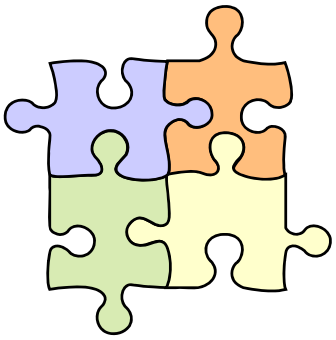
Luca Cabibbo

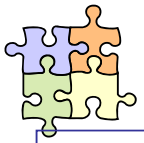


Architetture Software

Punti di vista e viste

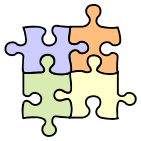
Dispensa AS 3

ottobre 2008



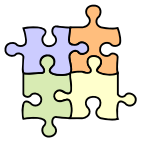
- Fonti

- [SSA] Chapter 3, Viewpoints and Views
- [IEEE-1471] IEEE Recommended Practice for Architectural Description of Software Intensive-Systems
- [SAP/2e] Chapter 2, What is Software Architecture?
- [Kruchten95] Kruchten, Architectural Blueprints – The “4+1” View Model of Software Architecture, IEEE Software, 1995



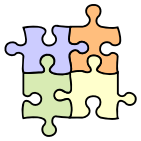
- Obiettivi e argomenti

- Obiettivi
  - introdurre i concetti di punto di vista e vista
- Argomenti
  - introduzione
  - viste architetturali
  - punti di vista
  - modello a 4+1 viste
  - relazioni tra concetti
  - benefici nell'uso di punti di vista e viste
  - rischi legati alle viste
  - un catalogo di punti di vista
  - altri cataloghi di punti di vista



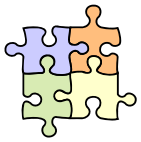
* Introduzione

- L'architettura software
  - descritta da una descrizione architetturale
    - 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
  - vantaggi
    - comunicazione con le parti interessate
    - analisi delle decisioni iniziali di progetto
    - guida allo sviluppo



Descrizione architetturale

- Alcune domande comuni nell'analisi e progettazione di architetture software
  - quali sono i principali elementi funzionali?
  - come interagiscono questi elementi tra loro e con il mondo esterno?
  - come vengono gestite, memorizzate e presentate le informazioni?
  - quali elementi hardware e software sono necessari a supportare questi elementi funzionali e queste informazioni?
  - come vengono allocati gli elementi software a quelli hardware?
  - quali caratteristiche e capacità operative devono essere fornite?
  - quali ambienti di sviluppo, test, supporto e formazione?
  - come organizzare il codice? chi sviluppa che cosa?

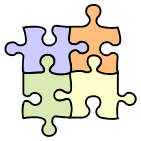


Descrizione architetturale

- Alcune domande comuni nell'analisi e progettazione di architetture software
  - quali son[illegible]incipali elementi funzionali?
  - come intera[illegible]cono [illegible] mondo estern[illegible]
  - c[illegible]
  - fornit[illegible]
  - quali ambienti di sviluppo, test, supporto e formazione?
  - come organizzare il codice? chi sviluppa che cosa?

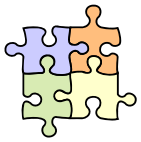
una tentazione a cui resistere:

rispondere a tutte queste domande mediante un singolo modello, sovraccarico, che considera insieme tutti questi aspetti



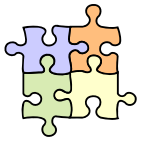
Esempio

- Si considerino nuovamente il sistema di prenotazioni aeree
  - i dati sono distribuiti tra diversi sistemi – in diverse locazioni
  - vanno supportate diverse tipologie di dispositivi di I/O
  - le informazioni vanno presentate in diversi linguaggi
  - stampa dei biglietti – su diverse tipologie di stampanti
  - leggi e regolamenti internazionali
  - ...
- Si immagini un unico modello che comprende tutti gli aspetti
  - un diagramma a box e linee
    - box e linee assumono significati differenti in casi diversi
  - con numerose annotazioni – con semantica eterogenea
  - è difficile – generare un tale modello – scriverlo con un linguaggio possibilmente adatto alle varie parti interessate – mantenerlo aggiornato – ...



