Appunti di processo e sviluppo del software

Daniele Besozzi

Anno accademico 2025/2026

Contents

Premesse

Questi sono appunti realizzati per riassumere e schematizzare tutti i concetti presentati durante il corso di processo e sviluppo del software tenuto presso il corso di laurea magistrale in informatica presso l'università degli studi di Milano Bicocca. Lo scopo di questo documento non è quello di sostituire le lezioni del corso o di essere l'unica fonte di studio, bensì integrare le altri fonti con un documento riassuntivo.
Mi scuso in anticipo per eventuali errori e prego i lettori di segnalarli contattandomi via mail all'indirizzo d.besozzi@campus.unimib.it.

Chapter 1

Requirements Engineering

Per essere sicuri che una soluzione software risolva correttamente un problema del mondo reale dobbiamo prima comprenderlo completamente e definire:

- Quale problema sia da risolvere
- Il contesto da cui il problema parte

Definiamo come **mondo** la parte problematica del mondo reale, composto da componenti umani e componenti fisici. Chiamiamo invece **macchina** (o sistema) ciò che deve essere installato per risolvere il problema, cioè software o soluzione hardware-software. Il requirements engineering (RE) riguarda gli effetti della macchina sul mondo reale, le assunzioni e proprietà rilevanti del mondo.
Il **system as-is** è il sistema allo stato attuale, precedente l'installazione della macchina. Il **system to-be** è il sistema futuro, come sarà una volta installata la macchina.
In una definizione preliminare di RE possiamo dire che è un insieme di attività volto a esplorare, valutare, documentare, consolidare, ripassare e adattare gli obiettivi, capacità, qualità, requisiti e assunzioni di un sistema software-intensive. È basato sui problemi che sorgono nel sistema as-is e le opportunità portate dalle nuove tecnologie. Le difficoltà principali del RE sono:

- Scope ampio: diverse versioni del software (as-is, to-be, to-be-next) e ambienti ibridi (organizzazioni, leggi, politiche, device)
- Considerazioni multiple: funzionali, di qualità, di sviluppo.
- Livelli di astrazione
- Stakeholder multipli: con background diversi, interessi diversi e punti di vista contrastanti.
- Task intersecate durante il processo iterativo di elicitazione, valutazione, specifica e consolidazione.

Tre domande fondamentali che dobbiamo porci sono: perché installare un nuovo sistema? Quali servizi? Chi è responsabile per cosa?

- **Perché?** Identificare, analizzare, rifinire gli obiettivi del sistema to-be per risolvere problematiche rilevate nel sistema as-is, allinearsi con gli obiettivi business e sfruttare nuove tecnologie.
  Le difficoltà principali sono:
  - Acquisire conoscenze del dominio
  - Valutare le varie alternative
  - Identificare e risolvere conflitti tra obiettivi
- **Cosa?** Identificare, definire i servizi funzionali del sistema to-be
  - Soddisfare gli obiettivi identificati
  - In concomitanza con i requisiti di qualità: prestazioni, sicurezza…
  - Basarsi su assunzioni realistiche dell'ambiente.

  Le difficoltà principali sono:

  - Identificare le feature corrette.
  - Specificarle in maniera precisa in maniera che tutti possano capire.
  - Assicurare la tracciabilità tra obiettivi.

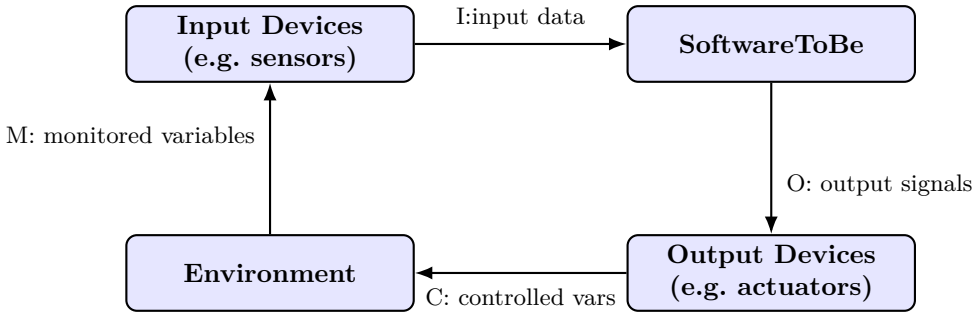
- **Chi?** Assegnare le responsabilità per gli obiettivi, servizi requisiti tra i componenti del sistema to-be.
  - Basandosi sulle capacità e obiettivi del sistema.
  - Definire il limite tra software e ambiente.

  Le difficoltà principali sono:

  - Valutare soluzioni alternative per decidere il giusto livello di automazione.

1.1 Tipi di requisiti

Il modo in cui i requisiti vengono espressi possono essere divisi in **descrittivi** (modo indicativo) e **prescrittivo** (modo ottativo).



1.1.1 Requisiti di sistema

Sistemi prescrittivi che si riferiscono a fenomeni dell'ambiente (non necessariamente condivisi). Vengono soddisfatti dal sistema to-be, possibilmente integrato con altri componenti del sistema. Devono essere comprensibili da tutti gli stakeholder. $SysReq \subseteq M \times C$ relazione tra ambiente monitorato e variabili controllate.

1.1.2 Requisiti software

Affermazioni prescrittive che si riferiscono a fenomeni condivisi tra ambiente e software. Vengono soddisfatti unicamente dal software, formulati nel vocabolario degli sviluppatori. $SofReq \subseteq I \times O$ relazione tra input e output del software.

1.1.3 Proprietà di dominio

Affermazioni descrittive riguardanti i fenomeni del mondo (rimangono veri a prescindere del sistema to-be). $Dom \subseteq M \times C$ leggi che non possono essere violate.

