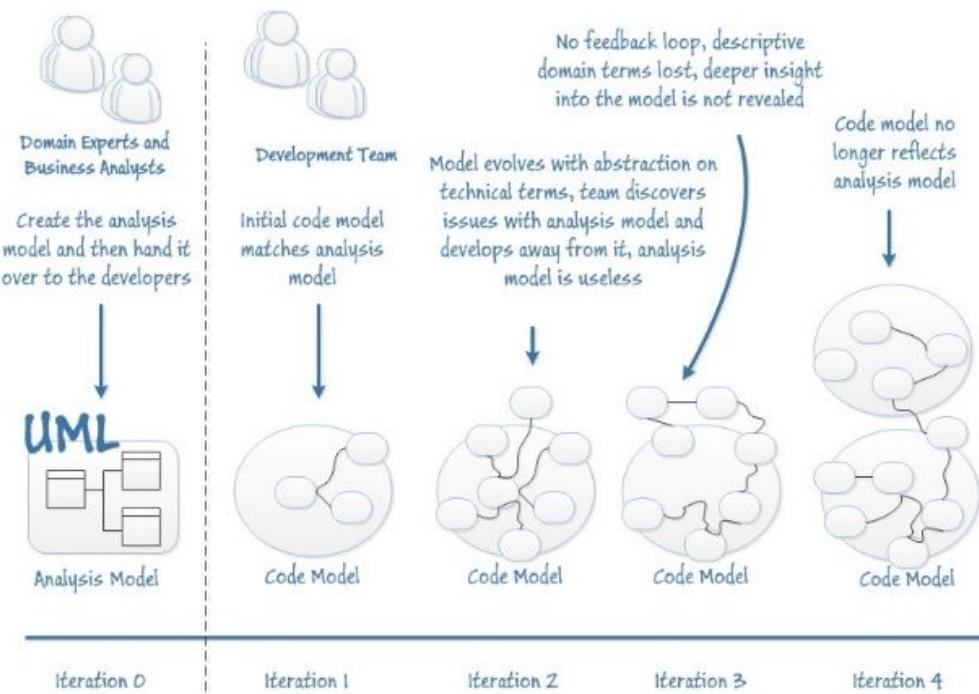
**Area in cui le modifiche sono frequenti in risposta a specifiche esigenze di business**



