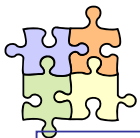


# Architetture Software

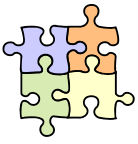
## Architetture software: Concetti

Dispensa AS 2  
ottobre 2008



### - Fonti

- [SSA] Chapter 2, Software Architecture Concepts
- [IEEE-1471] IEEE Recommended Practice for Architectural Description of Software Intensive-Systems
- [SAP/2e] Chapter 2, What is Software Architecture?



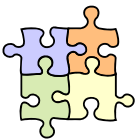
## - Obiettivi e argomenti

### □ Obiettivi

- introdurre alcuni concetti fondamentali delle architetture software

### □ Argomenti

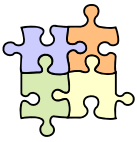
- architetture software
- elementi architettonici
- parti interessate ed interessi
- descrizioni architettoniche
- relazioni tra concetti
- discussione



## \* Architetture software

### □ Le architetture software riguardano le strutture dei grandi sistemi software

- i grandi sistemi possono sempre essere decomposti in elementi/parti/sotto-sistemi del sistema
  - si tratta di elementi a grana grossa
  - ogni sistema può essere opportunamente decomposto in un numero (relativamente piccolo) di elementi
- le architetture sono interessate (almeno a)
  - gli elementi del sistema
  - le relazioni tra questi elementi
- le architetture influiscono sul raggiungimento di qualità (requisiti non funzionali) desiderate dalle parti interessate

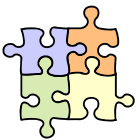


## Architettura software - alcune definizioni

- Alcune possibili definizioni di architettura software

 [Perry&Wolf, 1992]

- **software architecture** =  
{ elements, form, rationale }
- elements
  - processing elements
  - data elements
  - connecting elements
- form
  - vincoli sulle relazioni e interazioni tra elementi
- rationale
  - motivazione per la scelta degli elementi e della loro organizzazione



## Architettura software - alcune definizioni

 [SAP/2e]

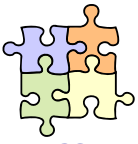
- the **software architecture** of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them

 [IEEE-1471]

- **architecture** – the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution

 [Eoin Woods]

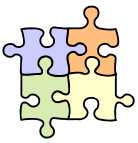
- **software architecture** is the set of design decisions which, if made incorrectly, may cause your project to be cancelled



# Architettura software

 [SSA]

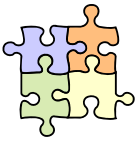
- l'**architettura** di un sistema software-intensive è la struttura o le strutture del sistema, che comprendono elementi software, le proprietà visibili esternamente di questi elementi, e le relazioni tra di essi



# Architettura software: elementi

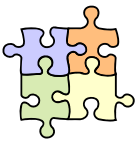
 [SSA]

- l'architettura di un sistema software-intensive è la struttura o le strutture del sistema, che comprendono *elementi software*, le proprietà visibili esternamente di questi elementi, e le relazioni tra di essi



## Elementi software

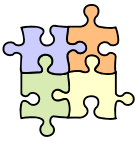
- Un'architettura software comprende/definisce degli **elementi software**
  - un'architettura è un'astrazione del sistema – che mostra gli elementi del sistema e le loro interazioni
  - gli elementi possono essere di varia natura
    - ad es., codice (moduli), processi, basi di dati, unità di deployment, ...
  - è interessata al comportamento *all'interfaccia* degli elementi e alle loro relazioni/collaborazioni *esterne*
  - di solito non è interessata all'implementazione interna degli elementi
    - il comportamento di ciascun elemento è parte dell'architettura – ma solo nella misura in cui quel comportamento può essere osservato dal punto di vista di un altro elemento



## Architettura sw: strutture

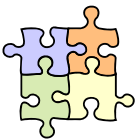
 [SSA]

- l'architettura di un sistema software-intensive è la *struttura o le strutture* del sistema, che comprendono elementi software, le proprietà visibili esternamente di questi elementi, e le relazioni tra di essi



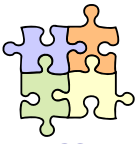
## Strutture del sistema

- [SAP/2e] identifica due nozioni correlate
  - una **struttura** comprende un insieme di *elementi*, nonché le *relazioni* tra questi elementi
    - ad es., un insieme di moduli (codice) – e le relazioni tra moduli
    - ad es., un insieme di processi – e la relativa comunicazione interprocesso
  - una **vista** è la *descrizione* di una struttura – in termini dei suoi elementi e delle relazioni tra di essi
    - ad es., un diagramma UML
  - i termini *struttura* e *vista* sono talvolta usati in modo intercambiabile



## Strutture del sistema

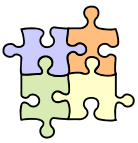
- Un'architettura comprende *una o più strutture*
  - ad es., una struttura statica (moduli) e una dinamica (processi)
  - ciascuna struttura è interessata a/descrive aspetti diversi
  - è interessante comprendere anche le relazioni tra strutture diverse



## Struttura statica

 [SSA]

