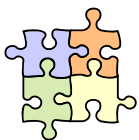


## Requisiti, interessi e scenari

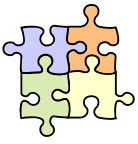
Dispensa ASW 250

ottobre 2011



### - Fonti

- [SAP] Chapter 4, Understanding Quality Attributes
- [SSA] Chapter 8, Scope, Concerns, Principles, and Constraints
- [SSA] Chapter 10, Identifying and Using Scenarios
- [Larman] Applicare UML e i pattern – analisi e progettazione orientata agli oggetti



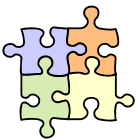
## - Obiettivi e argomenti

### □ Obiettivi

- discutere requisiti architetaturalmente significativi e interessi
- comprendere gli scenari e le loro applicazioni nella definizione dell'architettura

### □ Argomenti

- requisiti
- interessi
- scenari
- applicare gli scenari



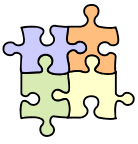
## \* Requisiti

### □ In generale, ogni sistema software ha lo scopo di risolvere un determinato *problema* di un utente

- deve consentire l'esecuzione di un certo numero di *funzionalità*
  - solitamente, relative alla gestione di alcune tipologie di informazioni
- con certe caratteristiche di *qualità*
  - ad es., sicurezza, prestazioni, affidabilità, ...

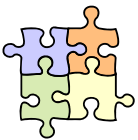
### □ Un *requisito* è una capacità o condizione a cui il sistema deve essere conforme

- una capacità che il sistema deve possedere – ad es., essere capace di gestire certe informazioni o di fornire certe funzionalità
- una condizione che deve essere verificata affinché sia possibile considerare risolto il problema



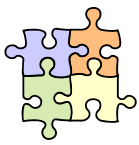
## Gestione dei requisiti

- La **gestione dei requisiti** è il processo di ricerca, analisi, organizzazione, documentazione, verifica e tracciatura dei requisiti (che cambiano)
  - quali sono le parti interessate al sistema? quali i loro interessi e bisogni reali?
  - quali le categorie di requisiti?
  - come possono essere identificati i requisiti?
  - come possono essere organizzati e descritti in modo efficace?
    - ad es., per la successiva progettazione – oppure ai fini della verifica del sistema
    - ad es., per correlare i requisiti alle diverse parti del sistema ed alle relative decisioni di progetto
  - come gestire la loro variazione ed evoluzione?



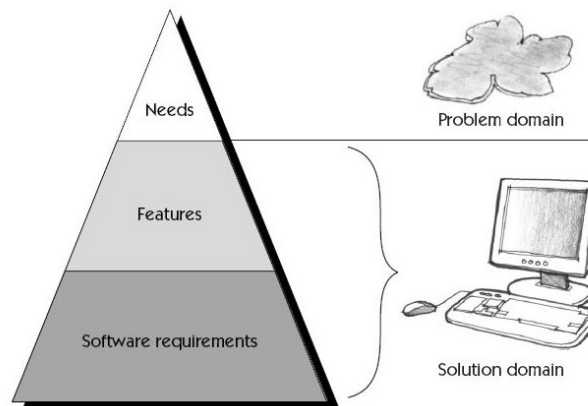
## Requisiti

- Un **requisito** è [Dorfman & Thayer]
  - una capacità del software richiesta dall'utente per risolvere un problema che consente di raggiungere un obiettivo
  - una capacità del software che deve essere soddisfatta o posseduta da un sistema o componente di un sistema per soddisfare un contratto, uno standard, una specifica o quanto imposto da un altro documento formale



## Requisiti

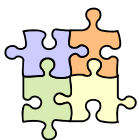
- [Leffingwell & Widrig] collocano i requisiti in una struttura gerarchica
  - la cosa più importante sono i bisogni delle parti interessate
  - un sistema software può soddisfare questi bisogni mediante opportune capacità/features/servizi
  - i requisiti sono una specifica precisa delle capacità/features/servizi che consentono di soddisfare i bisogni delle parti interessate



7

Requisiti, interessi e scenari

Luca Cabibbo - ASw



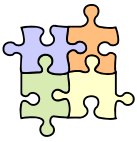
## Gestione dei requisiti

- La **gestione dei requisiti** è [Leffingwell & Widrig]
  - un approccio sistematico all'“elicitazione” (to elicit=far uscire, suscitare, dedurre, ricavare), organizzazione e documentazione dei requisiti del sistema,
  - e un processo che stabilisce e mantiene l'accordo tra il cliente e il team di progetto sui requisiti che cambiano del sistema

8

Requisiti, interessi e scenari

Luca Cabibbo - ASw



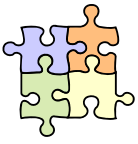
## Alcune possibili definizioni e classificazioni

- Una classificazione relativa al modo in cui i requisiti sono espressi
  - *requisiti utente*
    - in linguaggio naturale – il contesto è la raccolta
  - *requisiti di sistema (specifici)*
    - definisce con precisione che cosa deve essere realizzato – il contesto è la definizione del contratto



## Una classificazione fondamentale

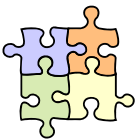
- Una classificazione relativa all'ambito a cui i requisiti si riferiscono
  - *requisiti funzionali*
    - funzionalità che il sistema deve fornire
  - *requisiti non funzionali*
    - non riguardano le specifiche funzioni del sistema, ma piuttosto proprietà (qualità) del sistema nel suo complesso
    - relativi ad interessi come sicurezza, prestazioni, usabilità, ...
    - i requisiti non funzionali sono anche chiamati *qualità*, *attributi di qualità* o *proprietà di qualità*



## Qualità

### □ Da Wikipedia (ottobre 2011)

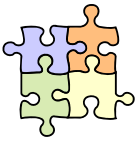
- Questa voce riguarda la qualità nell'accezione più usata, essenzialmente nell'ambito dell'ingegneria, dell'economia e della produzione, quando ci si riferisce ad un bene, materiale o immateriale, che viene prodotto per un determinato utilizzo.
- In generale, la misura della **qualità** indica *una misura delle caratteristiche o delle proprietà di una entità* (una persona, un prodotto, un processo, un progetto) *in confronto a quanto ci si attende da tale entità, per un determinato impiego.*
- L'uso che si intende fare è importante, poiché la valutazione della qualità varia a seconda dell'utilizzo. Per esempio, una persona può essere un ottimo scrittore, ma avere una valutazione molto bassa come atleta. Allo stesso modo, un gruppo di dati può avere un'alta qualità quando usati come informazione generica, divulgativa, ma una bassa qualità per un utilizzo di alta precisione.
- Il concetto di qualità è applicabile [...] ogni volta che un oggetto, una persona o altro, viene confrontato con quello che ci si attende da lui.