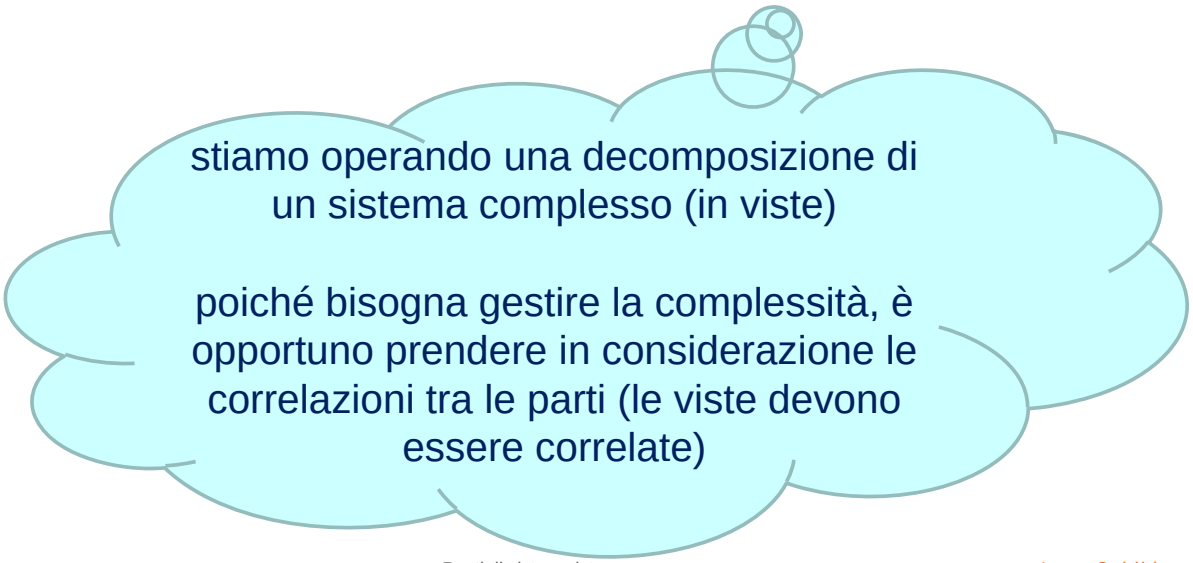
Non usare una singola vista

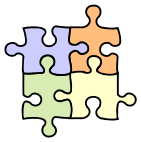
- *Principio*
  - non è possibile cogliere le caratteristiche funzionali e le proprietà di qualità di un sistema complesso mediante un singolo modello, che sia comprensibile e di valore a tutte le parti interessate
- Piuttosto, un'AD deve essere composta da un numero di *viste*, separate ma correlate, ciascuna delle quali descrive un aspetto diverso dell'architettura
  - collettivamente, le viste descrivono l'intero sistema



Uso di viste multiple

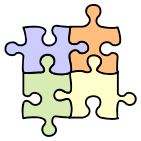
- *Strategia*
  - un sistema complesso viene descritto in modo molto più efficace usando un insieme di viste correlate, che collettivamente illustrano le sue caratteristiche funzionali e proprietà di qualità e dimostrano che raggiunge i suoi obiettivi





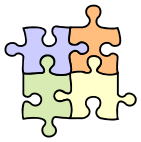
* Viste architetturali

- Una *vista architetturale* descrive quegli aspetti o elementi dell'architettura rilevanti rispetto ad un certo interesse (o insieme di interessi) – che la vista vuole affrontare
  - e pertanto anche per le parti interessate a quell'interesse
- Idea nata negli anni '70 (Parnas)
  - definitivamente accettata con le 4+1 viste di Kruchten (1995) e poi sposata da UP
  - formalizzata da [IEEE-1471]



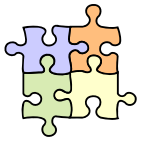
Vista

- [IEEE-1471]
  - una **vista** è una rappresentazione di un intero sistema dal punto di vista di un insieme correlato di interessi
- [SAP/2e]
  - una **vista** è una rappresentazione di un insieme coeso di elementi architetturali, che viene scritta e letta da parti interessate al sistema
- [SSA]
  - una **vista** è una rappresentazione di uno o più aspetti strutturali di un'architettura che illustra come l'architettura affronta uno o più interessi di una o più parti interessate
- Nota
  - ogni vista comprende uno o più modelli/diagrammi



