Ingegneria dei Requisiti

Corso tenuto dal Professor Mariano Ceccato

Università degli Studi di Verona

Alessio Gjergji

Indice

Capitolo 1

Introduzione

Ci sono problematiche legate al mondo circostante, quindi anche sistemi legacy. Anche la natura fa parte dei componenti fisici su cui interagire (pensiamo a dover pilotare dispositivi con una grossa massa). Dobbiamo tener contro di tutto, tenere in considerazione anche il contesto del mondo reale all'esterno dei software. Proprietà, parliamo di tutto l'insieme.

I requisiti descrivono solo i problemi del mondo reale, come il software deve interagire con il mondo reale. Ragioniamo su proprietà del mondo reale, la progettazione del software è solo una conseguenza.

Non troviamo soluzioni, ma raccogliamo problemi, responsabilità, ma non di soluzioni. Il software è solo una conseguenza.

Abbiamo due versioni del mondo:

- **System as is**: il sistema come è prima dello sviluppo del software. Il mondo senza il software.
- **System to be**: il sistema come sarà dopo lo sviluppo del software.

Dobbiamo catturare i vincoli e le opportunità del mondo reale, per mapparli in requisiti software. I requisiti si dividono in: L'unione del mondo e del software è il sistema. Dobbiamo ragionare in tre prospettive:

- **Quali sono gli obiettivi del sistema?** perché ci serve un nuovo sistema?
- **Quali sono le funzionalità del sistema?** cosa deve fare il sistema? Iniziamo a parlare di requisiti. Che dovranno dare una risposta a cosa deve fare il sistema.
- **Quali sono i responsabili del sistema?** chi deve fare cosa? Anche oggetti necressari.

1.0.1 Perché

Si tratta di identificare quali sono gli obiettivi del sistema. Identificare ma che vanno analizzati e rifiniti. Tipicamente quello che si raccoglie dagli stakeholders è ad alto livello, è anche opportuno sapere quando smettere, criteri di completezza.

Ci sono difficoltà intrinsiche. Complesso anche il dominio, bsiogna avere un background, capire il contesto, la complessità del dominio.

Loro danno tutto per assunto. Ci sono diverse alternative da valotare, è opportuno limitare le scelte per far scelte consapevoli dopo.

Potrebbero esserci obiettivi in contrasto, che vanno in direzioni opposte. Confidenzialità e accessibilità, ad esempio. si scontra con la facilità di utilizzo che vanno in direzioni opposte.

Opportuno a questo livello identificare obiettivi contrastanti.

1.0.2 Cosa

Mira a identificare le funzionalità del software. Parliamo di requisiti, l'obiettivo è dare risposta al perché. Cosa deve fare il sistema? rispondendo in maniera adeguata, ci saranno metriche di qualità. Risposta in maniera realistica, soluzione compatibile con i vincoli ambientali.

Esempio calcolo dell'accelerazione del treno, adeguata e sicura per i passeggeri. Non solo calcolo, ma anche sicurezza. Informazioni utili ai passeggeri all'interno del treno.

Difficoltà: identificare le necessità degli stakeholders, opportuno il linguaggio comprensibile a tutti. Garantire una tracciabilità degli obiettivi rispetto alle funzionalità del sistema. Deve esserci una motivazione alla funzionalità, quindi un link da mantenere. Se ci accorgiamo che il perché è cambiato, probabilmente anche il cosa cambierà.

1.0.3 Chi

Fornire le responsabilità, assegnate a device, hardware, software, utenti, collaboratori, ... Nel caso di accelerazione del treno potrebbe essere il macchinista, o un software di controllo. Il macchinista potrebbe avere anche un software di supporto.

La difficoltà sta nelle varie alternative che danno diverso grado di autonomia. è opportuno adottare un approccio iterativo dove le alternative vengono raffinate gradualmente.

Registro descrittivo: registro indicativo, raccontiamo le proprietà del sistema indipendentemente da come dovrebbe funzionare. Regole universali vincoli intrinseci fisici. "Se il treno è in accelerazione, allora la velocità non è nulla"

Prescrittive: le porte dovrebbero rmanere chiuse durante la marcia del treno. Proprietà desiderabile. Il secondo tipo otative possono essere manovrate, identificare il margine di manovra. Quelli descrittivi non sono negoziabili.

Le frasi possono essere collocate in contesti diversi, potrebbero riferirsi all'ambiente o sono condivisi tra ambiente e software. Non ci sono requisiti sulla parte software.

Requisiti di sistema e software. Tipicamente nei requisiti di sistema abbiamo frasi prescrittive che si riferiscono ai fenomenti ambientali e devono essere soddisfatti dal software. Formulate con un vocabolario comprensibile da tutti gli stakeholders. Nei requisiti software abbiamo frasi prescrittive che si riferiscono ai fenomeni condivisi che sono supportate dal software che deve essere soddisfatto. Grandezze modellate all'interno del software.