## Qualità

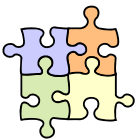
### □ Diversi schemi di classificazione delle qualità

- qualità **esterne** (ad es., prestazioni, usabilità, ...) e **interne** (ad es., modificabilità, integrità concettuale, ...)
- qualità **del prodotto** (ad es., affidabilità, sicurezza, ...) e qualità **del processo** (ad es., uso di un processo agile o burocratico)
  - le qualità del processo si riflettono in quelle del prodotto
  - ad es., un processo che prevede un'attività accurata di testing dovrebbe portare ad un'affidabilità migliore



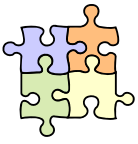
## Principali qualità del software

- Principali qualità del software
  - correttezza – soddisfazione delle specifiche
    - una qualità assoluta (si/no)
    - non sempre ciò che si vuole – specifiche errate?
  - affidabilità – probabilità che non si verifichino malfunzionamenti
    - una qualità relativa
    - bisogna anche tenere conto della gravità dei malfunzionamenti che si possono verificare
  - disponibilità e resilienza – capacità del sistema di essere completamente o parzialmente funzionante
    - anche a fronte di guasti di componenti del sistema
  - prestazioni e scalabilità
  - sicurezza



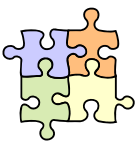
## Altre qualità del software

- Altre qualità del software
  - usabilità
  - costruibilità
  - verificabilità
  - manutenibilità (sia riparabilità che evolvibilità)
  - riusabilità
  - portabilità
  - comprensibilità
  - interoperabilità
  - tempestività
  - visibilità (trasparenza del processo)
  - ...



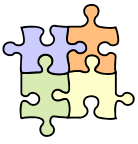
## Altre qualità del software

- Qualità di business
  - time to market
  - costi e benefici
  - vita attesa del sistema
  - programma dei rilasci
  - integrazione con sistemi legacy
  
- Altre qualità architettoniche
  - integrità concettuale



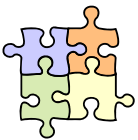
## Funzionalità e qualità

- Funzionalità e qualità sono caratteristiche ortogonali
  - se ciò non fosse, la scelta delle funzioni avrebbe (ad es.) implicazioni sul livello di affidabilità o sulle prestazioni
  
- In generale (in teoria) è possibile scegliere indipendentemente le funzionalità, nonché un livello desiderato di ciascuna qualità
  - in pratica, non tutti i livelli di qualità sono raggiungibili con tutte le funzioni – ad es., elaborazioni complesse di immagini e prestazioni
  - non tutti i livelli di qualità sono sempre raggiungibili – ad es., sistema con elevate prestazioni, facilmente modificabile, poco costoso e prodotto in poco tempo
  
- L'architettura determina il livello relativo di ciascuna qualità



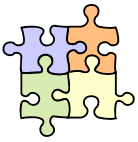
## \* Interessi

- L'architettura software è più interessata agli interessi di un sistema – che non a tutti i suoi requisiti
  - un *interesse* su un'architettura è un requisito, un obiettivo, un'intenzione o un'aspirazione che una parte interessata ha per quell'architettura [SSA]
    - si tratta di una caratterizzazione piuttosto ampia – gli interessi sono relativi agli obiettivi che le varie parti interessate al sistema vogliono perseguire
  - viceversa, un *requisito* è un interesse specifico, non ambiguo e misurabile [SSA]



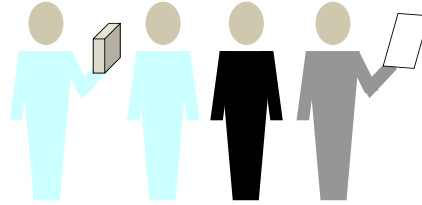
## Architettura e requisiti

- Due problemi (per l'architettura) con i requisiti
  - la forma di un'architettura deriva principalmente dagli attributi di qualità – e non solo da quelli funzionali
    - spesso invece la lavorazione dei requisiti si concentra soprattutto sugli aspetti funzionali – i requisiti di qualità sono talvolta ignorati o spesso specificati in modo inadeguato
  - talvolta, quello che è architeturalmente significativo (ovvero, veramente utile all'architettura) non compare nemmeno nei migliori requisiti
    - molte forze che guidano l'architettura non sono descritti da requisiti – appartengono piuttosto alla categoria degli interessi delle parti interessate
- Per questo, l'architettura si concentra sugli *interessi* e non sui *requisiti*

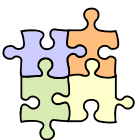


## Parti interessate

- 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 [IEEE-1471]

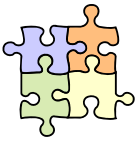


- Le parti interessate sono importanti perché
  - sono le persone che possono dirci quello che realmente serve
  - i loro interessi guidano (direttamente o indirettamente) l'intera forma ed organizzazione dell'architettura
  - sono le persone che possono dirci se quello che è stato costruito/progettato è giusto (o meno)



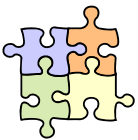
## Possibili parti interessate

- 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 – usano il sistema – possono definire i requisiti funzionali del 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



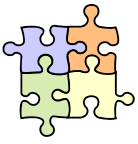
## I requisiti derivano dagli interessi

- I requisiti – soprattutto quelli di qualità – sono (dovrebbero essere) sempre motivati da un qualche obiettivo più ampio, che può essere descritto in termini di valore aggiunto
  - ad es., un certo sistema deve avere un tempo di risposta molto veloce – perché?
    - possibili risposte – per differenziare il prodotto da quello della competizione, e quindi per acquisire una maggiore quota di mercato – oppure per sostenere la sicurezza di un soldato in battaglia
  - è pertanto opportuno identificare e focalizzarsi sui veri obiettivi di un sistema – e in generale sulle forze che ne determinano la forma – in modo tale da poter sostenere questi veri obiettivi



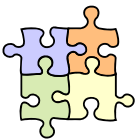
## Interessi

- Gli interessi per un sistema possono essere descritti a vari livelli di dettaglio/astrazione – può essere utile esprimere anche le loro correlazioni tra i diversi livelli di dettaglio/astrazione – ad es., con una struttura gerarchica, ad albero
  - un *business driver* è una forza che agisce su un'organizzazione, e richiede che essa si comporti in un certo modo
  - un *business goal* è un obiettivo specifico di un'organizzazione
  - un *interesse* è un requisito, un obiettivo, un'intenzione o un'aspirazione che una parte interessata ha su un sistema/architettura
  - un *requisito* è un interesse specifico, non ambiguo e misurabile
  - un interesse o requisito è *architetturalmente significativo* se ha un impatto ampio sulla struttura del sistema o su una sua qualità importante – come prestazioni, scalabilità, sicurezza