# BDUF

Non evolve

# Architettura

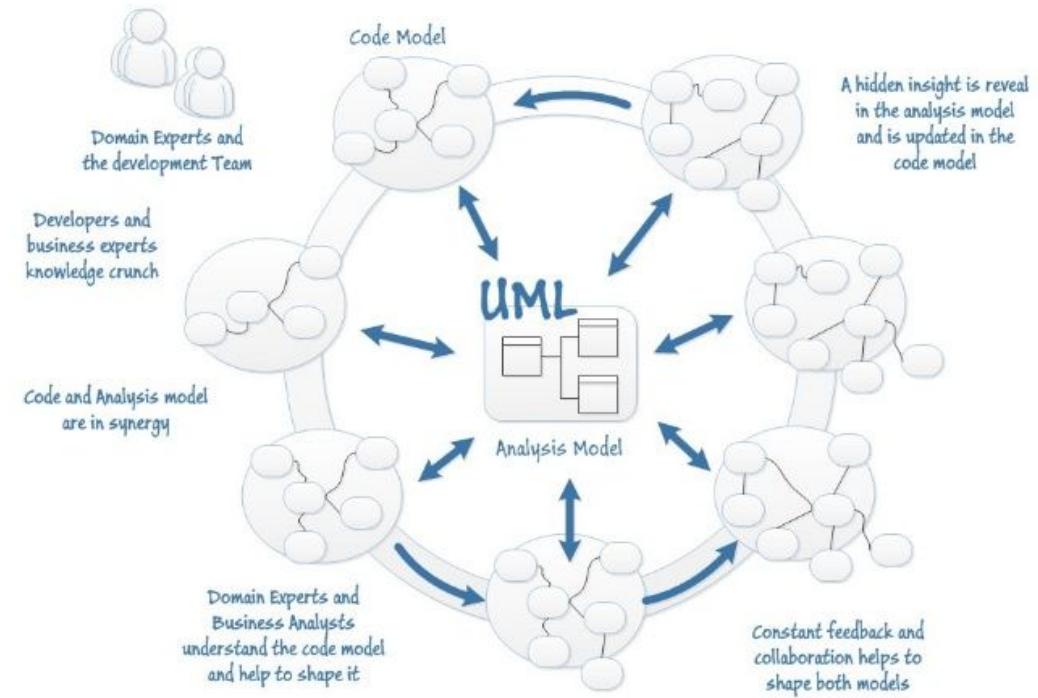




# Architettura

# DDD

## Le evoluzioni del codice tornano al modello





# Architettura

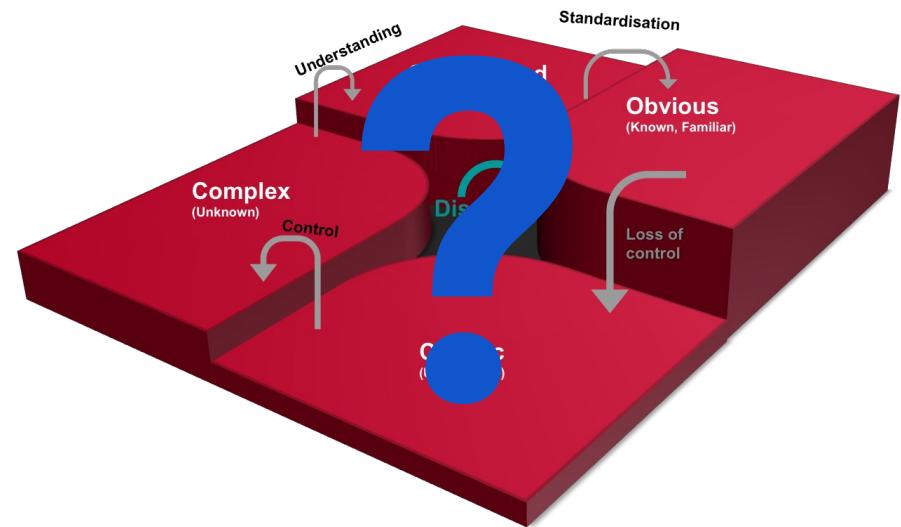
**DDD: perchè usarla?**  
**implicazioni sulla sicurezza**



# Architettura

## Complessità

non eliminabile  
non eludibile  
non lineare  
affrontabile empiricamente



*Cynefin framework, Dave Snowden 1999*



# Architettura

Ridurre la complessità:

***“separation of concerns”***

*“Descrivere l’architettura da punti di vista separati associati alle diverse preoccupazioni delle parti interessate. Queste descrizioni separate sono chiamate viste architettoniche.”*



# Architettura

## DDD: implicazioni con la Sicurezza

***La “separation of concerns”  
è difficile da raggiungere***



# Architettura

## DDD: implicazioni con la Sicurezza

*In alcuni casi, la separazione degli interessi prescritta dal DDD è difficile da raggiungere.*

*Ciò avviene quando viene considerata una funzionalità che dev'essere indipendente dal dominio, ma che lo pervade fortemente ed è strettamente correlata alle funzionalità relative al dominio.*



# Architettura

## DDD: implicazioni con la Sicurezza

***La sicurezza è concettualmente una funzionalità indipendente dal Dominio***

***Quindi dovrebbe essere modellata come un servizio (service) in DDD ...***



# Architettura

DDD: implicazioni con la Sicurezza

***E' difficile separare la sicurezza dalle funzionalità relative al Dominio***

***Per farla funzionare correttamente, occorre che pervada fortemente le funzionalità del Dominio***



# Architettura

## DDD: problemi con service

1. *Un security service deve autorizzare azioni che influiscono sull'implementazione degli oggetti di Dominio.*  
*Ciò succede quando queste azioni vengono avviate da altri servizi; ad es: un service che restituisce dei ValueObject*

*Quindi il security service dovrebbe essere consapevole di queste azioni, in modo da poterle regolare...*



# Architettura

## DDD: problemi con service

2. *Se la sicurezza è unicamente basata sul ruolo dell'utente, le funzionalità di sicurezza potrebbero essere contenute in un security service con basso accoppiamento.*

*Però quando l'implementazione dello stato del Dominio influisce sul controllo degli accessi, il disaccoppiamento delle funzionalità di sicurezza diventa più problematico. Il security service dovrebbe avere consapevolezza del Dominio.*



# Architettura

## DDD: implicazioni con la Sicurezza

***Un forte accoppiamento fra un service e l'implementazione del Dominio contravviene ai principi DDD.***



# Architettura

## DDD: implicazioni con la Sicurezza

*“I problemi di sicurezza dovrebbero essere gestiti all'esterno del Dominio”*

*“I requisiti di controllo degli accessi sono specifici del Dominio”*



# Architettura

DDD: implicazioni con la Sicurezza

Come fare?