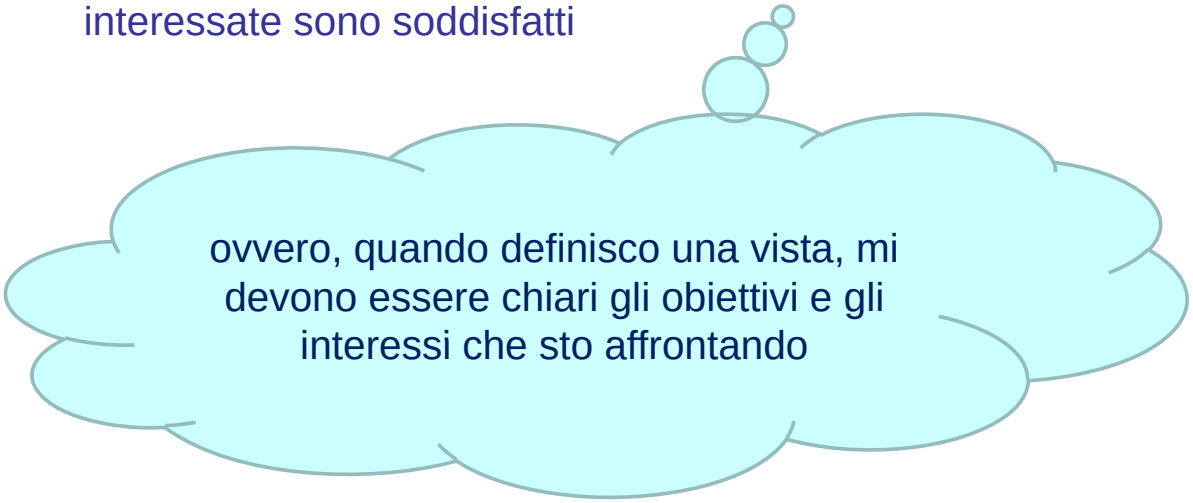
Creazione di una vista

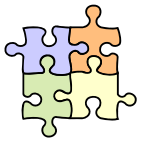
- Per decidere che cosa rappresentare in una vista
  - qual'è l'obiettivo della vista? quali gli interessi da cogliere? a che livello di dettaglio? per quali parti interessate?
  - quali i gruppi di parti interessate a cui è rivolta la vista?
  - quale la comprensione tecnica di tali parti interessate?
  - quali interessi delle parti interessati sono affrontati dalla vista? quale la conoscenza delle parti interessate su tali interessi?
  - quale il dettaglio richiesto per comunicare con le parti interessate non tecniche? quale per validare le qualità dell'architettura?



Creazione di una vista

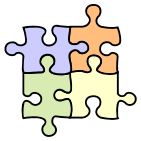
- *Strategia*
  - includi in una vista solo i dettagli che favoriscono gli obiettivi dell'AD
  - ovvero, quei dettagli che aiutano a spiegare l'architettura alle parti interessate o a dimostrare che gli interessi delle parti interessate sono soddisfatti





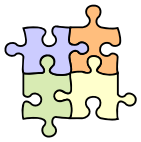
Un problema con le viste

- Problema
  - quali viste creare in un'AD?



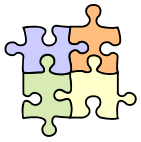
* Punti di vista

- Un *punto di vista* è una tipologia standard di vista che può essere usata in un'AD
  - la scelta delle viste da produrre viene fatta selezionando da un catalogo di punti di vista – anziché decidere quali viste creare sulla base di principi primi
- Si tratta di un'applicazione dei pattern alle AD
  - orientata al riuso di conoscenza relativa alla "soluzione esemplare di problemi significativi e ricorrenti"
- Analogia
  - *punto di vista : vista = classe : oggetto*



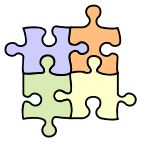
Punti di vista

- [SSA]
  - un **punto di vista** è una collezione di pattern, template e convenzioni per costruire un tipo di vista
  - un punto di vista definisce le parti interessate i cui interessi sono riflessi nel punto di vista stesso – nonché le linee guida, i principi e i modelli per costruire le sue viste
- [IEEE-1471]
  - un **punto di vista** è una specifica di una convenzione per costruire e usare una vista
  - un pattern o template da cui sviluppare viste individuali per determinare lo scopo e i destinatari di una vista – nonché le tecniche per la sua creazione ed analisi



Punti di vista, interessi e parti interessate

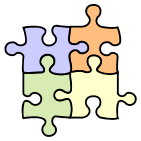
- *Strategia*
  - quando viene sviluppata una vista, sia che si usi un punto di vista ben definito o meno, siano chiari gli interessi che la vista intende affrontare, il tipo di elementi architetturali che contiene, e a chi la vista è rivolta
  - assicurati che anche le parti interessate comprendano ciò



Uno standard per i punti di vista?

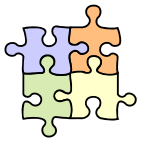
- Un'idea molto attraente – in teoria
  - far riferimento ad un linguaggio standard per la descrizione e l'analisi di AD
  - e magari avere anche un "compilatore" di AD in grado di generare automaticamente lo "scheletro" del sistema
- In pratica
  - nessun "accordo universale"
  - ciascuna AD usa le sue convenzioni

anche se non c'è uno standard universale, alcuni punti di vista (e loro combinazioni) sono diffusi



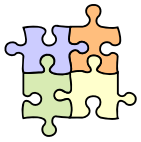
* Modello a 4+1 viste

- P. Kruchten – *Architectural Blueprints – The "4+1" View Model of Software Architecture* – IEEE Software, 1995
- Problema
  - spesso viene usato un solo diagramma per catturare l'architettura di un sistema – inoltre quel diagramma cerca di rappresentare più di quanto è ragionevole fare
  - che cosa rappresentano i rettangoli?
    - codice? processi? computer?
  - che cosa rappresentano le frecce?
    - dipendenze di compilazione? flussi di controllo? flussi di dati?
- Krutchen propone l'uso di più viste concorrenti
  - definisce un insieme di tipi di vista (punti di vista) – su cui basare le viste