1.1.4 Assunzioni

Affermazioni che l'ambiente del software-to-be deve soddisfare. Formulate in termini di fenomeni di ambiente. Generalmente prescrittive (e.g. sensori e attuatori). $Asm \subseteq M \times C \cup M \times I \cup C \times O$

1.1.5 Definizioni

Affermazioni che forniscono un preciso significato ai concetti di sistemi e termini ausiliari. Non hanno valore di verità, non ha senso contestarle. $sofReq = Map(SysReq, Asm, Dom)$

I requisiti si dividono ulteriormente in:

- **Funzionali**: descrivono quali servizi il sistema deve fornire.
- **Non funzionali**: descrivono come il sistema deve essere (e.g. prestazioni, usabilità, affidabilità…).

1.2 Qualità del software

- Completezza di obiettivi, requisiti e assunzioni
- Coerenza degli elementi del documento dei requisiti (RD)
- Adeguatezza di requisiti, assunzioni e proprietà del dominio
- Non ambiguità degli elementi del documento dei requisiti
- Misurabilità di requisiti e assunzioni
- Pertinenza di requisiti e assunzioni
- Fattibilità dei requisiti
- Comprensibilità degli elementi del documento dei requisiti
- Buona strutturazione del documento dei requisiti
- Modificabilità degli elementi del documento dei requisiti
- Tracciabilità degli elementi del documento dei requisiti

Errori nei documenti dei requisiti:

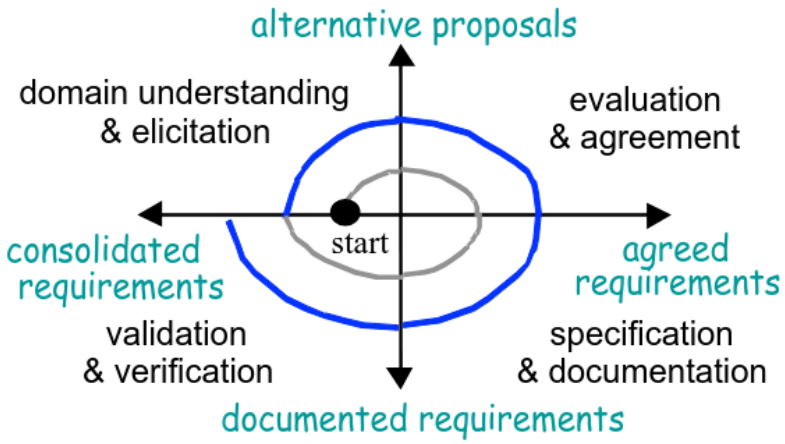
- **Omissione:** caratteristica del mondo del problema non indicata in alcun elemento del documento dei requisiti (RD).
  Esempio: nessun requisito riguardante lo stato delle porte del treno in caso di arresto di emergenza.
- **Contraddizione:** elementi del RD che descrivono una caratteristica del mondo del problema in modo incompatibile.
  Esempio: "Le porte devono essere sempre tenute chiuse tra le piattaforme" e "Le porte devono essere aperte in caso di arresto di emergenza."
- **Inadeguatezza:** elemento del RD che non descrive in modo adeguato una caratteristica del mondo del problema.
  Esempio: "Se un libro non è stato restituito, il prestatore negligente deve essere avvisato che deve pagare una multa."
- **Ambiguità:** elemento del RD che permette di interpretare una caratteristica del mondo del problema in modi diversi.
  Esempio: "Solo le persone che hanno partecipato ad almeno il 75% delle riunioni possono essere premiate alla fine dell'anno."
- **Non misurabilità:** elemento del RD che descrive una caratteristica del mondo del problema in modo tale da impedire il confronto tra opzioni o la verifica delle soluzioni.

Difetti nei documenti dei requisiti:

- **Rumore (Noise):** elemento del RD che non fornisce alcuna informazione su caratteristiche del mondo del problema.
  Variante: ridondanza incontrollata.
  Esempio: "Devono essere affissi cartelli di divieto di fumo sui finestrini del treno."
- **Sovraspecificazione (Overspecification):** elemento del RD che descrive una caratteristica non presente nel mondo del problema, ma nella soluzione della macchina.
  Esempio: "Il metodo setAlarm deve essere invocato al ricevimento di un messaggio Alarm."

- **Non fattibilità (Unfeasibility):** elemento del RD non implementabile entro i vincoli di budget o tempi.
  Esempio: "I pannelli a bordo del treno devono visualizzare tutti i voli in ritardo alla prossima fermata."
- **Inintelleggibilità (Unintelligibility):** elemento del RD incomprensibile per chi deve utilizzarlo.
  Esempio: "Negli Stati Uniti, la nozione di NWO è diventata popolare dopo gli attacchi terroristici al WTC. Tuttavia, i funzionari della NATO e dell'OMC raramente fanno riferimento a un NWO nelle procedure relative al GATT, e si può dire che MVTO, la clausola MFN e gli SRO abbiano poco a che fare con un NOW." (da un comunicato stampa)
- **Scarsa strutturazione (Poor structuring):** elemento del RD non organizzato secondo alcuna regola sensata e visibile di strutturazione.
  Esempio: "Interconnessione dei controllo di accelerazione e problemi al tracking del treno"
- **Riferimento anticipato (Forward reference):** elemento del RD che utilizza caratteristiche del mondo del problema non ancora definite.
  Esempio: uso multiplo del concetto di distanza di arresto nel caso peggiore prima che la sua definizione appaia alcune pagine dopo nel RD.
- **Rimpianto (Remorse):** elemento del RD che definisce una caratteristica del mondo del problema in modo tardivo o incidentale.
  Esempio: dopo molteplici utilizzi del concetto non definito di distanza di arresto nel caso peggiore, l'ultimo uso è seguito direttamente da una definizione incidentale tra parentesi.
- **Scarsa modificabilità (Poor modifiability):** elementi del RD i cui cambiamenti devono essere propagati in tutto il documento.
  Esempio: uso di valori numerici fissi per quantità soggette a variazioni.
- **Opacità (Opacity):** elemento del RD il cui razionale, autore o dipendenze sono invisibili.
  Esempio: "La velocità comandata del treno deve essere sempre almeno 7 mph superiore alla velocità fisica" senza alcuna spiegazione del razionale di questa scelta.