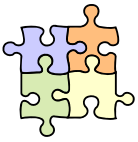
- la **struttura statica** di un sistema software definisce i suoi elementi interni a *design-time* e la loro organizzazione
- Una struttura statica descrive gli *elementi* (statici) del sistema e le loro *relazioni* (statiche)
  - elementi
    - moduli – ad es., classi, package o altre unità di codice
    - elementi per dati – ad es., file e schemi relazionali
    - elementi hardware – ad es., computer, dischi, reti, ...
  - relazioni
    - ad es., usa/richiede, estende, è composto da, ...
  - descritta da un diagramma statico – ad es., un diagramma delle classi di UML



## Struttura dinamica

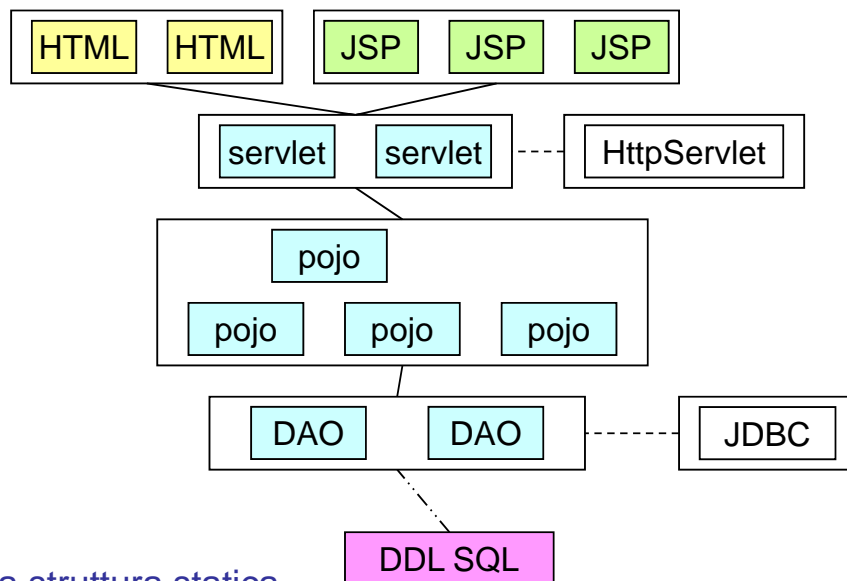
 [SSA]

- la **struttura dinamica** di un sistema software definisce i suoi elementi *runtime* e le loro interazioni
- Una struttura dinamica descrive come funziona il sistema
  - gli *elementi* (dinamici) sono comunemente processi
  - alcune possibili *relazioni* (dinamiche, ovvero interazioni)
    - scambio di messaggi (dati e/o informazioni di controllo) tra elementi
    - esecuzione sequenziale o parallela di task
    - effetto (trasformazione) su dati condivisi
  - descritta da un diagramma dinamico – ad es., un diagramma di interazione di UML

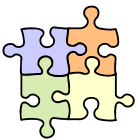


## Esempio

- Un'applicazione web per basi di dati potrebbe essere realizzata scrivendo/utilizzando i seguenti *moduli*

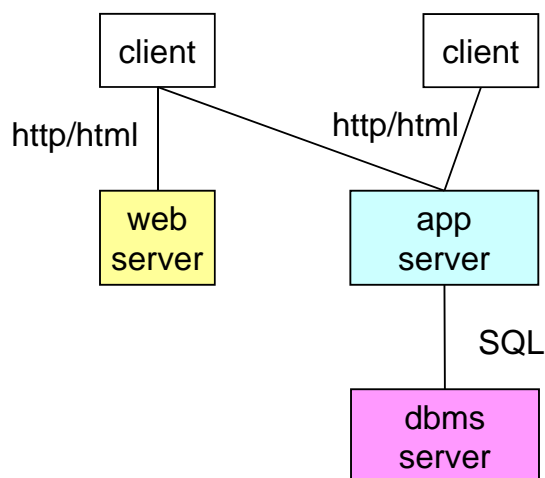


questa è una struttura statica  
gli elementi sono moduli

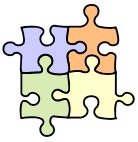


## Esempio

- L'esecuzione dell'applicazione web mediante diversi *processi*, in esecuzione su più *calcolatori*, che comunicano in vari modi

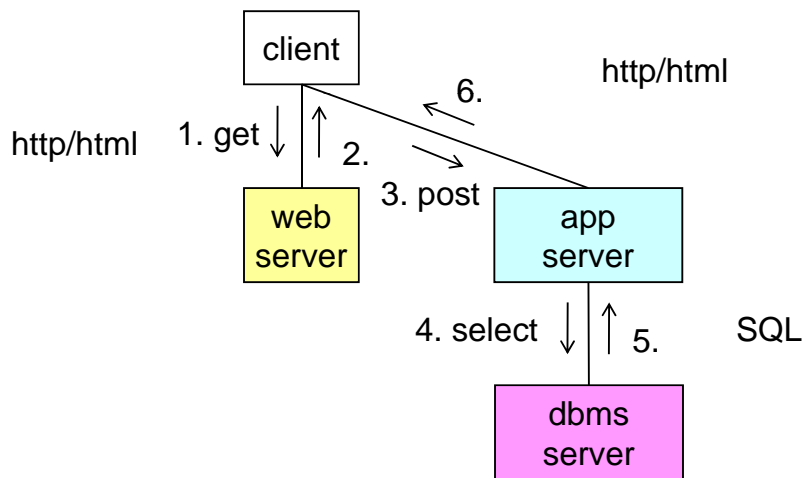


questa è una struttura dinamica, perché gli elementi sono processi  
tuttavia, non si parla ancora della dinamica delle interazioni

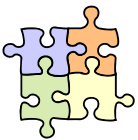


## Esempio

- Un possibile scenario di esecuzione dell'applicazione web
  - come viene gestita una richiesta da parte di un client?