Modello a 4+1 viste

- Il modello a **4+1 viste** di Kruchten
  - *vista logica* – modello a oggetti del progetto, ispirato dal dominio, che sostiene i requisiti funzionali
    - classi, package, script, ... – dipendenze
  - *vista dei processi* – coglie gli aspetti di concorrenza e sincronizzazione del progetto
    - processi – comunicazione interprocesso
  - *vista fisica* – descrive le corrispondenze tra software ed hardware, e riflette l'aspetto della distribuzione
  - *vista di sviluppo* – descrive l'organizzazione statica del software nel suo ambiente di sviluppo
    - organizzazione a strati, organizzazione dei file, ...
  - *vista dei casi d'uso* – descrive le funzionalità del sistema – consente di ragionare sulle relazioni tra viste



Il modello a 4+1 viste

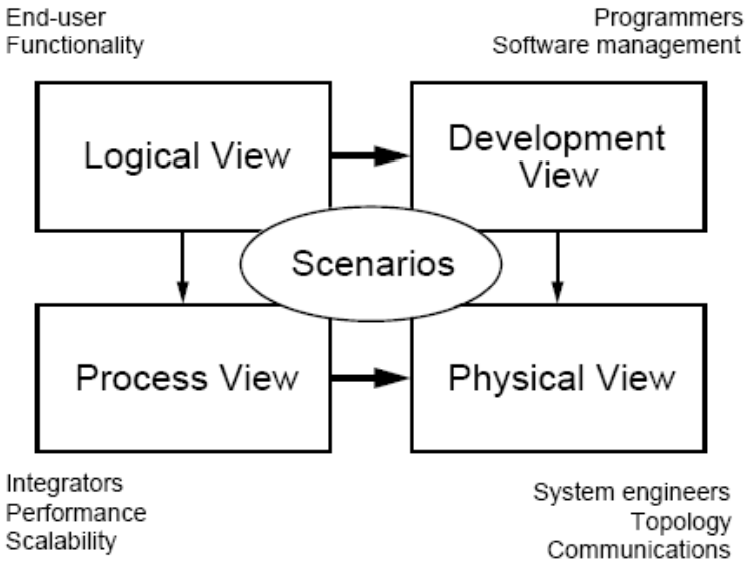
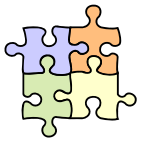
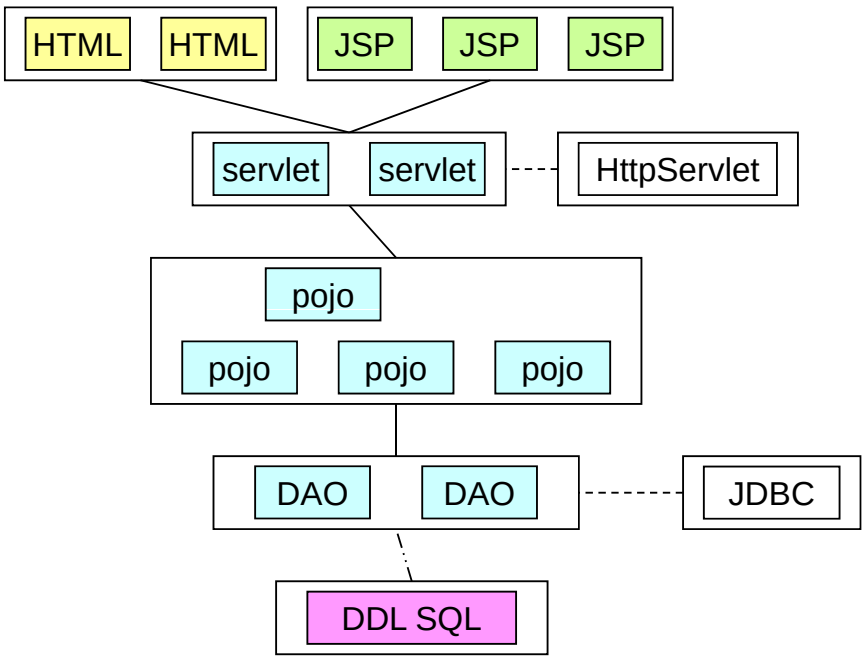
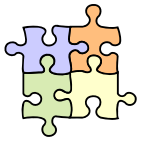