1.3 Processo di RE



Per comprendere il dominio bisogna studiare il sistema as-is, identificare gli stakeholders, così da generare delle prime proposte di prototipi e il glossario dei termini.

1.3.1 Elicitazione dei requisiti

L'**elicitazione** dei requisiti esplora i problemi del mondo facendo ulteriori analisi del problema del sistema as-is, individuandone i sintomi, cause e conseguenze. Vengono identificati:

- Opportunità tecnologiche
- Obiettivi di miglioramento
- Requisiti tecnico-organizzativi del sistema to-be
- Soluzioni alternative per soddisfare gli obiettivi e assegnare le responsabilità
- Scenari ipotetici di interazione software-ambiente
- Requisiti software, assunzioni sull'ambiente

1.3.2 Evaluazione e negoziazione dei requisiti

Presa di decisioni basate sulla negoziazione.

- Identificazione e risoluzione di conflitti
- Identificazione e risoluzione dei rischi del sistema proposto
- Confronto delle alternative tra obiettivi e rischi, selezione della soluzione preferita
- Priorità dei requisiti, per risolvere conflitti, considerare costi e supportare lo sviluppo incrementale.

Vengono prodotti le sezioni finali della proposta di prototipo che documentano gli obbiettivi concordati, i requisiti, le assunzioni e i ragionamenti che hanno portato ad essi.

1.3.3 Specifiche e documentazione

Definizione precisa di tutte le feature del sistema. Obiettivi, proprietà di dominio rilevanti, requisiti, assunzioni, responsabilità. Organizzare questi in una struttura coerente. Documentare in una forma comprensibile a tutti gli interessati. Viene prodotto il Requirements Document (RD).

1.3.4 Consolidazione dei requisiti

Attività per assicurare la qualità del RD. Vengono analizzate l'adeguatezza, la completezza, la mancanza di inconsistenze, vengono poi sistemati gli errori e i difetti. Viene prodotto un RD consolidato.

1.4 Analisi di dominio ed elicitazione dei requisiti

Il processo prevede:

- Identificare gli stakeholder e interagire con loro.
- Applicare tecniche di elicitazione *artefact-driven*.
  - Studio di background
  - Raccoglimento dati e questionari
  - Griglie di repertorio, card sorting per acquisizione di concetti.
  - Scenari, storyboard per esplorare i problemi del mondo.
  - Prototipi, mock-up per feedback rapido.
- Tecniche di elicitazione *stakeholder-driven*.
  - Interviste.
  - Osservazioni e studi etnografici.
  - Sessioni di gruppo.

Per comprendere i problemi del mondo è necessario selezionare stakeholder rappresentativi, gli aspetti rilevanti sono;

- Posizione nell'organizzazione.

- Ruolo nelle decisioni sul sistema to-be.
- Livello di conoscenza del dominio.
- Esposizione ai problemi percepiti.
- Influenza sull'accettazione del sistema.
- Obiettivi personali e conflitti di interessi.

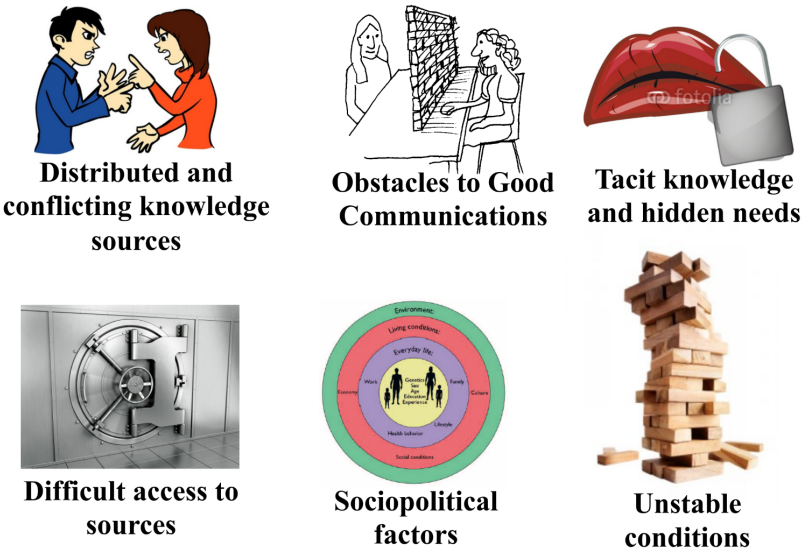


Figure 1.1: Ostacoli nell'acquisizione dei requisiti

L'interazione con gli stakeholder richiede skill di comunicazione: utilizzare la giusta terminologia, affrontare i punti chiave, fondare relazioni di fiducia. Per riformulare le conoscenze è necessario organizzare meeting per presentare la conoscenza del mondo, acquisita da fonti diverse, per integrarle in una forma strutturata.

1.4.1 Tecniche di elicitazione artefact-driven

Studio di background

Effettuare uno studio di background comprende acquisire, leggere e sintetizzare documenti riguardo:

- L'**organizzazione**: diagrammi di organizzazione, business plan, report finanziari, organizzazione meeting, etc …
- Il **dominio**: manuali, questionari, articoli, leggi, report su sistemi simili sullo stesso dominio.
- Il **sistema as-is**: workflow documentati, procedure, regole di business, documenti scambiati, report di difetti/lamentele, richieste di cambiamenti, etc …

Quali sono i problemi degli studi di background?