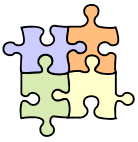
questa è una struttura dinamica  
gli elementi sono processi  
anche i messaggi scambiati sono elementi significativi



## Architettura software: proprietà esterne

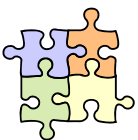
📖 [SSA]

- l'architettura di un sistema software-intensive è la struttura o le strutture del sistema, che comprendono elementi software, le *proprietà visibili esternamente* di questi elementi, e le relazioni tra di essi



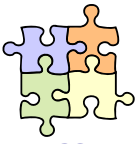
## Proprietà visibili esternamente

- Due tipologie di **proprietà visibili esternamente**
  - *comportamento visibile esternamente*
    - che cosa fa un elemento/il sistema
    - ovvero, che cosa gli posso chiedere? come glielo chiedo?  
come mi risponde?
  - *proprietà di qualità* – o, semplicemente, *qualità*
    - come un elemento/il sistema fa le cose
    - ad esempio, in termini di prestazioni, sicurezza, affidabilità,  
...



## Proprietà visibili esternamente

- Attenzione, possibili due punti di vista
  - proprietà visibili esternamente di un *sistema*
  - proprietà visibili esternamente di un *elemento* interno di un sistema
  - siamo interessati a capire come le proprietà esternamente visibili degli elementi interni contribuiscono alle proprietà esternamente visibili del sistema



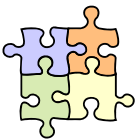
## Proprietà visibili esternamente

 [SSA]

- il **comportamento visibile esternamente** di un sistema software definisce le interazioni funzionali tra il sistema e il suo ambiente
- le funzionalità di un sistema possono essere descritte (a scatola nera) mediante, ad esempio
  - casi d'uso
  - diagrammi di sequenza di sistema e interfaccia del sistema
  - contratti delle operazioni di sistema

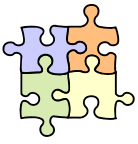
 [SSA]

- una **proprietà di qualità (qualità)** è una proprietà non funzionale visibile esternamente
- ad es., prestazioni, sicurezza, scalabilità, ...

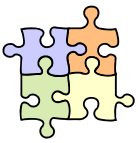


## Esempio

- Un **sistema di prenotazioni aeree** deve consentire un certo numero di tipi di transazioni
  - prenotare un posto, modificare o annullare una prenotazione, ...
- Comportamento visibile esternamente
  - quali richieste possono essere fatte al sistema
  - quali le sue risposte
- Qualità
  - tempo medio di risposta ad una transazione (in certe condizioni di carico), throughput massimo, disponibilità, tempo di ripristino in caso di fallimento, ...

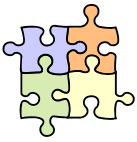


## Diagramma di contesto

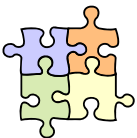
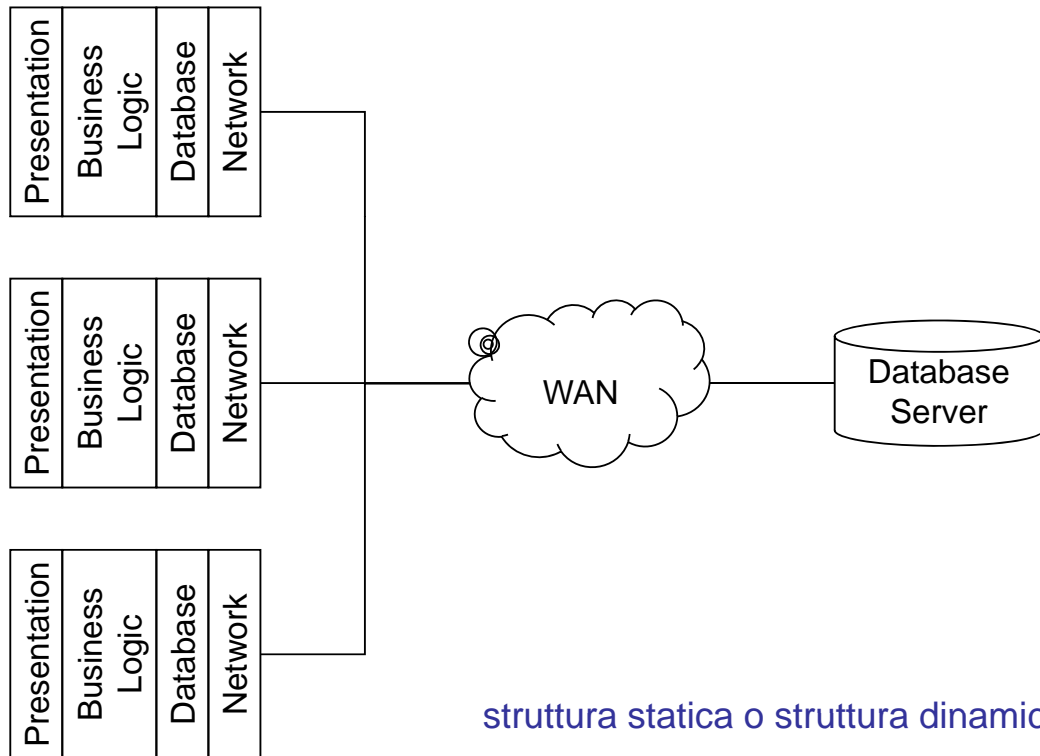