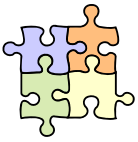
## Interessi

- Si consideri un'azienda che attualmente vende i suoi prodotti in modo tradizionale
  - l'azienda vuole iniziare a vendere i suoi prodotti mediante un servizio di commercio elettronico
  - business driver
    - bisogno competitivo dovuto al fatto che i principali concorrenti offrono, con successo, un servizio di questo tipo
  - business goal
    - nell'arco di un anno, il 15% delle vendite dovranno essere effettuate online
  - interessi
    - sito di commercio elettronico facilmente fruibile
    - l'esperienza di acquisto deve in qualche modo ricordare quella del negozio, per suggerire che si tratta di un modo alternativo per acquistare dalla stessa azienda



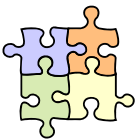
## Dagli interessi ai requisiti

- In ogni caso, la lavorazione dell'architettura opera su interessi espressi sotto forma di requisiti architetturealmente significativi – in modo specifico, non ambiguo e misurabile
  - un esempio di interesse
    - sito di commercio elettronico facilmente fruibile
  - una descrizione un po' più precisa – ma ancora troppo vaga
    - un sito di commercio elettronico deve essere facile da usare da clienti che hanno esperienza limitata con i calcolatori e con il commercio elettronico
  - una descrizione più dettagliata – ma non misurabile/ verificabile
    - la registrazione sul sito deve essere veloce e richiedere un data entry minimo
  - una descrizione ancora più dettagliata e verificabile
    - la registrazione deve richiedere meno passi e l'immissione di un numero minore di dati rispetto ai siti concorrenti

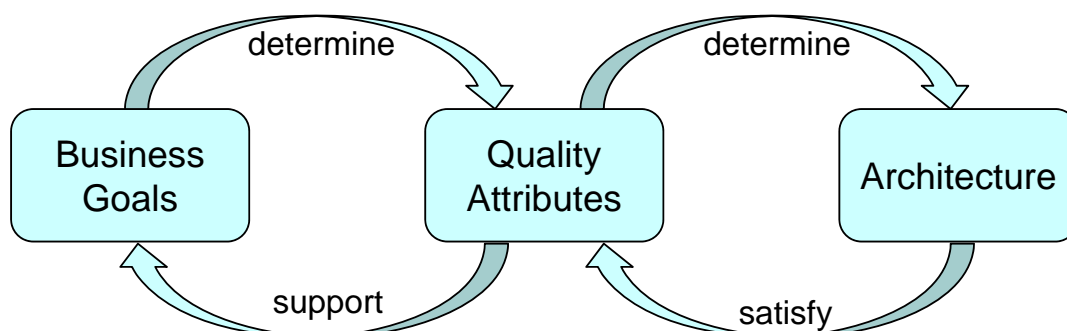


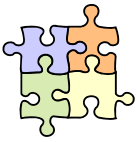
## Business goal

- La sorgente principale dei requisiti architetturealmente significativi – è costituita dall'insieme dei business goal che portano allo sviluppo del sistema
  - alcune categorie di business goal
    - mantenere la crescita e la continuità dell'organizzazione o azienda
    - perseguire obiettivi finanziari
    - perseguire obiettivi relativi agli impiegati
    - perseguire obiettivi per la società, lo stato, o la natura
    - perseguire obiettivi relativi agli azionisti
    - migliorare i processi di business
    - gestire la qualità dei prodotti e la reputazione
    - ...



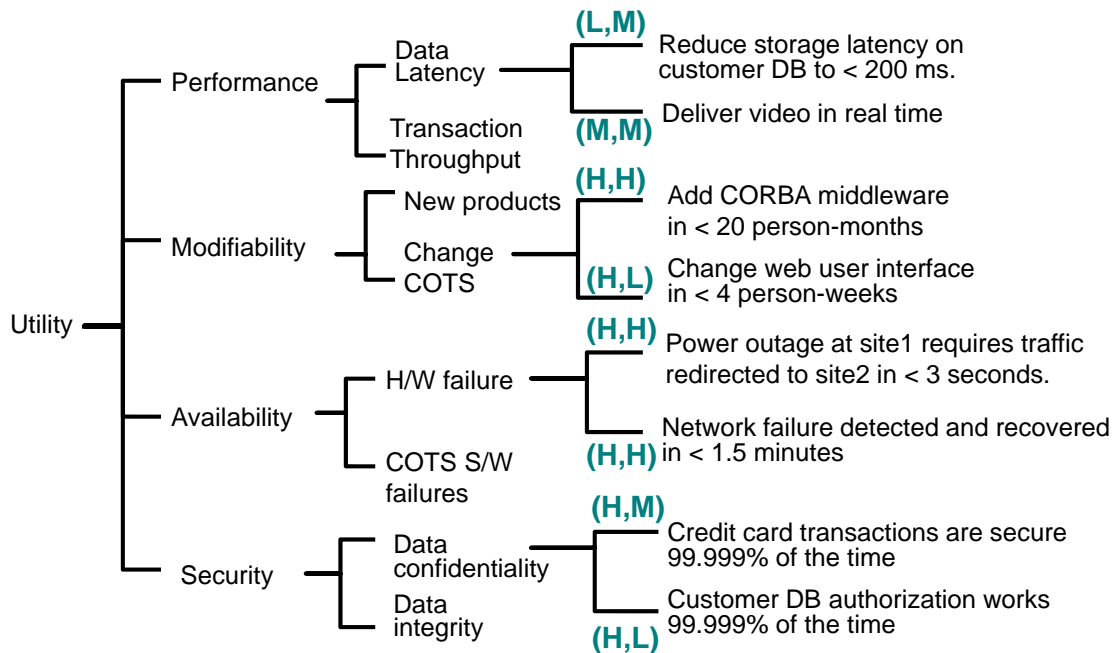
## Riassumendo





## Albero di utilità

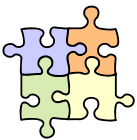
- E' utile poter descrivere le relazioni tra le qualità critiche per un sistema e la loro rappresentazione in termini di requisiti/scenari – ad es., mediante un *albero di utilità* [SAP]



27

Requisiti, interessi e scenari

Luca Cabibbo – ASw



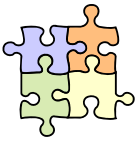
## \* Scenari

- L'*applicazione degli scenari* alle architetture è una delle tecniche più potenti per sviluppare e validare un'architettura
  - gli scenari sono un modo utile per descrivere requisiti architetaturalmente significativi
  - ma anche e soprattutto per progettare/ comprendere/ descrivere/ comunicare/ verificare come l'architettura funziona in pratica
  - l'applicazione degli scenari è una generalizzazione delle idee sottostanti la vista +1 del modello a 4+1 viste di Kruchten
  - due tipi di scenari: scenari funzionali e scenari di qualità