- **Contro**
  - Molti documenti voluminosi vanno letti.
  - Le informazioni chiave vanno estratte da molti dettagli irrilevanti.
  - Sfruttare meta-conoscenze per discriminare informazioni rilevanti da quelle inutili.
- **Pro**
  - Produce informazioni basilari utili per interagire con gli stakeholder.

Questionari

Somministrare una lista di domande a una selezione di stakeholder, ognuna con una lista di possibili risposte (con allegato un breve contesto se necessario). È possibile somministrare domande a risposta multipla o domande di peso, ovvero una serie di affermazioni d pesare in maniera:

- Qualitativa ("alto","basso",…)
- Quantitativa (percentuale)
- Per esprimere la percepita importanza, preferenza, rischio, etc …

Sono efficaci per acquisire velocemente informazioni soggettive, in maniera economica e remota per molte persone. Sono utili per preparare meglio interviste più specifiche.
La preparazione va effettuata con molta attenzione in quanto potrebbero introdurre **bias** o informazioni non affidabili (a causa di incomprensioni delle domande o delle risposte, risposte non consistenti, etc …). Le linee guida per la stesura dei questionari sono:

- Selezionare un campione di persone rappresentative e statisticamente significante. Fornire le ragioni dietro queste scelte.
- Controllare la copertura delle domande e delle possibili risposte.
- Assicurarsi che le domande, le risposte e le formulazioni siano senza bias e non ambigue.
- Aggiungere domande implicitamente ridondanti per rilevare risposte incoerenti.
- Far controllare il questionario da un ente terzo.

Storyboard

Necessaria per acquisire e validare informazioni da esempi concreti riguardanti il sistema as-is e to-be. Una story board racconta una storia tramite una sequenza di snapshot (frasi, schizzi, slide, immagini…) che descrivono un evento o una serie di eventi. Tipicamente annotati da chi è coinvolto, cosa gli succede, perché ciò accade, cosa succede se (non) accade, etc …Può essere formulato in maniera passiva (la storia viene raccontata agli stakeholder) o in maniera attiva (gli stakeholder contribuiscono).

Scenari

Gli scenari illustrano tipiche sequenze di interazioni tra componenti di sistema per raggiungere obiettivi impliciti. Possono essere utilizzati come specifiche di un test case. Vengono tendenzialmente illustrati tramite testo o diagrammi. Ne esistono di vari tipi:

- **Positivi:** un comportamento che il sistema dovrebbe coprire.
- **Negativi:** un comportamento che il sistema dovrebbe escludere.
- **Normali:** tutto procede come ci si aspetta.
- **Anormali:** una sequenza di una interazione desiderata nel caso di eccezioni (rimane positiva).

Gli scenari presentano sia vantaggi che svantaggi, pur essendo probabilmente il metodo di elicitazione più utile:

- **Pro**
  - Esempi concreti e contro esempi.
  - Appetibili per gli stakeholder.
  - Utilizzabili come test di accettazione.
- **Contro**
  - Inerentemente parziali.
  - Esplosione combinatoria.
  - Potenzialmente overkill.
  - Potrebbero contenere dettagli irrilevanti.
  - Livelli di granularità diversi tra stakeholder.

Prototipi e mock-up

Il loro obiettivo è quello di verificare l'adeguatezza dei requisiti tramite un feedback diretto degli utenti, attraverso uno schizzo del sistema to-be in azione. In particolare viene posta particolare attenzione sui requisiti non chiari o difficili da formulare, in modo da elicitarli ulteriormente. Un prototipo è una implementazione veloce di qualche aspetto, vi sono due tipi:

- **Prototipi di interfaccia utente:** si concentrano sull'usabilità mostrano form di i/o o pattern di dialogo.
- **Prototipi di funzionalità:** si concentrano su funzionalità specifiche.

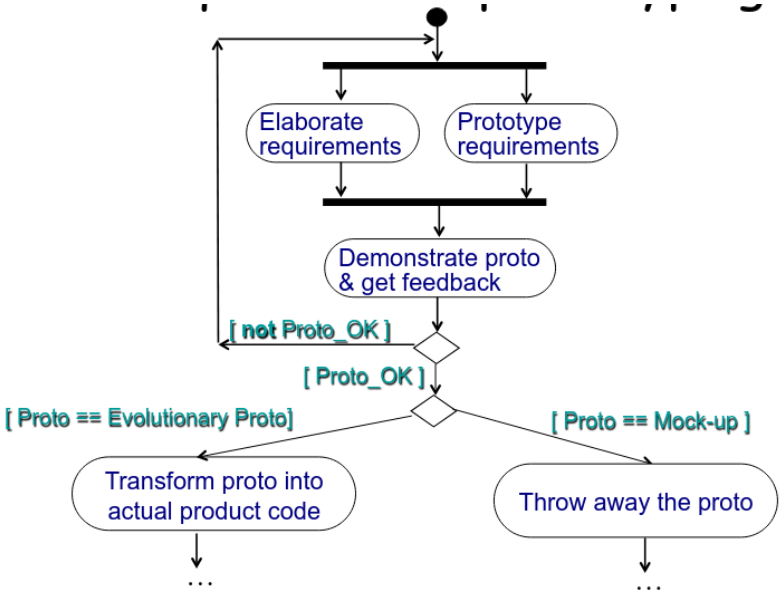


Figure 1.2: Schema di sviluppo di un prototipo

Come per le altre tecniche di elicitazione, anche i prototipi presentano vantaggi e svantaggi:

- **Pro**
  - Assaggio concreto di come sarà il software, vengono quindi chiariti i requisiti, elicitati quelli nascosti, in generale si miglioano quelli raccolti.
- **Contro**
  - Possono essere fuorvianti, alzando troppo le aspettative.
  - Il codice prodotto di fretta senza cura può essere difficile da riutilizzare.