## Possibili realizzazioni

- A fronte di questi requisiti (funzionali e non funzionali), il sistema potrebbe essere organizzato secondo diverse possibili architetture (architetture candidate)
  - ad es., architettura client/server a due livelli oppure a tre livelli

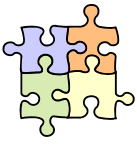


## Architetture client/server a due livelli

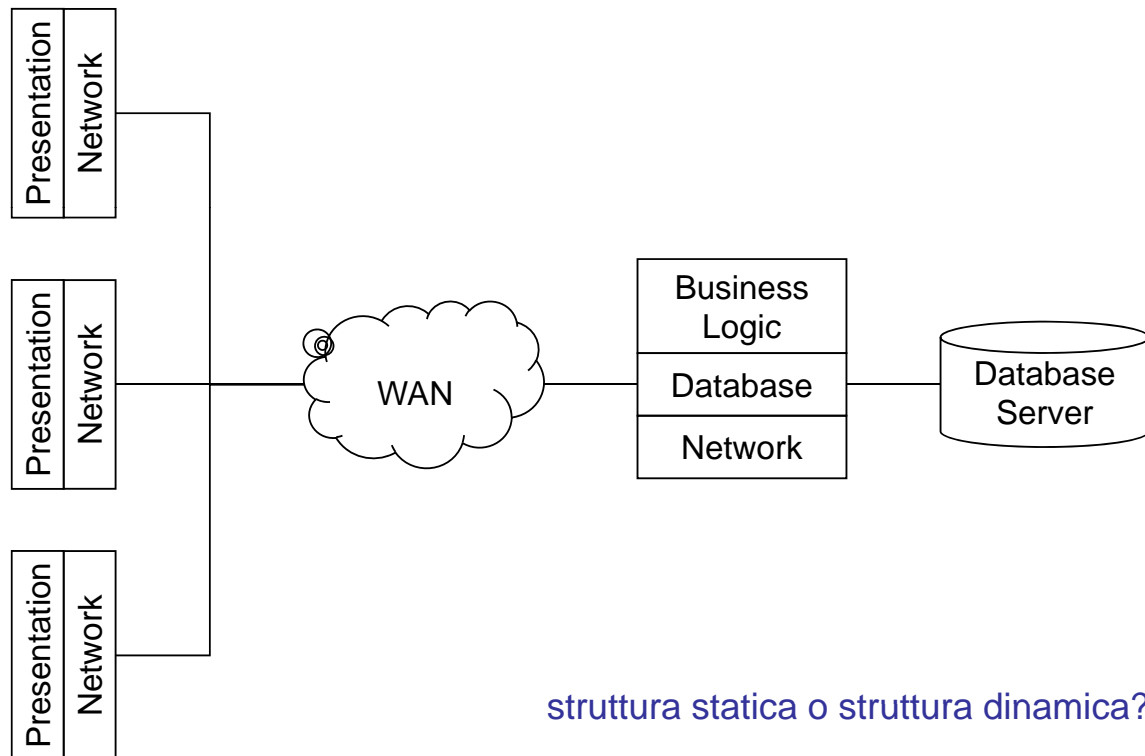


## Architettura client/server a due livelli

- **Client/server** è uno **stile architetturale**
- **Struttura statica**
  - programmi client (thick client) – un'architettura logica a strati, con strati Presentation, Business Logic, Database e Network
  - database server – schema della base di dati
  - connessioni tra di essi
- **Struttura dinamica**
  - modello request/response – il client effettua delle richieste (sulla base di un opportuno formato/protocollo) al database server e riceve delle risposte – tramite la WAN



## Architetture client/server a tre livelli



27

Architetture software: Concetti

Luca Cabibbo – SwA



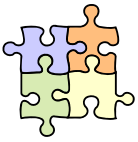
## Architettura client/server a tre livelli

- Una variante – architettura *client/server a tre livelli*
- Struttura statica
  - programmi client (thin client) – un'architettura logica a strati, con strati Presentation e Network
  - programma sul server applicativo – con un'architettura a strati, con gli strati Business Logic, Database e Network
  - database server – ad es., schema della base di dati
  - connessioni tra di essi
- Struttura dinamica
  - modello request/response a tre livelli
    - il client effettua delle richieste all'application server...
    - l'application server effettua delle richieste al database server...