Tipo di frasi all'interno dei requisiti: Abbiamo proprietà di dominio, assunzioni e definizioni. Una proprietà di dominio è una frase descrittivia su un fenomeno del mondo reale, indipendente dall'esistenza del sistema software.

$$\texttt{velocitaMisurata} \geq 0 \rightarrow \texttt{accelerazione} \neq 0$$

Frasi soddisfatte dall'ambiente

$$\texttt{velocitaMisurata} \neq 0 \Leftrightarrow \texttt{accelerazione} \neq 0$$

Definizione: una frase che definisce un concetto del dominio. La definizione di velocitaMisurata è la velocità del treno misurata dal tachimentro del treno.

Requisiti funzionali e non funzionali: I requisiti funzionali raccontano del funzionalmento del software. Responsabilità del software, funzionalità da esporre. I requisiti non funzionali raccontano della qualità del software. Come per esempio la velocità di risposta, la sicurezza, la confidenzialità. "il comando di apertura delle porte deve essere eseguito entro 2 secondi" Rappresentano vincoli che di fatto rescriveranno quelli che sono i vincoli architetturali che ci guidano nella scelta della architettura. è opportuno adottare una tassonomia dei requisiti non funzionali. In modo da avere una visione chiara dei requisiti non funzionali.

Ci sono alcuni punti in cui i requisiti non funzionali si trasformano in requisiti funzionali. Ad esempio la sicurezza. Nel contesto del firewall, la sicurezza è un requisito diventa funzionale.

Mandare i comandi troppo spesso ad esempio potrebbero causare overhead.

Tipicamente molto utili.

1.1 Ciclo di vita dei requisiti

Il processo di acquisizione dei requisiti è un processo iterativo. Prima di tutto è necessario comprendere il dominio, è importante comprendere il contesto in cui il sistema dovrà operare.

Principalmente si studia il mondo prima che il mondo as is. Si studia la organizzazione, ruoli, prassi, i punti di forza e deboli del sistema attuale. Si identificano gli stakeholders, definendo anche le tipologie di utenti. Chi prende decisioni.

Ci permette di formare un primo glossario di terminologia. Gli oggetti possono avere gestioni diverse.

Dopodicchè si passa alla elicitation, ovvero l'identificazione dei requisiti. Possiamo raccogliere i primi requisiti. Ci sarà una parte di negoziazione, per identificare i requisiti che sono in contrasto. Come gestire i conflitti. Vanno identificati i rischi e le opportunità. Diverse alternative, le prioritizzazioni. Il prodotto finale è il documento dei requisti, che verrà popolato man mano.

Il passo successivo è quindi la documentazione dei requisiti, ha come obiettivo fissare in maniera precisa i requisiti. Deve essere comprensibile e formulato in maniera che tutti possano comprenderlo.

La fase successiva è la validazione e verifica, per capire se è completo corretto, se ci sono lacune, se ci sono errori, se ci sono ambiguità. Ci sono varie metodologie per fare la validazione e verifica. Se danno risposta coretta e completa alla necessità degli stakeholders. Si controllano le contraddizioni per essere risolte. Dovrà essere verificato rispetto a target di qualità. e tutto andrà discusso. L'output è un documento di requisiti validato e verificato.

Può fornire il vantaggio per il piano di test di accettazione, scenari ad alto livello che hanno senso per un utente del sistema. Non sono eseguibili non appena il sistema è stato implementato. Serve copertura esaustiva, test per ogni requisito.

Posso abbozzare un piano di sviluppo e questo documento può essere usato come contratto di fornitura del software. Una sorta di preventivo.

Quando ho la prima versione itero nel ciclo. L'attività può essere continua. Soprattutto nel caso di modelli agili. Insegue quindi i requisiti mutevoli.

1.2 Obiettivi di qualità dei requisiti

- Completezza: tutti i requisiti sono stati catturati?
- Consistenza: non ci sono contraddizioni.
- Adeguatezza
- Non ambiguità: non devo lasciar spazio a interpretazioni implicite.
- Misurabilità: devo poter misurare se il requisito è stato soddisfatto.
- Pertinenti: devono essere rilevanti per gli stakeholders.
- Realizzabilità: devono essere realizzabili.
- Flessibilità: devono essere flessibili.
- Comprensibilità: devono essere comprensibili.
- Buona struttura: devono essere ben strutturati.
- Modificabilità: devono essere modificabili.
- Traceability: devono essere tracciabili. Link tra requisiti e sorgenti. Obiettivi di alto livello che rendono necessario un requisito. Analogamente anche per decisioni alternativi.

Gli errori nei requisiti sono molto costosi. Non bisogna mai omettere qualcosa, contraddire qualcosa, avere ambiguità. Altrimenti potrei avere soluzioni sub-ottimali. Soluzioni di ripiego quindi, errori di progettazione e sviluppo in caso di contraddizioni. Se i requisiti sono inadeguati potrei avere un sistema che non soddisfa le aspettative.

Capitolo 2

Elicitazione dei requisiti

Arrivare a una comprensione del dominio e capire informazioni. Perchè è la sorgente di informazione. Prima bisogna raggiungere informazione sul dominio, studiando il sistema prima che il software arrivi, in modo da capire i flussi di lavoro, le prassi, le regole, le procedure, le politiche, le normative, le leggi, i regolamenti. Analizzare i problemi prima del software, terminologia di base. Individuare le opportunità e identificare gli stakeholders, suddividendoli per ruoli e responsabilità. Obiettivi vincoli e tutto questo è acquisizione di conoscenza che prende il nome di elicitazione di requisiti.