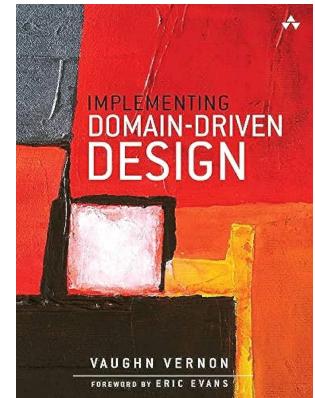


# Architettura

## Implementing DDD

Sicurezza e permessi  
andrebbero centralizzati nel  
proprio Bounded Context, che  
viene poi utilizzato da gli altri  
Bounded Context

(anche “Generic Subdomain”)



Vaughn Vernon, 2005



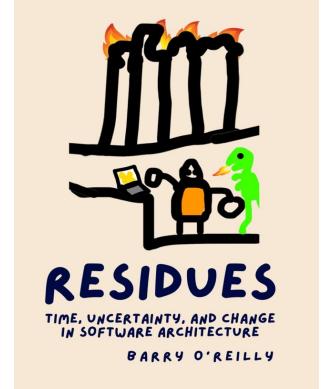
# Altre idee...



# Architettura

## Residuality Theory

Teoria della Residualità: un insieme di idee per produrre un set di strumenti molto leggero che ci aiuti a gestire l'incertezza e a trasformare situazioni incerte in architetture software coerenti. (anche "Architettura Antifragile")



Barry O'Reilly, 2024



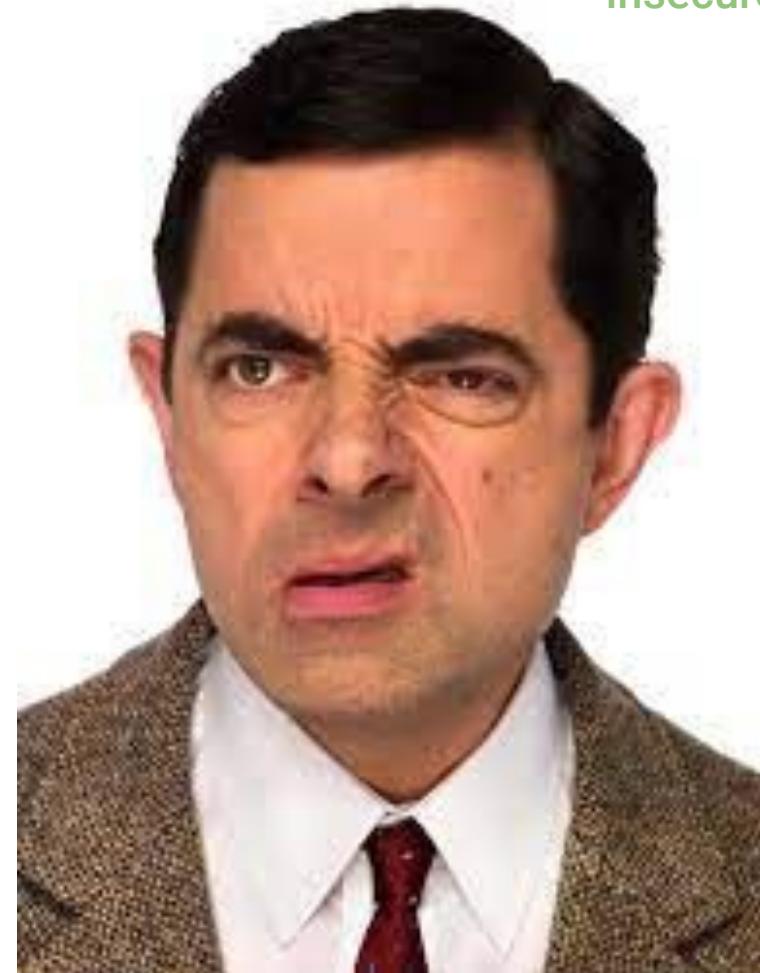
# Architettura

Residuality Theory: implicazioni...





# Domande?





# Grazie