28

Architetture software: Concetti

Luca Cabibbo – SwA

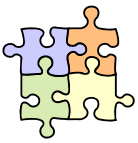


## Qualità

- Entrambe queste architetture candidate *dovrebbero* consentire di soddisfare i requisiti

come facciamo a saperlo?  
come scegliamo tra le due?

- L'architettura C/S a due livelli presenta i seguenti vantaggi
  - più semplice
  - più veloce da realizzare e gestire
  - ...
- L'architettura C/S a tre livelli presenta i seguenti vantaggi
  - migliore scalabilità – se aumenta il carico
  - minori esigenze sull'hardware dei client
  - migliori opzioni di gestione della sicurezza
  - ...

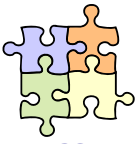


## Architettura candidata



[SSA]

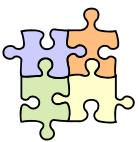
- un'**architettura candidata** per un sistema è un particolare arrangiamento di strutture statiche e dinamiche che ha il potenziale per esibire le proprietà visibili esternamente (funzionali e di qualità) del sistema
- Le architetture candidate per un sistema soddisfano (*dovrebbero soddisfare*) i requisiti (funzionali e non) indicati
  - potrebbero farlo, ma in misura diversa
  - tuttavia, probabilmente differiscono rispetto ad alcune qualità
    - ad es., più facile da mantenere ma più costosa da realizzare
- L'architetto deve essere in grado di
  - identificare le architettura candidate – comprendere le loro qualità – scegliere l'architettura “migliore” per il sistema



# Architettura software: relazioni tra elementi

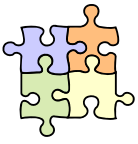
 [SSA]

- l'architettura di un sistema software-intensive è la struttura o le strutture del sistema, che comprendono elementi software, le proprietà visibili esternamente di questi elementi, e le *relazioni tra di essi (gli elementi)*



# Organizzazione interna e proprietà esterne

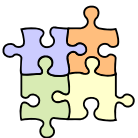
- Relazione tra organizzazione interna e proprietà esterne
  - il comportamento visibile esternamente di un sistema è determinato dal comportamento funzionale combinato dei suoi elementi interni
  - le qualità esterne di un sistema derivano dalle qualità dei suoi elementi interni
    - di solito, una qualità complessiva del sistema è buona come la qualità del suo elemento interno peggiore o più debole
    - è talvolta possibile costruire sistemi di qualità migliore di ciascuno dei suoi elementi interni preso individualmente
  - in un'architettura sono di interesse anche le proprietà (funzionali e di qualità) esterne degli elementi interni
- Come facciamo a comprendere le qualità esterne complessive di un sistema organizzato secondo una certa architettura?



## Ulteriori osservazioni

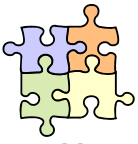
### ❖ *Principio*

- ogni programma o sistema informatico ha un'architettura software – sia se essa è stata documentata e compresa o meno
- l'architettura di un sistema potrebbe essere casuale
- se l'architettura di un sistema non è nota, allora potrebbe essere utile/necessario ricostruirla
- un sistema ha una sola architettura – che però può essere descritta in modi diversi



## Ulteriori osservazioni

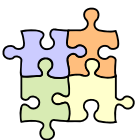
- La definizione di architettura software non distingue tra architetture buone e non buone
  - è importante saper valutare le architetture



## Ulteriori osservazioni

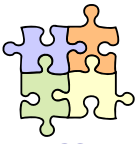
 [IEEE-1471]

- **architecture** – the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution
- Questa definizione enfatizza l'importanza per le architetture
  - dell'ambiente (contesto) del sistema
    - comprende sia l'ambiente di esecuzione, ma anche quello di sviluppo, operativo, politico, ...
  - dei principi che guidano la progettazione e l'evoluzione
    - ad es., lo stile architeturale



## Che cos'è architetturealmente significativo?

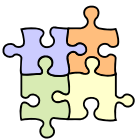
- Le architetture software sono a cavallo tra requisiti e progettazione
  - quali i confini?
- ❖ *Principio*
  - un interesse, un requisito, un problema, un elemento del sistema o una decisione di progetto è **architetturalmente significativo/a** se ha un impatto ampio sulla struttura del sistema o su una sua qualità importante – come prestazioni, scalabilità, sicurezza, ...



## \* Elementi architetturali

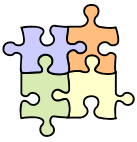
 [SSA]

- un **elemento architetturale** (**elemento**) è un pezzo fondamentale che costituisce un sistema
- gli elementi possono essere di varia natura
  - ad es., codice, librerie, processi, prodotti software riusabili (un DBMS), uno schema di base di dati, intere applicazioni, ...
- Caratteristiche di un elemento architetturale – nella progettazione di un'architettura, per ciascun elemento devono essere definiti/descritti chiaramente
  - un insieme di *responsabilità*
  - *confini*
  - *interfaccia* – definisce i *servizi* che l'elemento fornisce agli altri elementi



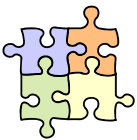
## Esempio

- Si consideri l'applicazione web per basi di dati
  - quali responsabilità, confini e interfaccia per
    - i client?
    - le servlet?
    - i DAO?
    - il DDL SQL?