Riutilizzo della conoscenza

Come obiettivo vogliamo velocizzare il processo di elicitazione riutilizzando la conoscenza acquisita in progetti pregressi affini. In generale il processo consiste in:

1. Ottenere la conoscenza rilevante proveniente da altri sistemi.
2. Trasportarla nel sistema target.
3. Validare il risultato, adattarlo se necessario e integrarlo con la conoscenza già conseguita.

La conoscenza potrebbe dipende o meno dal dominio. Chiaramente sono presenti vantaggi e svantaggi:

- **Pro**
  - Il processo avviene in maniera naturale.

  - Una guida significative riduce lo sforzo necessario per l'elicitazione.
  - Vengono ereditate strutture e qualità di aspetti astratti di dominio.
  - Efficace per completare i requisiti con aspetti mancanti.
- **Contro**
  - Utile solo se il dominio astratto è sufficientemente simile e accurato.
  - Definire domini astratti per facilitare il riuso è difficile.
  - Richiede lavoro per valutare l'integrazione.
  - Affinità parziali richiedono adattamenti insidiosi.

Modelli minori

- **Card sorting:** chiedere agli stakeholder di partizionare un insieme di cartelli
  - Ogni carta cattura un concetto testualmente o graficamente.
  - Le carte vengono raggruppate secondo criteri dello stakeholder.
  - L'obiettivo consiste nell'acquisire ulteriori informazioni riguardanti concetti che sono già stati elicitati.
  - Per ogni sottoinsieme di carte viene chiesto le proprietà comuni implicite utilizzate per raggrupparle.
  - Iterare con le stesse carte per nuovi raggruppamenti e proprietà.
- **Data collection:** dati marketing, statistiche d'utilizzo, metriche di performance, costi…
  - L'obiettivo è raccogliere fatti e cifre non documentati. Ciò viene fatto tramite esperimenti o una selezione di dataset di rappresentativi presi da fonti disponibili.
  - Potrebbe complementare lo studio di background.

1.4.2 Tecniche di elicitazione stakeholder-driven

Interviste

Tecnica primaria per elicitare la conoscenza.

1. Selezionare lo stakeholder specificatamente per l'informazione da acquisire.
2. Organizzare un incontro con l'intervistato, fare domande e segnare le risposte.
3. Scrivere un report con trascritto dell'intervista.
4. Sottomettere il report per la validazione e il rifinimento.

Un'intervista può coinvolgere più stakeholder, ciò può far risparmiare tempo ma può anche inibire la comunicazione. Le interviste possono essere di due tipi:

- **Strutturate:** domande predefinite, specifiche per la ragione dell'intervista. Alcune domande aperte, altre a risposta multipla.
- **Non strutturate:** nessuna domanda predefinita, discussione libera sul sistema as-is, problemi percepiti, soluzioni proposte. Vengono esplorati problemi che potrebbero essere stati tralasciati.

Una intervista efficace dovrebbe mischiare le due modalità, partendo da una parte strutturata per poi aprirsi a domande più libere quando necessario. È preferibile seguire le seguenti linee guida:

- Preparati in anticipo, concentrandoti sul problema giusto al momento giusto
  - Evita domande ovvie per l'intervistato (es. studia prima il suo background)
  - Progetta in anticipo una sequenza di domande specifica per quell'intervistato
- Centra l'intervista sul lavoro e sulle preoccupazioni dell'intervistato

- Mantieni il controllo dell'intervista
- Fai sentire l'intervistato a proprio agio
  - All'inizio: rompi il ghiaccio, fornisci una motivazione, poni domande semplici
  - Considera la persona, non solo il suo ruolo
  - Mostrati sempre come un partner affidabile
  - Fai domande del tipo "Perché?"
- Evita certi tipi di domande:
  - di parte o faziose
  - affermative
  - ovvie o impossibili da rispondere per quell'intervistato
- Rivedi e struttura le trascrizioni dell'intervista finché sono ancora fresche nella mente
  - includendo reazioni personali, atteggiamenti, ecc.
- Mantieni l'intervistato coinvolto
  - co-rivedi la trascrizione per validazione e perfezionamento

Osservazioni e studi etnografici

A volte comprendere un'azione è più facile osservandola piuttosto che tramite una spiegazione verbale o testuale. L'osservazione può essere fatta in due modi:

- **Passiva:** nessuna interferenza con colui che esegue l'azione.
  - Osservazione dall'esterno, registrazione, trascritti e analisi.
  - Analisi di protocollo: gli esecutori spiegano cosa fanno mentre lo fanno.
  - Studio etnografico: osservazione prolungata per comprendere proprietà emergenti del gruppo sociale coinvolto.
- **Attiva:** l'osservatore partecipa all'azione, diventando anche un membro del gruppo.

Vantaggi e svantaggi:

- **Pro**
  - Rivelazione della conoscenza tacita che non emergerebbe altrimenti.
  - Rivelazione di problemi nascosti che emergono da modi di fare cose macchinosi.
  - Rivelazione di aspetti sociali e culturali che influenzano il lavoro.
  - Contestualizzazione delle informazioni acquisite.
- **Contro**
  - Lento e costoso: da svolgere su lunghi periodi, a tempi diversi, su condizioni di lavoro diverse.
  - Potenzialmente inaccurato: le persone si comportano in modo diverso quando sono osservate.
  - Concentrato sul sistema as-is.