28

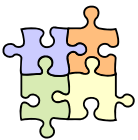
Requisiti, interessi e scenari

Luca Cabibbo – ASw



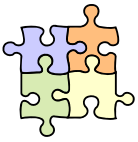
## Scenari e architetture

- Possibili applicazioni degli scenari nelle architetture
  - descrizione di requisiti
  - sviluppo (progettazione) guidato da scenari
  - gli scenari consentono di correlare viste multiple – ovvero di ragionare in modo coordinato su più viste
  - pianificazione iterativa basata su scenari
  - anche validazione e verifica (testing) sono solitamente basate su scenari



## - Richiami sullo sviluppo guidato dai casi d'uso

- Prima di introdurre gli scenari di qualità, ricordiamo che cosa sono e come vengono lavorati gli scenari di casi d'uso nello Unified Process
  - per i dettagli si faccia riferimento al corso di Analisi e progettazione del software – oppure direttamente a [Larman, Applicare UML e i pattern]

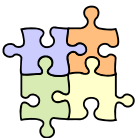


## - Richiamo: Casi d'uso

- I **requisiti funzionali** di un sistema possono essere espressi sotto forma di casi d'uso
  - un **caso d'uso** è una storia scritta che racconta come gli utenti (attori) possono raggiungere un loro obiettivo usando il sistema

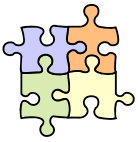
### **Elabora Vendita:**

Un cliente arriva alla cassa con alcuni articoli da acquistare. Il cassiere usa il sistema POS per registrare ogni articolo acquistato. Il sistema mostra il totale e i dettagli per ogni articolo. Il cliente inserisce informazioni sul pagamento, che il sistema convalida e registra. Il sistema aggiorna l'inventario. Il cliente riceve dal sistema una ricevuta e poi se ne va con gli articoli acquistati.



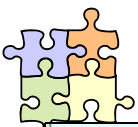
## Attori e scenari

- Un **attore** è qualcosa o qualcuno dotato di comportamento e che interagisce con il sistema
  - persone con un ruolo, sistemi informatici, organizzazioni
- Uno **scenario di caso d'uso** è una sequenza specifica di azioni e interazioni tra il sistema e alcuni attori
  - descrive una particolare storia nell'uso del sistema
    - tipi di azioni/interazioni
      - interazioni tra attori e Sistema,
      - cambiamenti di stato e validazioni (effettuate dal Sistema)
    - descrive quello che fa il Sistema a fronte di interazioni con gli attori
  - uno scenario può essere di **successo** o di **fallimento**



## Casi d'uso e scenari

- Un **caso d'uso** è un insieme di scenari correlati, di successo e fallimento, che descrivono un attore che usa un sistema per raggiungere un obiettivo (un risultato osservabile e di valore)



## Esempio (in formato informale)

### **Gestisci Restituzione**

#### *Scenario principale di successo:*

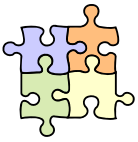
Un cliente arriva alla cassa con alcuni articoli da restituire. Il cassiere usa il sistema POS per registrare ciascun articolo restituito ...

#### *Scenari alternativi:*

Se il cliente aveva pagato con la carta di credito, e l'operazione di rimborso sulla carta di credito è stata respinta, allora il cassiere informa il cliente e lo rimborsa in contanti.

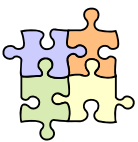
Se il codice identificativo dell'articolo non viene trovato nel sistema, il sistema avvisa il cassiere e suggerisce l'inserimento manuale del codice (può darsi che sia danneggiato).

Se il sistema rileva un fallimento nella comunicazione con il sistema esterno di gestione della contabilità, ...



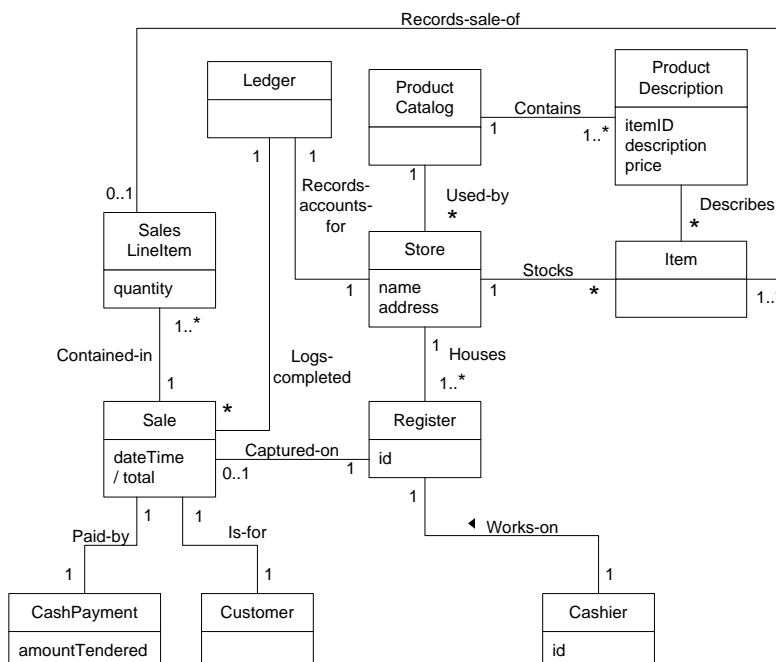
## - Richiamo: Sviluppo guidato dai casi d'uso

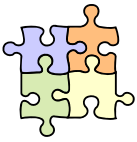
- Un processo software (ovvero, alcune attività nello sviluppo del software) può essere guidato dai casi d'uso – ad es., UP
  - **requisiti** identificati e scritti come casi d'uso
    - quali le parti interessate? quali gli attori? quali gli obiettivi degli attori che useranno il sistema?
  - **progettazione orientata agli oggetti** guidata dai casi d'uso – **realizzazioni di casi d'uso**
    - che cosa succede durante l'esecuzione di uno scenario? quali oggetti sono coinvolti? come collaborano i vari oggetti?
    - obiettivo è progettare una decomposizione ad oggetti della logica applicativa
      - identificare gli oggetti, le loro responsabilità ed interfacce, e le interazioni tra gli oggetti
      - l'identificazione degli oggetti si può ispirare al modello di dominio