Figure 1 — The "4+1" view model

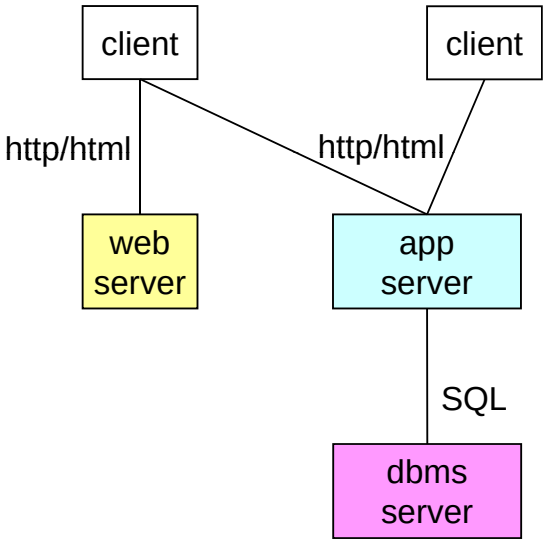


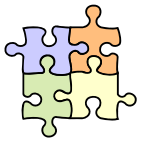
Esempio: vista logica



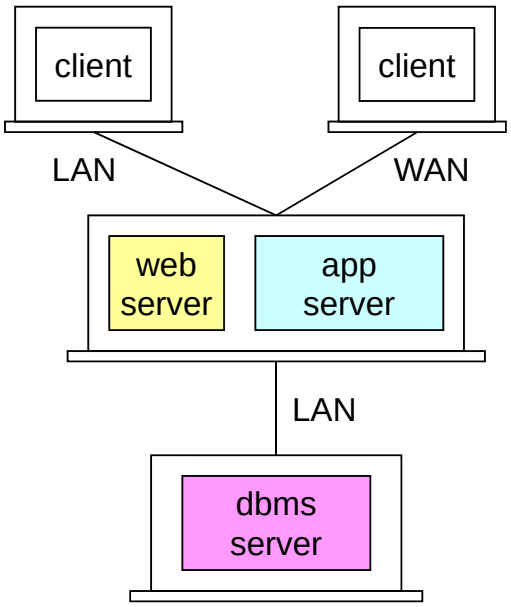


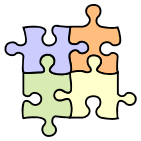
Esempio: vista dei processi





Esempio: vista fisica





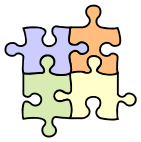
Esempio: vista dei casi d'uso

- Contiene alcuni scenari di casi d'uso significativi
  - per ciascuno scenario – o per alcuni dei suoi passi significativi – illustra l'interazione tra gli elementi architetturali (presenti nelle diverse viste)

- Ad es., che succede quando un utente preme un pulsante per confermare un ordine?
  1. il client invia una richiesta HTTP/POST remota
  2. l'applicazione server
  3. ...
  8. ...

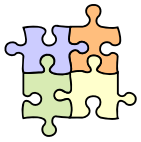
questo "che succede" può essere descritto separatamente, vista per viste

la lettura congiunta permette di comprendere le relazioni tra gli elementi delle varie viste