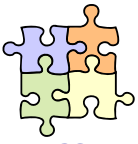
## Elementi e componenti

- [IEEE-1471] parla di “componenti” anziché di “elementi”
  - in UML, un *componente* è “una parte modulare di un progetto di un sistema che nasconde la sua implementazione dietro un insieme di interfacce esterne”
    - definizione abbastanza generale che comprende, ad es., moduli, processi, una base di dati, ...
- Attenzione: in alcune accezioni questi termini sono sinonimi, ma in altre descrivono concetti diversi
  - alcuni termini più specifici comunemente usati nelle architetture
    - *modulo* – un costrutto del linguaggio di programmazione, ad es., una classe o un package
    - *componente* – un elemento runtime – specifico di un modello per componenti – ad es., un enterprise bean



## \* Parti interessate ed interessi

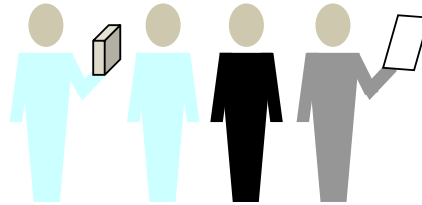
- Lo sviluppo di un sistema software è guidato (in parte) dai requisiti degli utenti
  - ma gli utenti non sono le uniche parti interessate al sistema
  - il sistema non è solo usato, ma è anche costruito, verificato, deve essere riparato e migliorato, deve essere pagato, gestito, monitorato, ...
  - ciascuna di queste attività è svolta da un gruppo di persone – indicato collettivamente come una *parte interessata*



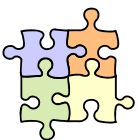
## Parti interessate

 [IEEE-1471]

- una **parte interessata (stakeholder)** in un'architettura software è una persona, un gruppo o un'entità che ha un interesse circa la realizzazione dell'architettura

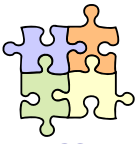


- Le parti interessate sono importanti perché
  - sono le persone che possono dirci quello che serve
  - sono le persone che possono dirci se quello che è stato costruito/progettato è giusto (o meno)



## Possibili parti interessate [SSA, Ch. 9]

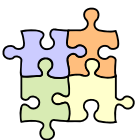
- Possibili parti interessate ad un sistema
  - acquirenti – pagano il sistema
  - valutatori/periti – ne valutano la conformità alle leggi o standard
  - supporto – forniscono supporto agli utenti, quando il sistema è in uso
  - amministratori di sistema – amministrano il sistema
  - utenti – definiscono i requisiti funzionali del sistema e usano il sistema
  - comunicatori – spiegano il sistema alle altre parti interessate
  - sviluppatori – costruiscono e rilasciano il sistema
  - tester – verificano il sistema
  - manutentori – gestiscono l'evoluzione del sistema dopo che è stato rilasciato
  - fornitori – costruiscono o forniscono hardware o software



## Interessi architetturali

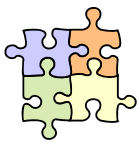
 [SSA]

- un **interesse (concern)** circa un'architettura è un requisito, un obiettivo, un intento o un'aspirazione che una parte interessata ha per l'architettura
- Si tratta di una definizione molto generale
  - gli interessi comprendono i requisiti funzionali e non funzionali
  - un interesse può essere architettralmente significativo anche se (inizialmente) vago