## Esempio di sviluppo guidato dai casi d'uso

- Modello di dominio
  - un modello concettuale delle informazioni di interesse

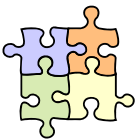
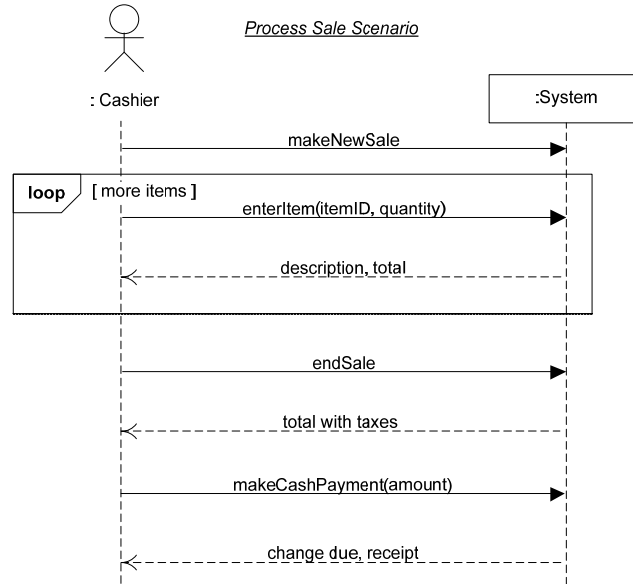




# Esempio di sviluppo guidato dai casi d'uso

## ▣ Diagramma di sequenza di sistema

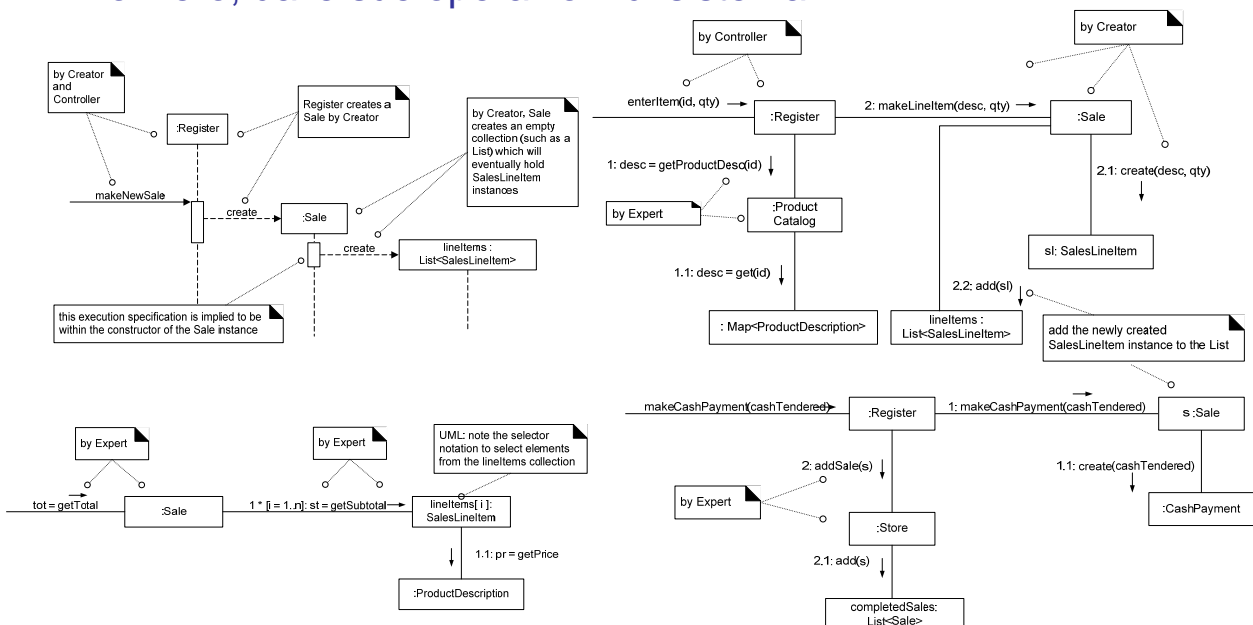
- identifica le operazioni di sistema relative ad uno scenario di caso d'uso – sono le operazioni per cui progettare

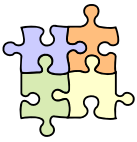


# Esempio di sviluppo guidato dai casi d'uso

## ▣ Realizzazione di caso d'uso

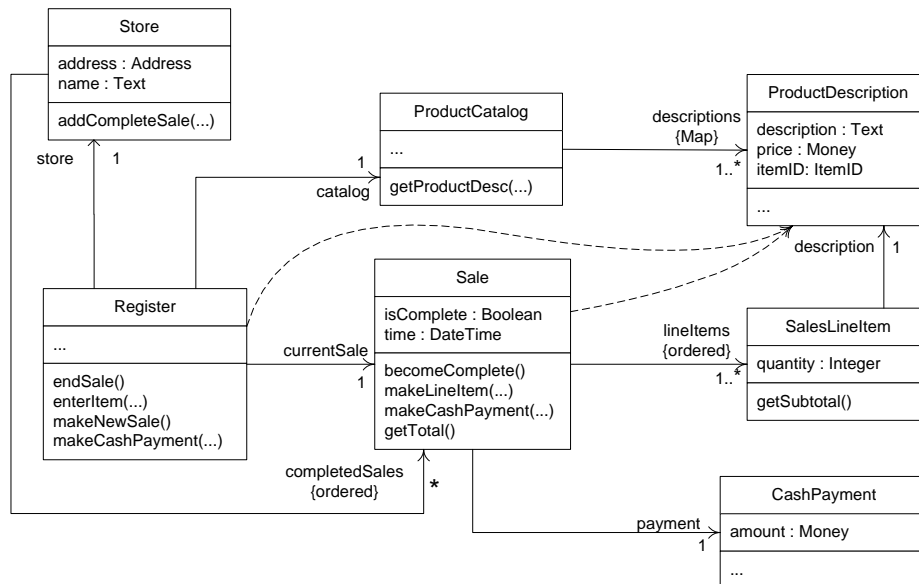
- le interazioni (oggetti e scambio di messaggi) che consentono di soddisfare le responsabilità implicate da un caso d'uso – ovvero, dalle sue operazioni di sistema





## Esempio di sviluppo guidato dai casi d'uso

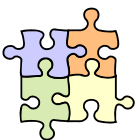
- Diagramma delle classi di progetto
  - rappresentazione statica delle scelte di progetto – classi e loro interfacce



39

Requisiti, interessi e scenari

Luca Cabibbo – ASw



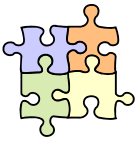
## - Sviluppo dell'arch. sw guidata da scenari

- Anche la **definizione/progettazione dell'architettura** può essere guidata dagli scenari – come vedremo, sia da scenari funzionali che da scenari di qualità
  - quali sono gli scenari rilevanti?
  - che succede durante l'esecuzione di uno scenario? quali elementi/componenti/sottosistemi sono coinvolti? come collaborano i vari elementi?
  - l'obiettivo è progettare una decomposizione in elementi dell'intero sistema
    - identificare gli elementi, le loro responsabilità ed interfacce, e le interazioni tra gli elementi
  - domande a cui non sappiamo ancora rispondere
    - ma quale la natura degli elementi? quali le interazioni ammesse? quali criteri e linee guida? a che cosa è possibile ispirarsi nel progettare questa decomposizione?