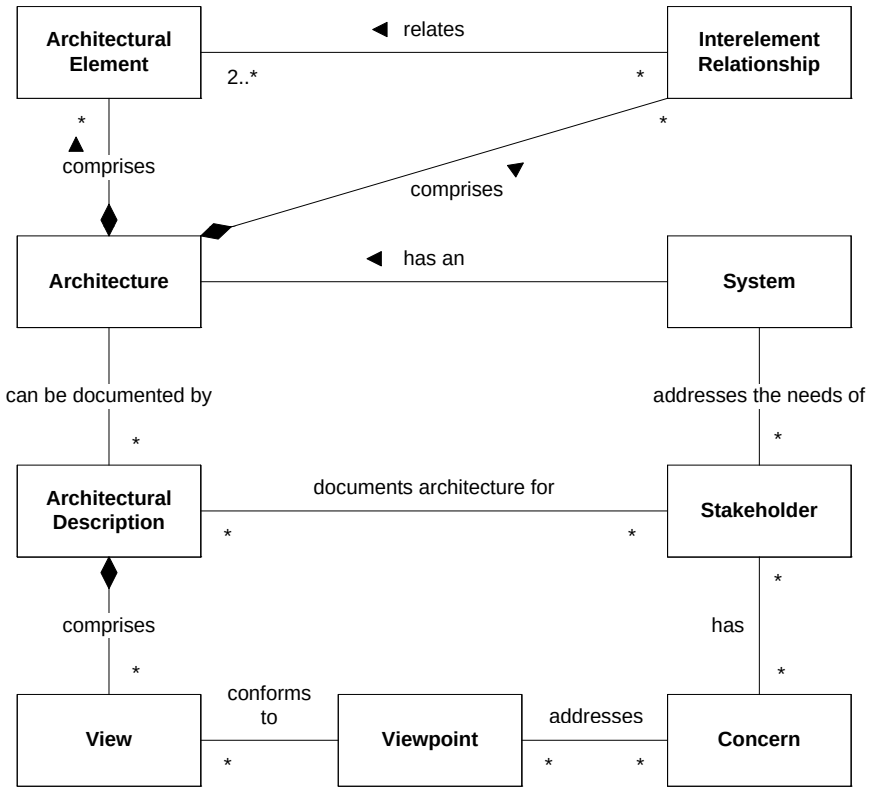


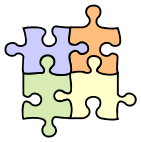
Discussione

- Krutchen suggerisce di
  - effettuare una decomposizione distinta (in elementi e relazioni tra elementi) per ciascuna vista
  - correlare le varie viste/decomposizioni tramite la vista dei casi d'uso
    - la vista dei casi d'uso è basata su *scenari* significativi
    - utile nella progettazione, comunicazione, validazione e miglioramento delle viste
  - adottare uno *stile architetturale* in ciascuna vista
    - la presenza di più viste favorisce la coesistenza di più stili architetturali
  - utilizzare un processo iterativo ed evolutivo



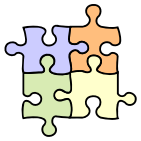
* Relazioni tra concetti fondamentali





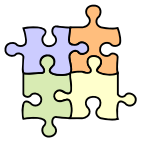
* Benefici nell'uso di punti di vista e viste

- *Separazione degli interessi* – *separation of concerns*
  - principio di progettazione fondamentale
    - separa interessi distinti in aree diverse, in modo che ciascuna abbia uno scopo coeso
    - dividi il progetto in caratteristiche che si sovrappongono il meno possibile
  - modellare un sistema con descrizioni separate ma correlate favorisce i processi di analisi, progettazione e comunicazione, permettendo di concentrarsi separatamente su ciascun aspetto



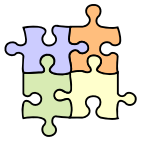
Benefici nell'uso di punti di vista e viste

- Comunicazione con gruppi di parti interessate
  - ciascuna parte interessata è probabilmente interessata solo ad un sottoinsieme dell'AD
- Gestione della complessità
  - gli interessi possono essere gestiti separatamente
- Miglioramento dell'attenzione/focalizzazione dello sviluppatore
  - l'AD contiene non solo modelli per comunicare con gli utenti o l'acquirente, ma anche modelli focalizzati sugli interessi degli sviluppatori



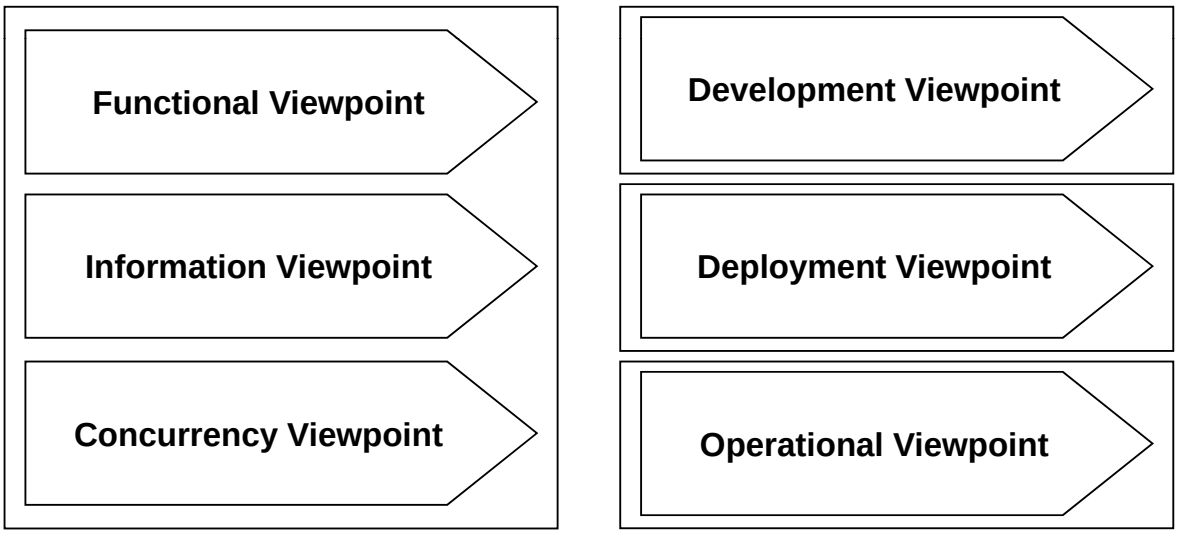
* Rischi legati alle viste

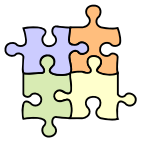
- Alcuni possibili problemi/rischi nell'uso delle viste
  - inconsistenza
    - la verifica della mutua coerenza delle viste è un processo manuale – può essere sostenuto da una vista "+1" e da una checklist di verifiche da fare
  - selezione di un insieme sbagliato di viste
    - non sempre è ovvio capire quali viste usare per descrivere un certo sistema
  - frammentazione
    - uso di un numero eccessivo di viste, che complica la comprensione e l'analisi dell'AD
    - meglio concentrarsi su viste che affrontano solo interessi veramente importanti
    - può talvolta essere accettabile avere viste "ibride"