Sessioni di gruppo

Le sessioni di gruppo permettono di avere una migliore percezione, giudizio e inventiva grazie alle interazioni con un gruppo diversificato. L'elicitazione avviene tramite una serie di workshop di gruppo (su più giorni) seguiti da azioni di follow-up. Vengono utilizzati media audio visivi, grafici per stimolare la discussione e registrare i risultati. Si dividono in due tipi:

- **Strutturate:** Ogni partecipante ha un ruolo specifico, contribuisce alla elaborazione dei requisiti relativamente al suo ruolo per raggiungere una sinergia. Generalmente si focalizza su requisiti ad alto livello.
- **Non strutturate (brainstorming):** I partecipanti non hanno ruoli ben definiti, viene diviso in due fasi:
  - Generazione di idee: più idee possibili, nessuna critica.
  - Valutazione delle idee: discussione da tutti i partecipanti secondo dei criteri concordati per dare priorità alle idee.

Vantaggi e svantaggi:

- **Pro**
  - Grosse potenzialità per esplorare problemi e soluzioni.
  - Proposte più inventive per risolvere i problemi.
- **Contro**
  - Composizione del gruppo critica.
  - Consuma molto tempo per persone chiave, che generalmente sono molto occupate.
  - Richiede skill ed expertise del leader.
  - Le dinamiche di gruppo possono inibire la comunicazione.
  - Rischio di perdere il focus e la struttura, generando poco materiale utile e sprecando tempo.
  - Copertura superficiale di problemi più tecnici.

1.5 Valutazione dei requisiti

In generale si vuole compiere le seguenti attività:

- Gestione delle inconsistenze
  - Capire i tipi di inconsistenze.
  - Gestirle.
  - Gestire i conflitti in maniera sistematica.
- Valutare opzioni alternative per prendere decisioni.
- Prioritizzare i requisiti.

1.5.1 Conflitti

Una inconsistenza è una violazione delle regole di consistenza tra oggetti. Possono nascere dalla pluralità dei punti di vista degli stakeholder o da conflitti tra requisiti di qualità (ad es. sicurezza e usabilità). Le inconsistenze devono essere identificate, analizzate e risolte. Questo deve essere fatto non troppo presto, per permettere una ulteriore elicitazione e non troppo tardi, per permettere lo sviluppo software. Le inconsistenze si dividono in:

- **Conflitti di terminologia:** Stessi concetti chiamati in modo diverso in affermazioni diverse.
- **Conflitti di designazione:** Stesso nome per concetti diversi.
- **Conflitti di struttura:** Stesso concetto strutturato in maniera diversa in affermazioni diverse.

- **Conflitto forte:** Affermazioni non possono essere vere contemporaneamente.
- **Conflitto debole (divergenze):** Affermazioni non soddisfacibili contemporaneamente sotto alcuni vincoli.

Le inconsistenze di terminilogia, designazione e struttura sono generalmente risolvibili tramite l'utilizzo di un glossario dei termini. I conflitti forti e deboli hanno cause più profonde e difficili da risolvere. Potrebbero esserci obiettivi personali degli stakeholder in conflitto, che devono essere gestiti alla base e propagati. Alcune inconsistenze vengono inerentemente portate dalla presenza di requisiti non funzionali.

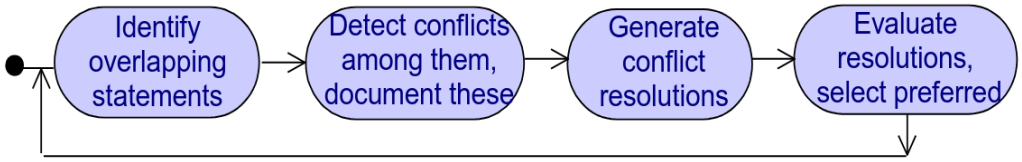


Figure 1.3: Gestione dei conflitti in modo sistematico

- Un accavallamento è la presenza di riferimenti agli stessi termini o concetti in affermazioni diverse.
- Il rilevamento dei conflitti viene svolto in maniera informale, applicando delle euristiche su categorie di requisiti in conflitto. Viene poi svolto un processo formale.
- Per una risoluzione ottimale è meglio esplorare più soluzioni alternative, confrontarle e scegliere la migliore. Per generare soluzioni è possibile utilizzare tecniche di elicitazione o applicare tattiche di risoluzione. Queste ultime si dividono in:
  - Evitare requisiti limitanti.
  - Restaurare affermazioni conflittuali.
  - Indebolire affermazioni conflittuali.
  - Rinunciare a requisiti di minor importanza.
  - Specializzare i la sorgente del conflitto, o l'oggetto del conflitto.
- I criteri per la valutazione di una soluzione sono:
  - Contribuzione a requisiti non funzionali critici.
  - Contribuzione alla risoluzione di altri conflitti.
  - Applicazione di principi di analisi dei rischi.

1.5.2 Prioritizzazione dei requisiti

I requisiti dopo essere stati elicitati e valutati devono essere prioritizzati per:

- Risolvere conflitti tra requisiti.
- Assegnare risorse limitate.
- Supportare lo sviluppo incrementale.
- Gestire cambiamenti futuri inaspettati.

Alcuni principi per la prioritizzazione sono:

1. Pochi livelli di priorità.
2. Livelli relativi.
3. Rendere i requisiti confrontabili (stesso livello di granularità, stesso livello di astrazione).
4. Non rendere i requisiti mutualmente esclusivi.
5. Far accordare i requisiti a tutti gli stakeholder.

Prioritizzazione valore-costo

Tecnica sistematica che rispetta i principi (1) e (2). Diviso in tre fasi:

1. Stima relativa del contributo di ogni requisito al valore del progetto.
2. Stima relativa del costo di ogni requisito.
3. Plot del grafico valore-costo e selezione dei requisiti.