40

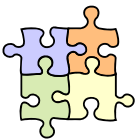
Requisiti, interessi e scenari

Luca Cabibbo – ASw



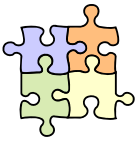
## Sviluppo dell'architettura guidata da scenari

- La **progettazione dell'architettura** può essere guidata dagli scenari
  - domande a cui non sappiamo ancora rispondere
    - ma quale la natura degli elementi? quali le interazioni ammesse? quali criteri e linee guida? a che cosa è possibile ispirarsi nel progettare questa decomposizione?
  - alcune intuizioni
    - gli scenari di qualità più importanti guidano la scelta degli stili architettonici rilevanti
    - ciascuno stile architettonico dà indicazioni sul tipo degli elementi e delle interazioni ammesse – e inoltre fornisce criteri e linee guida per la loro progettazione
    - di solito, la decomposizione è guidata da un'opportuna modellazione del dominio del problema – non necessariamente da una modellazione a oggetti oppure da una modellazione delle informazioni



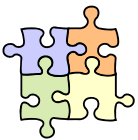
## - Scenari architettonici

- Uno **scenario architettonico** [SSA] è
  - una descrizione chiara e concisa di una **situazione** che il sistema dovrà probabilmente affrontare nel suo ambiente di produzione,
  - insieme a una definizione della **risposta** richiesta dal sistema



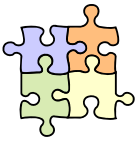
## - Tipi di scenari architeturali

- Due tipologie di scenari architeturali
  - **scenari funzionali** – sequenze di eventi esterne a cui il sistema deve rispondere
    - ad es., scenari di casi d'uso
  - **scenari di qualità** – definiti in termini di come il sistema deve reagire a un cambiamento nel suo ambiente per esibire una o più proprietà di qualità
    - un esempio di scenario di modificabilità
      - il sistema è in produzione – è richiesto che il sistema debba gestire un attributo in più (che può essere rappresentato come nuovo attributo di una relazione esistente) – il sistema deve poter essere modificato in due giorni lavorativi
    - scenari raggruppati per (macro-obiettivo di) qualità
    - scenari dettagliati in modo tale da essere verificabili



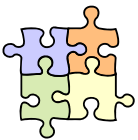
## - Identificazione di scenari

- Gli scenari possono essere identificati a partire da
  - requisiti – l'identificazione dei requisiti può partire dall'identificazione delle parti interessate e proseguire con l'identificazione degli interessi di ciascuna parte interessata – espressi (se il caso) come scenari
  - parti interessate
  - esperienza – con progetti simili



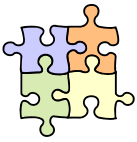
## - Prioritizzazione di scenari

- Nella definizione dell'architettura guidata da scenari
  - non tutti gli scenari sono ugualmente importanti
  - agli scenari va assegnata una *priorità* – in base a
    - valore per le parti interessate
      - importanza, frequenza, valore economico, ...
    - rischio
      - difficoltà tecniche, copertura dell'architettura, ...
  - bilanciando il grado di importanza relativa indicato dalle diverse parti interessate



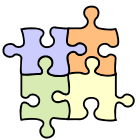
## - Usi degli scenari

- Gli scenari possono essere usati in vari modi nel processo di definizione dell'architettura
  - per fornire un input alla definizione dell'architettura
  - per guidare il processo di definizione dell'architettura
  - per valutare l'architettura
  - per la comunicazione con le parti interessate
  - per trovare requisiti mancanti
  - per guidare il processo di verifica dell'architettura



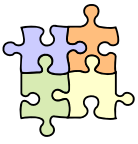
## - Descrizione di scenari

- Descrizione di *scenari funzionali*
  - *sommario* – breve descrizione di ciò che lo scenario dovrebbe illustrare
  - *stato del sistema* – prima dell'occorrenza dello scenario
  - *ambiente del sistema* – di solito, ambiente di produzione (normale oppure degradato)
  - *stimolo esterno* – definizione di ciò che causa l'occorrenza dello scenario
  - *risposta richiesta del sistema* – descrizione di come il sistema dovrebbe reagire allo scenario, dal punto di vista di un osservatore esterno
  
- Nella definizione dell'architettura, sono sufficienti descrizioni che riportano solo informazioni (architetturalmente) significative



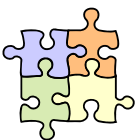
## Esempio

- Scenario: Incremental Statistics Update
- Sommario
  - come il sistema gestisce un aggiornamento di una base di dati esistente
- Stato del sistema
  - esistono già dei dati aggregati per le vendite trimestrali a cui le statistiche incrementali si riferiscono
- Ambiente del sistema
  - il sistema è in esecuzione normale, nell'ambiente di produzione
- Stimolo esterno
  - arriva un aggiornamento relativo a parte delle transazioni di vendita del trimestre precedente, mediante l'interfaccia esterna *Bulk Load Data*
- Risposta richiesta del sistema
  - i dati entranti devono attivare automaticamente l'elaborazione delle statistiche in background, per aggiornare i dati aggregati per il trimestre interessato, per riflettere i nuovi dati – i dati aggregati precedenti devono rimanere disponibili fino a quando non sono pronti i nuovi dati aggregati



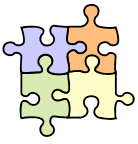
## Descrizione di scenari di qualità

- Descrizione di *scenari di qualità*
  - *sommario* – breve descrizione di ciò che lo scenario dovrebbe illustrare
  - *ambiente del sistema* – con eventuale riferimento a fatti significativi circa lo stato iniziale del sistema
  - *cambiamento dell'ambiente* – spiegazione di ciò che è cambiato nell'ambiente – ovvero la causa dell'occorrenza dello scenario
  - *comportamento richiesto del sistema* – descrizione di come il sistema dovrebbe reagire al cambiamento dell'ambiente



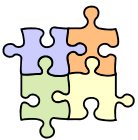
## Esempio

- Scenario: Daily Data Update Trebles (triplo) in Size
- Sommario
  - come il sistema si comporta quando il volume regolare dei dati viene ecceduto in modo significativo
- Ambiente del sistema
  - il sistema è in esecuzione normale, nell'ambiente di produzione – esistono già dei dati aggregati nella base di dati – il volume giornaliero normale dei dati è di 1-1.5GB – in questo caso l'elaborazione richiede circa 3-4 ore
- Cambiamento dell'ambiente
  - arriva un aggiornamento giornaliero di 4GB
- Comportamento richiesto del sistema
  - il sistema deve elaborare i dati fino a quando il periodo di elaborazione non eccede un tempo prestabilito (e configurabile) – a quel punto l'elaborazione dei dati deve essere interrotta, devono essere mantenute le statistiche precedenti, e lasciato un messaggio diagnostico sulla console del sistema



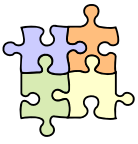
## Esempio

- Scenario: Failure in Summary Database Instance
  - cambiamento dell'ambiente
    - nello scrivere le nuove statistiche nella base di dati, il sistema riceve un'eccezione (ad es., la base di dati segnala un errore, perché un disco è guasto)
  - ...