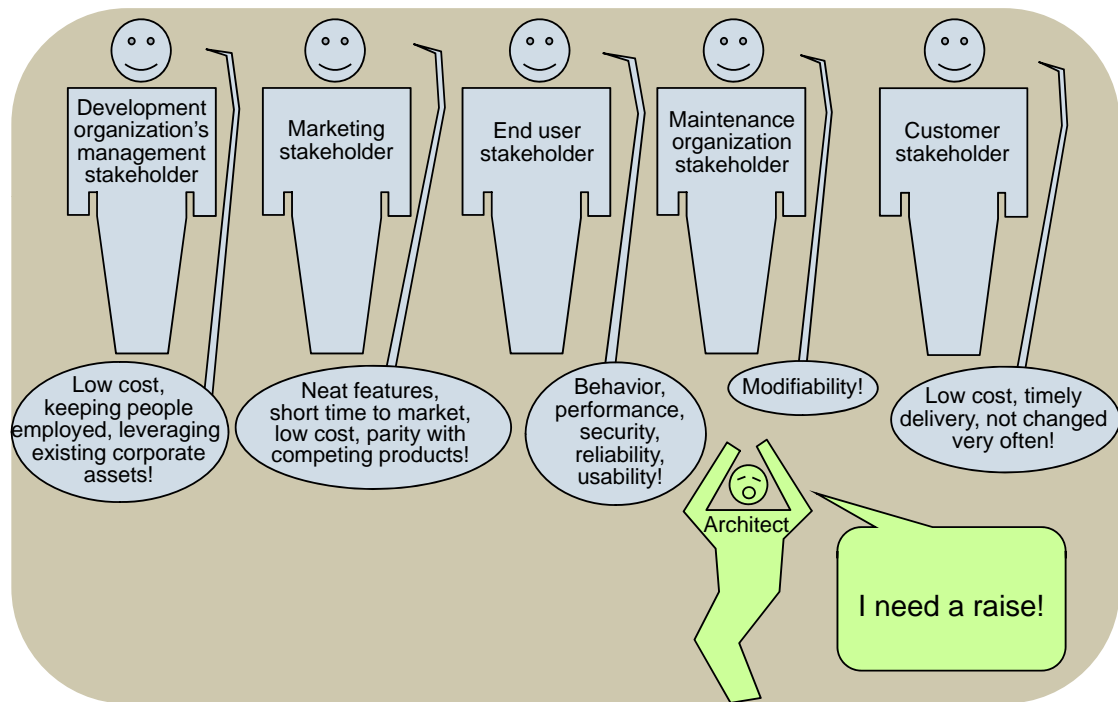


## Parti interessate e interessi

- Le parti interessate possono (devono) guidare la forma e l'organizzazione del sistema
  - per motivi tecnici
  - per motivi legati all'uso del sistema
  - per motivi meramente politici o di potere...
- L'architetto deve saper
  - identificare le parti interessate
  - catturarne gli interessi
  - riconciliare interessi contrastanti
  - saper comunicare le proprie decisioni alle parti interessate



## Parti interessate e interessi



45

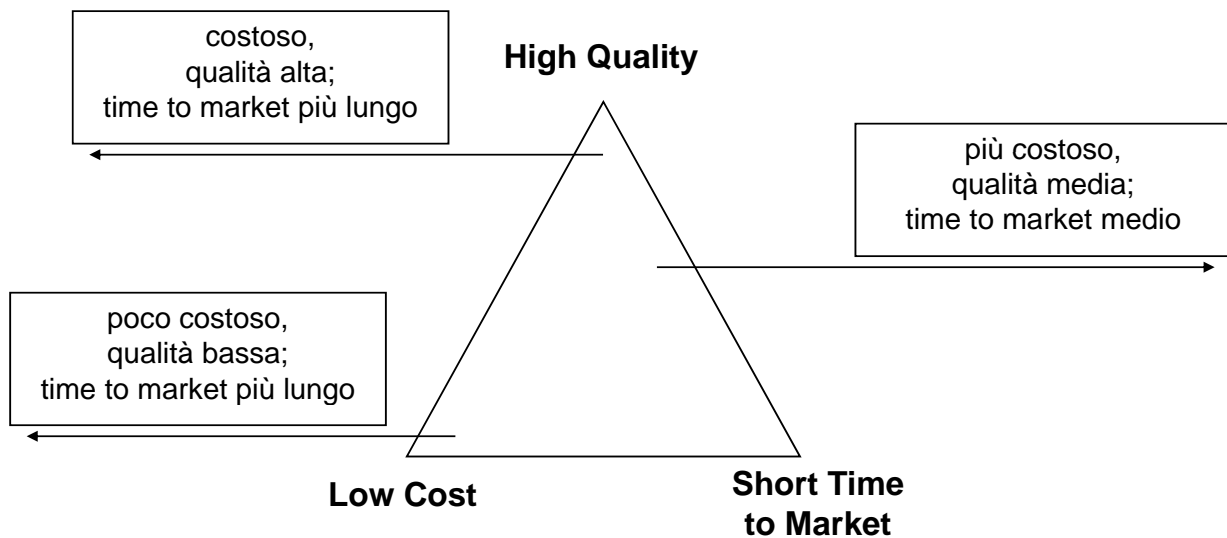
Architetture software: Concetti

Luca Cabibbo - SwA



## Esempio: Interessi e compromessi

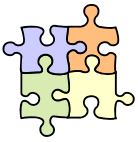
- ▣ Triangolo della qualità
  - mostra tre interessi contrastanti – qualità, costo, time to market



46

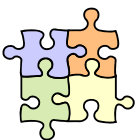
Architetture software: Concetti

Luca Cabibbo - SwA



## Importanza delle parti interessate

- Le parti interessate guidano (direttamente o indirettamente) l'intera forma ed organizzazione dell'architettura
  - l'architettura (il sistema) viene realizzata solo per soddisfare i bisogni delle parti interessate
    - senza parti interessate, non servirebbe sviluppare il sistema
  - sotto la guida dell'architetto
- ❖ *Principio*
  - le architetture vengono create solo per soddisfare i bisogni delle parti interessate
- ❖ *Principio*
  - un buon architetto è uno che coglie con successo gli obiettivi, gli scopi e i bisogni delle sue parti interessate



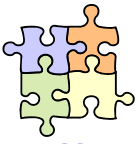
## \* Descrizioni architetture

- L'architettura di un sistema può essere incredibilmente complessa
  - l'architetto deve gestire questa complessità
  - ma deve anche saper descrivere alle parti interessate come essa viene gestita



[IEEE-1471]

- una **descrizione architetture (AD)** è un insieme di prodotti che documentano un'architettura
- Prodotti che costituiscono un'AD
  - modelli architetture (viste)
  - ma anche definizione della portata, vincoli, principi

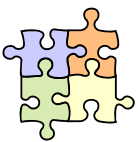


# Descrizioni architeturali



[SSA]