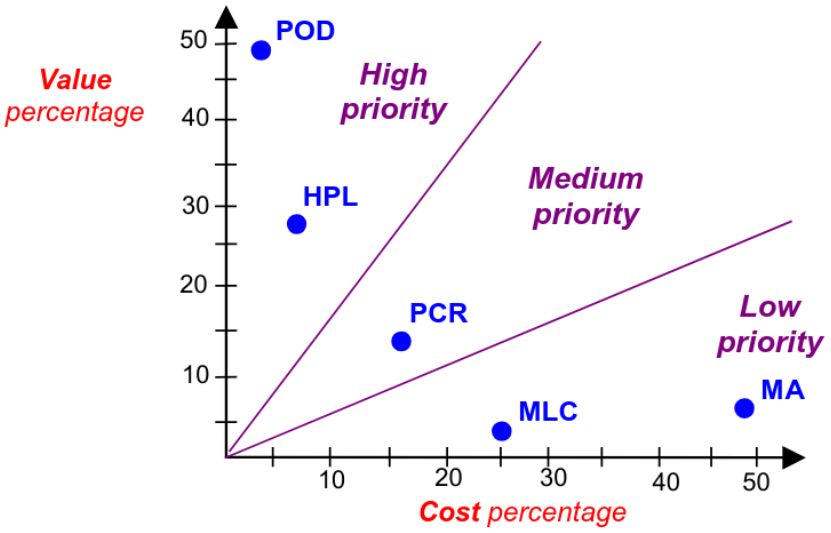


Figure 1.4: Prioritizzazione valore-costo

In particolare i punti segnati in figura sono:

- POD: Produce Optimal meeting Dates
- HPL: Handle Preferred Locations
- PCR: Parameterize Conflict Resolution
- MLC: Support Multi-Lingual Communication
- MA: Provide Meeting Assistant

La tecnica AHP permette di determinare in quale proporzione ciascun requisiti $R_1, R_2, \ldots, R_n$ contribuisce al criterio $Crit$. Viene prima applicato con $Crit = Valore$ e poi con $Crit = Costo$.

1. Viene prima costruita una matrice di confronto, dove il contributo a $Crit$ di ogni requisito R_i viene confrontato con quello di ogni altro requisito R_j.
2. Viene poi determinato come $Crit$ è distribuito tra gli R_i.

Step 1

Scala per confrontare il contributo di R_i al *Criterio* rispetto a R_j:

1. Contribuisce in egual misura
2. Contribuisce leggermente di più
3. Contribuisce decisamente di più
4. Contribuisce molto di più
5. Contribuisce estremamente di più

Nella matrice di confronto: $R_{ji} = \frac{1}{R_{ij}} \quad (1 \leq i, j \leq N)$
Matrice di confronto:

Criterio: valore	Produrre data ottimale	Gestire località preferite	Parametrizzare strategia di risoluzione dei conflitti	Comunicazione multilingue	Assistente per riunioni
Produrre data ottimale	1	3	5	9	7
Gestire località preferite	1/3	1	3	7	7
Parametrizzare strategia di risoluzione dei conflitti	1/5	1/3	1	5	3
Comunicazione multilingue	1/9	1/7	1/5	1	1/3
Assistente per riunioni	1/7	1/7	1/3	3	1

Step 2

Per determinare la distribuzione vengono usati gli autovalori della matrice di confronto.

- 2.a) Normalizzazione delle colonne: $R'_{ij} = R_{ij} / \sum_i R_{ij}$
- 2.b) Media sulle righe: $Contrib(R_i, Crit) = \sum_j R'_{ij}/N$

Criterio: valore	Produrre data ottimale	Gestire località preferite	Parametrizzare strategia di risoluzione dei conflitti	Comunicazione multilingue	Assistente per riunioni
Produrre data ottimale	0.56	0.65	0.52	0.36	0.38
Gestire località preferite	0.19	0.22	0.31	0.28	0.38
Parametrizzare strategia di risoluzione dei conflitti	0.11	0.07	0.10	0.20	0.16
Comunicazione multilingue	0.06	0.03	0.02	0.04	0.02
Assistente per riunioni	0.08	0.03	0.03	0.12	0.05

1.6 Specifiche e tecniche di documentazione

Come documentare i requisiti?

1.6.1 Linguaggio naturale

- **Documentazione libera in linguaggio naturale senza restrizioni:** Prosa scritta in linguaggio naturale senza vincoli. L'espressività non ha limiti, la comunicazione è efficace e non richiede skill particolari, Tuttavia è prona a errori e difetti. Le ambiguità sono inerenti al linguaggio naturale.
- **Documentazione in linguaggio naturale con vincoli:** Regole locali riguardanti la scrittura dei requisiti. Regole globali riguardanti l'organizzazione del documenti dei requisiti.

Buone regole per la scrittura dei requisiti:

- Identificare i lettori e scrivere di conseguenza.
- Dichiarare prima cosa si intende fare, poi eseguirlo.
- Motivare prima le scelte; riassumere le conclusioni alla fine.
- Assicurarsi che ogni concetto sia definito prima dell'uso.
- Chiedersi continuamente: "È comprensibile? È sufficiente? È pertinente?".

- Non inserire più di un requisito, un'assunzione o una proprietà del dominio in una singola frase. Tenere le frasi brevi.
- Usare "deve" per prescrizioni obbligatorie e "dovrebbe" per prescrizioni desiderabili.
- Evitare gergo e acronimi non necessari.
- Usare esempi esplicativi per chiarire affermazioni astratte.
- Fornire diagrammi per relazioni complesse tra elementi.