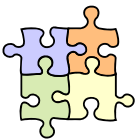
## Esempio

- Scenario: Additional Summary Dimension Required
  - cambiamento dell'ambiente
    - bisogna gestire una nuova dimensione rispetto alla quale aggregare i dati
  - comportamento richiesto del sistema
    - il team di sviluppo deve essere in grado di aggiungere la caratteristica richiesta con uno sforzo complessivo inferiore a un mese-uomo, senza cambiare la struttura complessiva del sistema

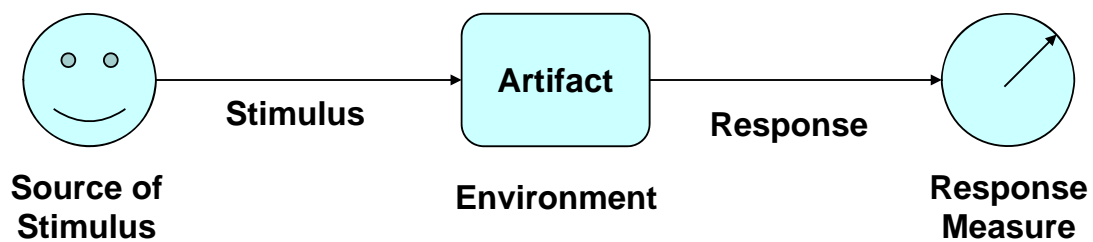


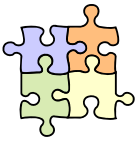
## - Scenari per attributi di qualità

- Una descrizione diversa degli scenari di qualità
- Uno **scenario per un attributo di qualità** [SAP] è
  - un requisito specifico per un attributo di qualità,
  - composto da sei parti
    - *sorgente dello stimolo* – l'entità (o attore) che genera lo stimolo
    - *stimolo* – una condizione che deve essere considerata quando arriva nel sistema
    - *ambiente* – il contesto in cui si verifica lo stimolo
    - *elaborato* – parte del sistema soggetta allo stimolo
    - *risposta* – l'attività intrapresa quando si verifica lo stimolo
    - *misura della risposta* – specifica il requisito in modo quantitativo



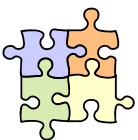
## Scenari per attributi di qualità





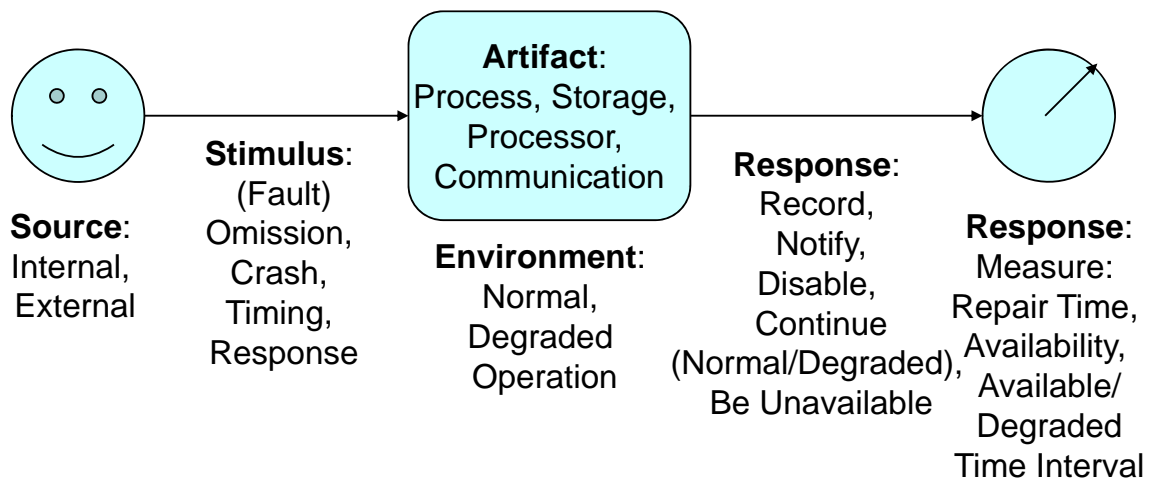
## Scenari per attributi di qualità

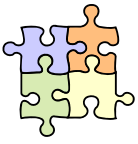
- Un esempio di scenario per le prestazioni
  - sorgente dello stimolo
    - esterna al sistema
  - stimolo
    - arriva un aggiornamento giornaliero di 1/1,5GB
  - ambiente
    - il sistema è in esecuzione normale
  - elaborato
    - sistema – o sottosistema di gestione delle statistiche
  - risposta
    - le statistiche sono aggiornate
  - misura della risposta
    - entro 3-4 ore



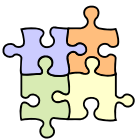
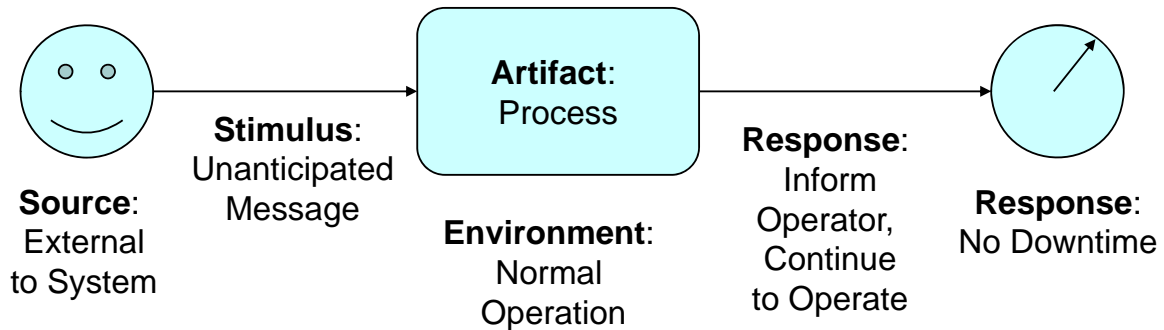
## Scenari per attributi di qualità

