Appunti di
Architetture del Software

A cura di:
Francesco Refolli
Matricola 865955

Anno Accademico 2024-2025

Chapter 1

Architetture del Software

1.1 Definizioni

Ci sono diverse definizioni di Architettura del Software, tutte ugualmente valide, ma noi ci concentriamo su alcune:

Perry & Wolf 1992 L'Architettura ha a che fare con la selezione degli elementi architetturali, le loro interazioni, i loro vincoli necessari a disporre un framework che soddisfi i requisiti del sistema (funzionali e non).

Garlan & Shaw L'architettura del software è un livello di design che va oltre gli algoritmi e le strutture della computazione; si specifica la struttura del sistema ad alto livello.

Garlan & Shaw 1996 L'architettura di un ssitema software definisce il sistema in termini di componenti computazionali e le loro interazioni in modo da soddisfare i requisiti del sistema.

IEEE 1420-2100 L'architettura del software è l'organizzazione del sistema, strutturato nei suoi componenti, nella sue relazioni tra questi e l'ambiente e i principi che governano il design e l'evoluzione del sistema.

E ci interessa questa definizione:

Clemens & al. 2010 L'architettura del software di un sistema è l'insieme delle strutture necessarie a ragionare sul sistema che comprendono: elementi <software>, relazioni e proprietà.

1.2 Elementi

Una struttura include o definisce elementi. Un elemento architetturale è una parte fondamentale che costituisce il sistema: che impatta sulla design e sulle qualità chiave del sistema. Es: un algoritmo di ordinamento di dati estratti da query non è un elemento, ma la struttura che si sceglie per immagazzinare o organizzare i dati è un **elemento architetturale**.

Gli elementi possono essere **software** (moduli, processi, servizi ...) o **non software** (unità fisiche di storage, unità di dispiegamento, squadra di sviluppo ...).

Elementi Software È importante definire bene quali sono le **responsabilità**, il **livello di qualità** e le **interfacce** (cosa offre e di cosa necessità) di un elemento software. Si dice che cosa ad alto livello quel componente fa, ma non come viene sviluppato o implementato.

1.3 Strutture

Un'architettura contiene moltemplici (tipi di) strutture: struttura statica (struttura), dinamica (flusso), di dispiegamento, di sviluppo ... con vari punti di vista perchè ognuna ha l'obiettivo di catturare i requisiti e le necessità degli stakeholders. Ogni struttura definisce solo un sottoinsieme delle informazioni e può contenere relazioni con componenti di altre strutture.

1.3.1 Categorie

strutture di moduli Partizionano il sistema in moduli (architettura logica)...

componenti-e-connettore Si concentrano sulle relazioni a runtime tra i componenti del sistema. I componenti per definizione sarebbero le unità indivisibili di esecuzione/dispiegamento, ma noi trattiamo un programma come composto da componenti. Le relazioni tipicamente implementano le relazioni tra i moduli.

allocazione Descrivono la mappatura della struttura del software all'ambiente in cui esiste, si sviluppa ed esegue.

1.3.2 Relazioni e Proprietà

Le relazioni possono essere tra componenti di strutture diverse o nella stessa struttura. Le proprietà degli elementi descrivono il suo comportamento o la qualità delle sue azioni. Sono assunzioni che possono essere fatte su ogni elemento dal punto di vista degli altri elementi.

Un attributo di qualita' è una proprieta' **misurabile** e **testabile**.

1.3.3 Dettaglio

L'architettura del software descrive come le proprietà dei singoli componenti determinano le proprietà ad alto livello del sistema.

Nei diagrammi non si vuole modellare se non il caso normale, c'è una vista apposita per le eccezioni. Di norma i casi anomali non devono essere modellati nelle strutture generali. La vista delle eccezioni può essere fondamentale per definire dei comportamenti di sistemi critici dove è necessario prevedere un comportamento di emergenza del sistema (esempio SPID).

Si omettono tutti i dettagli di implementazione. Sono documenti di **astrazione**.

1.3.4 Stakeholders

Utenti, gruppi, soggetti, entità che hanno interessi nel sistema sviluppato. Anche il programmatore, anche i manutentori, anche i tester. Letteralmente chiunque, anche chi disegna l'architettura del software. Se ho tanti punti di vista (gli stakeholder di norma hanno il proprio) allora avrò tante strutture (numero indefinito). Loro definiscono le necessità e le condizioni di vittoria.

Un interesse (concern) è definito come un requisito, un obiettivo, un vincolo, un'intenzione o un'aspirazione dello stakeholder per il sistema. È una definizione molto larga. L'architettura deve concentrarsi sugli interessi più significativi e astratti.

Un requisito che ha un impatto significativo nell'architettura del software e sugli attributi di qualità del sistema è definito **Requisito Architetturalmente Significativo**.

Esistono informazioni aggiuntive di cui bisogna tenere nota e che sono specificate a parte.

I requisiti, i vincoli, le assunzioni vanno documentate. Le scelte architetturali vanno giustificate e spiegate. Supporta la comunicazione con gli Stakeholders e aiuta lo sviluppo.

L'architetto identifica gli stakeholder e capisce i loro interessi, prende decisioni o aiuta a prendere decisioni, pretende cambiamenti e trova buoni compromessi.

È necessario coordinare le squadre che sviluppano, mantengono o testano il sistema perchè ognuna è responsabile di una piccola parte del sistema ed è necessario fare in modo che si parlino per tenere le dipendenze tra moduli, componenti e parti assegnate.

Legge di Conway La struttura dei progetti ricalca la struttura organizzativa di chi lo sviluppa.

1.4 Cosa rende una architettura "buona"

Non esiste. Esistono solo compromessi. Un'architettura deve aderire a vincoli, interessi e intenti degli stakeholders. Esistono poi delle "regole" su come si struttura l'architettura del software:

- L'architetto dovrebbe essere uno solo, o un gruppo molto ristretto.
- l'architetto dovrebbe basare l'architettura su una lista di requisiti (funzionali, non funzionali) ordinata secondo le priorità.
- l'architettura dovrebbe essere ben documentata, altrimenti i cambiamenti e la gestione delle emergenze sono lunghe e costose.
- l'architettura dovrebbe essere valutata costantemente e se possibile da soggetti esterni.
- fare l'architettura in modo incrementale in modo da riparare rapidamente agli errori.
- Ogni elemento deve essere definito in modo chiaro, ma nascondento le informazioni non importati o suscettibili.
- Usare pattern e strategie per raggiungere gli obiettivi di qualita' imposti.
- Bisogna evitare di dipendere da determinate versioni o funzionalita' di tecnologie (sistema operativo, API, librerie).

- I moduli che consumano dati devono essere diversi da quelli che li producono.
- Non aspettarsi corrispondenze uno-a-uno tra moduli e componenti.

...

...

...