1.6.2 Notazioni semi-formali in diagramma

Vengono utilizzati per complementare la documentazione in linguaggio naturale o per rimpiazzarla. Vengono dedicati ad aspetti specifici del sistema (as-is o to-be). Dato che sono strumenti grafici rendono la comunicazione facile e veloce. Sono comunque semi-formali dato che alcune dichiarazioni rimangono in linguaggio naturale. Quando si vuole combinare dati e attività si possono utilizzare i **diagrammi**

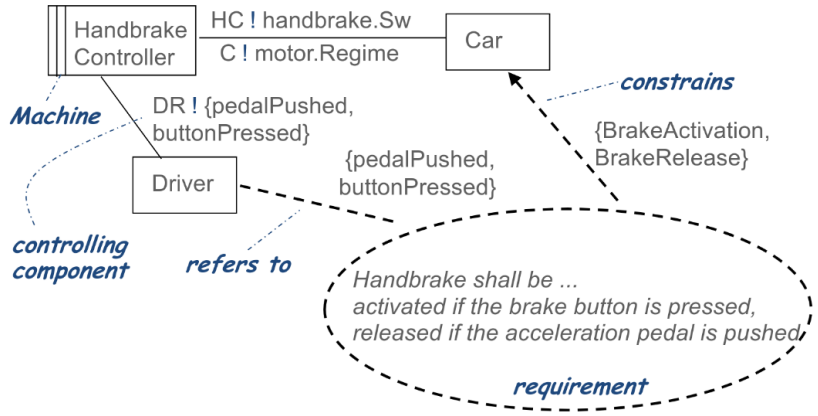


Figure 1.5: Esempio di diagramma dei problemi

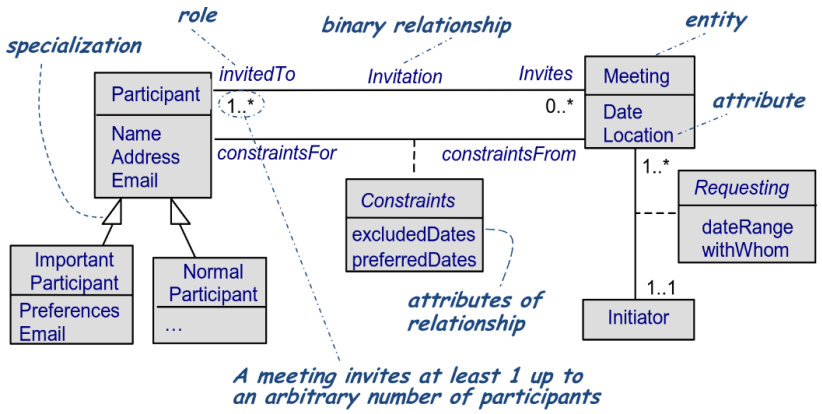


Figure 1.6: Esempio di diagramma delle entità e relazioni

SADT. Servono a catturare le attività e i dati del sistema as-is e to-be. È diviso in due parti:

- **Actigram:** Relaziona le attività in base a legami di dipendenza dei dati.
- **Datagram:** Relaziona i dati in base a legami di dipendenza di attività.

I dati che compaiono in un actigram devono apparire nel datagram e viceversa.
Le regole di consistenza e completezza possono essere controllate tramite l'uso di tool specifici. Ogni attività deve avere un input e un output. Tutti i dati devono avere un produttore e un consumatore. I

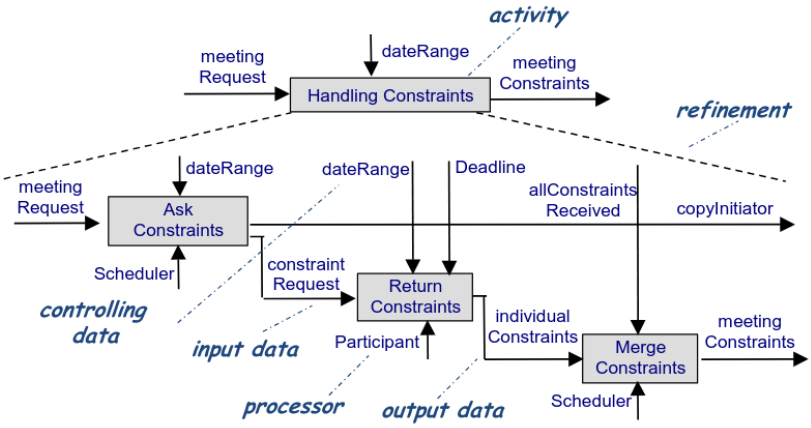


Figure 1.7: Esempio di diagramma SADT

dati di I/O di un'attività devono essere dati di I/O delle sotto attività. Ogni attività nel datagram deve essere definito in un actigram.
I diagrammi per documentare le operazioni di sistema sono i casi d'uso, mentre per rappresentare scenari di interazioni si usano i diagrammi che tracciano gli eventi, come i sequence diagram. Infine per documentare i comportamenti di sistema si possono usare i diagrammi di macchine a stati.
Le notazioni a diagramma portano vantaggi e svantaggi:

- **Pro**
  - Dichiarazione formale di diversi aspetti del sistema + notazioni informali sulle proprietà per maggiore precisione.
  - Notazione grafica, quindi facilmente comprensibile e fornisce una struttura.
  - Possibilità di effettuare query tramite tool.
- **Contro**
  - La semantica potrebbe essere vaga (interpretazioni diverse).
  - Forme di analisi limitate.
  - Modellazione dei soli aspetti funzionali e strutturali.

Sorge quindi il bisogno di avere notazioni formali.

1.6.3 Notazioni formali

Vengono utilizzate per complementare le specifiche in linguaggio naturale o in forma di diagramma, specialmente per aspetti critici. Viene effettuata una formalizzazione degli elementi del documento dei requisiti.

- Dichiarazione: struttura degli elementi (tipicamente diagrammi).
- Asserzioni: proprietà degli elementi.
- Meccanismi per strutturare specifiche grandi in modo modulare.

Per formale si intende un formato processabile da una macchina, spesso basato su logica matematica. Prevede quindi una sintassi, una semantica e delle regole di inferenza. I benefici portati sono:

- Precisione più alta nelle formulazioni.
- Regole più precise di interpretazione.
- Automazione di controlli e derivazioni.

1.7 Tecniche per il controllo qualità