- Inoltre, [SAP] identifica alcune classi di qualità (ad es., disponibilità) e propone possibili valori per i vari parametri della descrizione
  - offre dunque un catalogo di scenari (parametrici) predefiniti



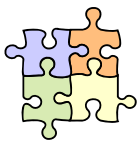


## Esempio



## \* Applicare gli scenari

- Ci sono vari modi per applicare gli scenari
  - ovvero, per descrivere come l'architettura reagisce/deve reagire allo stimolo che caratterizza lo scenario
- Attenzione, non tutti gli scenari sono importanti allo stesso modo
  - le tecniche più costose e precise vanno applicate solo nei casi veramente importanti, relativi ad aree di rischio maggiore



## - Modelli

### □ Modelli (cartacei)

- uno scenario viene usato durante la creazione di un modello
- ad es., diagrammi di interazione di UML o descrizione dei flussi di dati/controllo

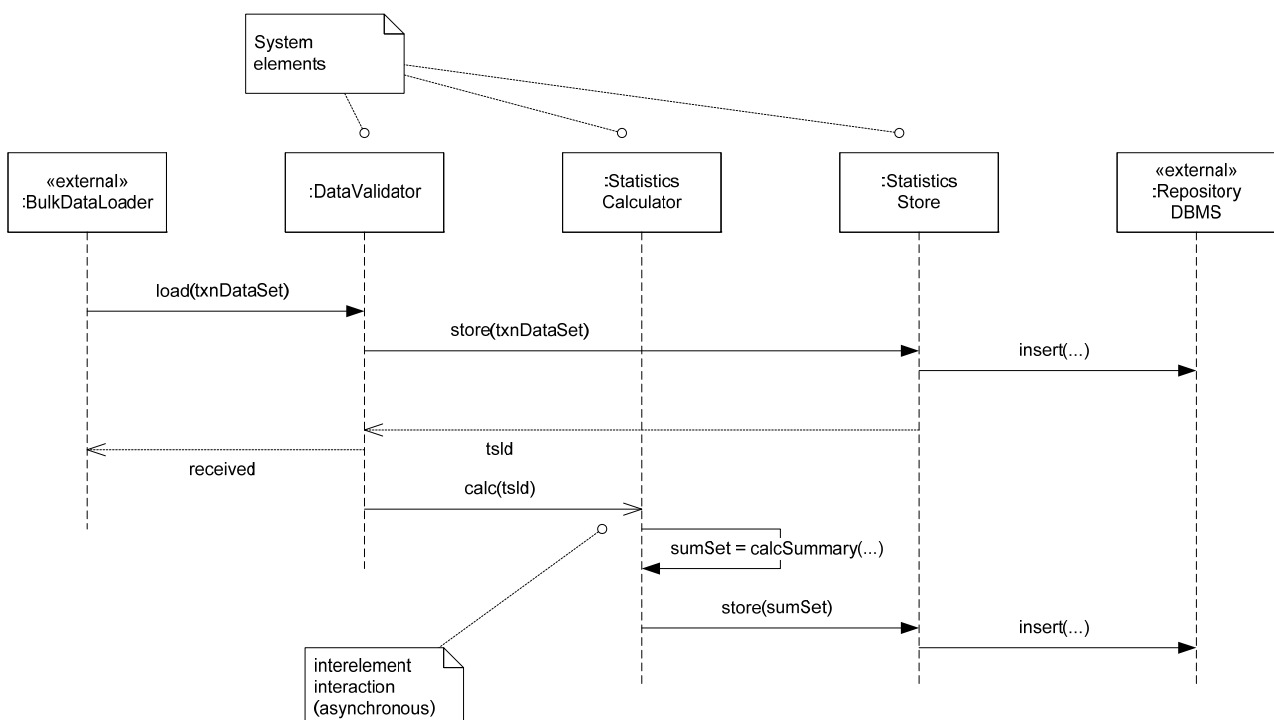
### □ Conseguenze

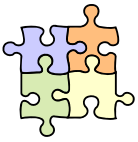
😊 semplici da creare e comprendere

☹️ sono modelli inerti, che possono essere solo rivisti – ma non verificati



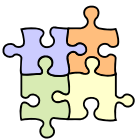
## Esempio





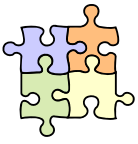
## - Walkthrough

- I modelli “cartacei” possono essere valutati e validati rivedendoli in un gruppo di persone
  - *walkthrough* – “passeggiata” o “prova” (prova teatrale)
  - una presentazione formale, con un uditorio competente e attivo, in cui viene illustrato come i vari elementi del sistema si comportano nella gestione di uno o più scenari
    - gli scenari guidano l’organizzazione della presentazione
  - simile ad una *revisione formale* di un documento – ma l’ordine della presentazione è guidato dagli scenari e non dalla sequenzialità del documento
- Conseguenze
  - ☺ efficace nel trovare difetti e dimenticanze
  - ☹ preparazione ed organizzazione complessa
  - ☹ i risultati effettivi dipendono dalla preparazione dei partecipanti



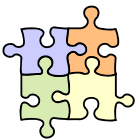
## - Simulazioni

- Simulazioni
  - basate su opportuni modelli matematici
  - ci sono strumenti di modellazione grafica (ad es., UML) che consentono alcune simulazioni
  - ci sono simulatori dedicati ad alcune qualità – ad es., prestazioni
- Conseguenze
  - ☺ approccio efficace in alcuni casi, e meno costoso dello sviluppo di prototipi
  - ☹ i risultati sono realistici? i risultati non possono essere riusati in fasi successive dello sviluppo (un prototipo potrebbe esserlo)



## Altri modi di applicare gli scenari

- Prototipazione
- Conseguenze
  - 😊 può fornire un livello di confidenza elevato – da usare nei casi di rischio più elevato
  - ☹️ costoso
- Test del sistema effettivo
  - gli scenari possono fornire la base per pianificare i test di sistema (di accettazione) del sistema reale



## - Uso efficace di scenari

- Alcune buone pratiche per l'applicazione efficace degli scenari
  - identifica un insieme focalizzato di scenari
    - utile trovare molti degli scenari – ma è bene concentrarsi principalmente su quelli più importanti
  - usa scenari distinti
    - molti scenari possono variare tra loro di poco
    - utile identificare gruppi di scenari, con alcuni scenari principali – dando minore importanza alle variazioni
  - usa gli scenari presto
    - quando l'architettura sta prendendo forma – difficile imporre scenari significativi in un secondo momento
  - usa anche scenari di qualità – oltre a quelli funzionali
  - usa anche scenari di fallimento
  - coinvolgi le parti interessate