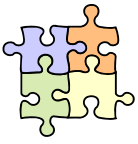
- una **descrizione architeturale (AD)** è un insieme di prodotti che documentano un'architettura
  - in un modo comprensibile dalle parti interessate
  - e che dimostra che l'architettura soddisfa i diversi interessi
- **Attenzione**
- [IEEE-1471] definisce un'AD come “un insieme di prodotti che documentano un'architettura”
  - [SSA] definisce invece una “buona AD”
  - un'AD non buona è un ostacolo – non un beneficio



# Qualità di un'AD

□ **Un'AD**

- deve presentare, allo stesso tempo
  - l'essenza dell'architettura
  - i suoi dettagli
- ovvero
  - deve riassumere l'intero sistema
  - ma deve anche decomporlo in modo sufficientemente dettagliato – affinché l'architettura possa essere validata ed il sistema costruito



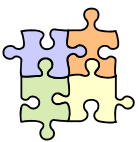
## Osservazioni

### ❖ *Principio*

- anche se ogni sistema ha un'architettura, non tutti i sistemi hanno un'architettura che è comunicata in modo efficace da una descrizione architeturale

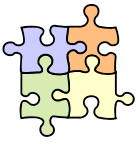
### ❖ *Principio*

- una buona descrizione architeturale comunica in modo efficace e coerente gli aspetti fondamentali dell'architettura alle parti interessate appropriate

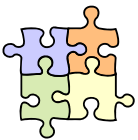
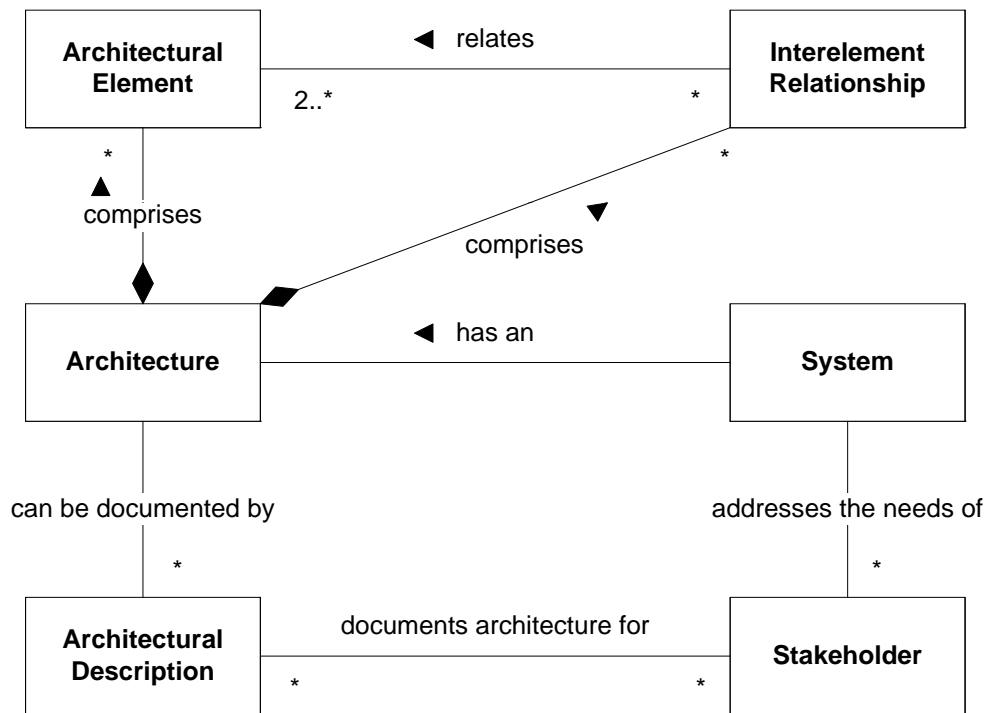


## Esempio

- L'AD per il sistema di prenotazioni aeree è largamente incompleta
  - descrizione della struttura statica
  - non sono descritte le richieste che l'utente può fare
  - le qualità non possono essere valutate in dettaglio
    - ad es., le prestazioni – non è nemmeno specificato il carico atteso
- Le qualità richieste vanno identificate il prima possibile
  - va identificata la parte interessata a ciascuna qualità
  - le qualità vanno risolte a livello architeturale
    - requisiti di qualità (soprattutto se alta) non possono essere aggiunti al sistema in un secondo momento

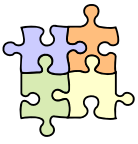


## \* Relazioni tra concetti fondamentali



## \* Discussione

- Vantaggi nel progettare e documentare esplicitamente un'architettura software
  - comunicazione tra/con le parti interessate
  - analisi delle decisioni iniziali di progetto – specialmente dei requisiti non funzionali
  - riuso dell'architettura
    - per sviluppare un sistema “simile” ad uno sviluppato in precedenza
    - pattern e stili architettureali
    - linee di prodotto
    - generatori di codici
    - ...



## Discussione

- Alcune attività legate alle architetture software
  - comprensione dei requisiti
  - creazione o scelta dell'architettura
  - comunicazione dell'architettura
  - analisi e valutazione dell'architettura
  - implementazione del sistema – guidati dall'architettura
  - ricostruzione dell'architettura