* Un catalogo di punti di vista

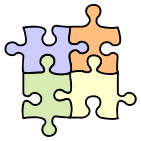
- Diversi cataloghi di punti di vista descritti in letteratura
  - [SSA] propone un catalogo di sei punti di vista





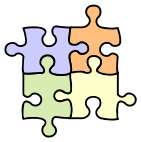
Un catalogo di punti di vista

- Catalogo di SSA
  - i punti di vista Funzionale, Informazioni e Concorrenza caratterizzano l'organizzazione fondamentale del sistema
  - il punto di vista Sviluppo ha lo scopo di sostenere la costruzione del sistema
  - i punti di vista Deployment (rilascio) e Operazionale (operativo, di gestione) caratterizzano l'uso del sistema
- Caratteristiche e uso
  - (abbastanza) indipendenti
  - per descrivere un sistema, alcune viste saranno più importanti di altre – dipende dal sistema!
  - non tutti i sistemi richiedono tutte le viste
  - un sistema descritto solo da alcune viste



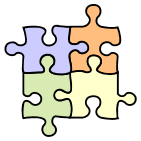
I punti di vista di [SSA]

- Punto di vista *Funzionale*
  - descrive gli elementi funzionali del sistema, le loro responsabilità, interfacce e interazioni principali
    - pilastro di molti AD
    - guida la forma di altre strutture e viste
    - impatto significativo su alcune qualità del sistema – modificabilità, sicurezza, prestazioni, ...
- Punto di vista delle *Informazioni*
  - descrive il modo in cui l'architettura memorizza, manipola, gestisce e distribuisce informazioni – in termini di strutture di dati statiche e di flussi di informazioni
    - importante perché lo scopo dei sistemi informatici è gestire informazioni



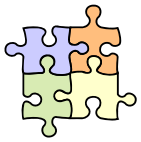
I punti di vista di [SSA]

- Punto di vista della *Concorrenza*
  - descrive l'organizzazione della concorrenza e mappa gli elementi funzionali su unità di concorrenza, nonché le parti concorrenti del sistema e le loro necessità e modalità di sincronizzazione
    - struttura di processi e thread e meccanismi di comunicazione interprocesso
- Punto di vista dello *Sviluppo*
  - descrive l'architettura che supporta il processo di sviluppo
    - ad es., organizzazione del codice, dei moduli, dei test
    - di interesse per chi sviluppa, verifica, mantiene e migliora il sistema – e per chi deve gestire tali persone



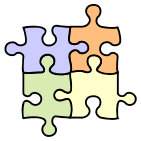
I punti di vista di [SSA]

- Punto di vista del *Deployment*
  - descrive l'ambiente in cui il sistema sarà rilasciato, comprese le dipendenze dall'ambiente runtime
    - elementi hardware (calcolatori, dischi, reti, ...), requisiti per tali elementi, corrispondenze tra elementi software e l'ambiente di esecuzione
- Punto di vista *Operazionale*
  - descrive come il sistema sarà usato, amministrato e supportato quando sarà in esecuzione nell'ambiente di produzione
    - per affrontare interessi relativi alla gestione del sistema – ad es., installazione, aggiornamento, monitoraggio, gestione delle configurazioni, ...



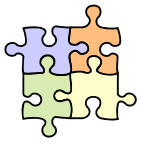
* Altri cataloghi di punti di vista

- Abbiamo già visto il modello a 4+1 viste di Kruchten
  - esso è stato recepito ed esteso da UP



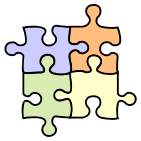
- Viste di RUP

- Viste di RUP – dal Software Architecture Document (SAD)
  - *vista dei casi d'uso*
    - sottoinsieme del modello dei casi d'uso
  - *vista logica*
    - sottoinsieme del modello di progetto
    - package significativi
    - realizzazioni di caso d'uso significative
  - *vista dei processi*
  - *vista di deployment*
  - *vista di implementazione*
  - *vista dei dati*



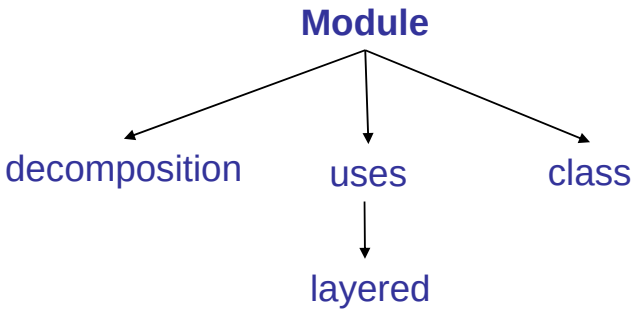
- Viste di [SAP/2e]

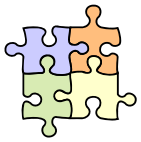
- [SAP/2e] adotta un approccio apparentemente più sintattico a viste e punti di vista – chiamati *viewtype*
  - tre categorie di viewtype
    - viste a moduli
    - viste a componenti e connettori
    - viste di allocazione



Viste a moduli

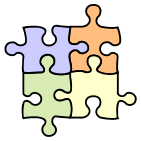
- **Viste a moduli**
  - gli elementi sono moduli – unità di implementazione (codice)
    - ai moduli vengono assegnate aree di responsabilità funzionali
    - meno interesse sul comportamento al tempo di esecuzione
  - relazioni possibili tra moduli – usa, estende, dipende da, strati, ...





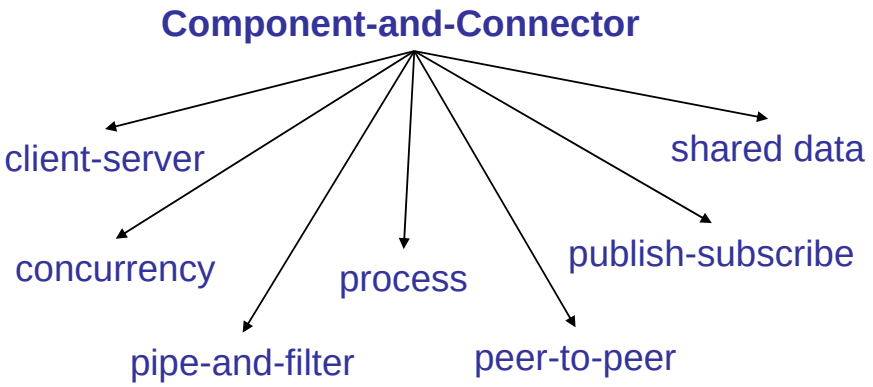
Viste a moduli

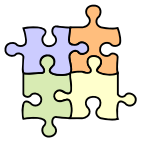
- Per rispondere a domande
  - quale è la responsabilità principale di un modulo?
  - quali moduli possono essere usati da un modulo?
  - quali moduli sono effettivamente usati da un modulo?
  - quali moduli specializzano altri moduli?



Viste a componenti e connettori

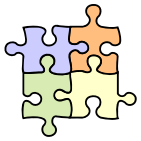
- **Viste a componenti e connettori**
  - gli elementi sono *componenti runtime* (unità di calcolo, processi) e *connettori* (meccanismi di comunicazione tra componenti)
  - punto di vista basato su processi/task/thread
  - relazioni – connessioni tra componenti e connettori – su porte





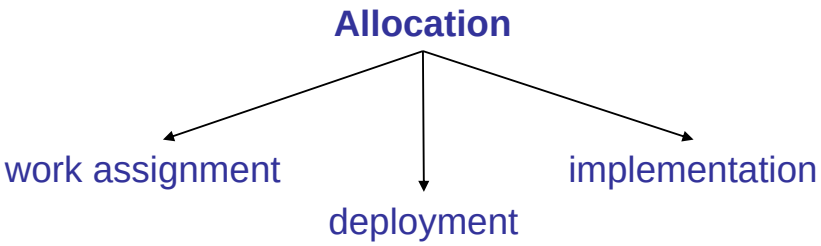
Viste a componenti e connettori

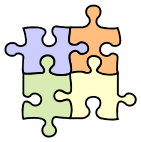
- Per rispondere a domande
  - quali sono le principali componenti in esecuzione? come interagiscono?
  - quali sono i dati condivisi?
  - quali parti del sistema sono replicate?
  - come si muovono i dati nel sistema?
  - quali parti del sistema sono eseguite in parallelo?
  - la struttura del sistema può cambiare durante l'esecuzione?



Viste di allocazione

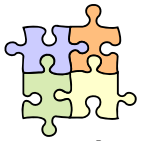
- **Viste di allocazione**
  - gli elementi sono *elementi software* (ad es., moduli) ed *elementi in ambienti esterni* (in cui il software viene sviluppato o eseguito)
  - relazioni – di corrispondenza/allocazione





Viste di allocazione

- Per rispondere a domande
  - su quale processore viene eseguito un componente software?
  - in quali file viene memorizzato un elemento – durante l'implementazione, il test e la costruzione del sistema?
  - quale è l'assegnazione di elementi software a team di sviluppo?



- Relazioni tra cataloghi

- È possibile identificare delle relazioni/corrispondenze tra punti di vista nei diversi cataloghi
  - ad es., la vista Funzionale di [SSA] è un'estensione della vista logica di Kruchten
  - la vista Informazioni di [SSA] corrisponde alla vista dei dati di RUP
  - la vista di Sviluppo di [SSA] può essere descritta con una vista a moduli di [SAP/2e]
- Allo stesso tempo, è difficile stabilire delle corrispondenze esatte tra punti di vista in cataloghi diversi