Ci sono due classi principali:

- Artefact-driven: si basa su documenti, modelli, specifiche,
- Stakeholder-driven: si basa su interviste, focus group, brainstorming.

Coinvolgere un campione rappresentativo di stakeholders, quindi almeno un individuo per ogni ruolo. Ci possono essere conflitti di interesse, quindi bisogna mediare. Acquisire questa conoscenza dagli stakeholders è complesso, perché le sorgenti di informazioni possono essere diverse, questo va gestito. Un alto problema è che spesso è difficile agganciare le persone chiave, perché possono essere impegnate in altre attività legate all'azienda. Diverse persone possono avere un background diverso, quindi possono avere una visione diversa del problema. Spesso danno per scontato concetti impliciti. La differenza tra una fattura e ordine di acquisto per esempio. bisogna quindi fare domande per capire questi concetti impliciti. Il problema duale è che a volte ci sono dettagli irrilevanti, che possono non essere rilevanti per il sistema. Ci potrebbe essere resistenza al cambiamento, non da sottovalutare, ci può essere a vari livelli, dai dipendenti ai manager. Ci può essere un turnover di personale, tipico di aziende di consulenza.

Sono necessarie soft-skill, buon comunicatore, ascoltare, fare domande. Validare quello che si impara precedentemente, ristrotturare quando ci sono conoscenze nuove.

2.1 Studio di background

Capire il dominio, raccogliendo documentazione e riassumendo. Ci sono i diagrammi organizzazione, che tipicamente ricalca le principali aree aziendali e ci fa capire molto sull'azienda. Leggere i business plan ci fa capire le strategie dell'azienda. Leggere i report finanziari ci fa capire la salute dell'azienda.

Dopo aver capito l'organizzazione si fa uno studio più letterario, ci informiamo sul dominio, leggendo libri, articoli, riviste, blog, forum, social network.

Importante analizzare il sistema as is, prima dell'avvento del software, come sono documentati i workflow, le regole di business, report di lamentele...

SOno fonti interessanti per raccogliere i requisiti i problemi degli utenti. Molte aziende mantengono le lamentele dei clienti. Pone le basi per quando andremo a incontrare gli stakeholders. Sapremo la terminologia, i problemi, le opportunità, i vincoli, le regole di business. In modo da arrivare preparati.

L'attività può essere dispendiosa, alcune informazioni possono essere irrilevanti. Una soluzione è la meta-conoscenza per tagliare fonti. Studio solo quella che mi serve, capire quello che mi serve e andare a cercare solo quello.

2.1.1 Collezione di dati

Consiste nel raccogliere dati aziendali per capire numericamente di cosa sto parlando.Si tratta di informazioni quantitative, dati del mercato, statistiche, dati finanziari, dati di produzione, dati di vendita, dati di marketing. Come si colloca nel mercato di riferimento. Sono utili per i requisiti non funzionali, come performance, scalabilità, usabilità, affidabilità, sicurezza. Le difficoltà sono di raggiungere fonti di informazioni affidabili.

2.1.2 Questionario

Il questionario è un modo efficace per raccogliere informazioni dagli stakeholders. Tipicamente sono domande con una breve spiegazione, domande a risposta multipla, domande aperte, domande a scala. Efficace per informazione veloce, anche per persone che non sono in presenza. Si tratta di informazione soggettiva, utile per sgrezzare la nostra conoscenza.

è importante preparare correttamente questi questionari, è facile sbagliare. Potremmo far errore di suggerire la risposta e devono essere formulate in maniera neutrale senza inserire bias. Stare attenti a ambiguità, ci sono linea guida. Per avere anche un campione significativo. Aggiungere ridondanza nel questionario, chiedendo le cose in maniera diversa, in modo da avere conferme e verificare inconsistenze. Prima di somministrare i questionari si fanno validare da altri membri del team.

2.1.3 Card sort

Ci sono altre tecniche per informazione. Acquisire informazioni in più rispetto alle informazioni in più rispetto ai questionari. Si scrivono le info sulle carte. CHe vengono messe insieme

e si chiede il motivo per cui sono state messe insieme. In questo modo nascono dei requisti.

2.1.4 Repertory grid

Si tratta di una tecnica per acquisire informazioni sulle relazioni tra concetti. Si chiede di mettere insieme concetti simili e diversi, in modo da capire le relazioni tra i concetti. Gli utenti suggeriscono valori per i concetti, in modo da capire le relazioni tra i concetti.

Conceptual labeling: i concetti simili vengono raggruppati insieme, in modo da capire le relazioni tra i concetti dando un nome al gruppo. Rappresenta un livello di astrazione più elevato. Sono metodi semplici ed economici che permettono di acquisire informazioni che ci siamo persi precedentemente. Lo svantaggio è che potrebbero essere largamente soggettivi.

2.1.5 Scenari e storyboard

L'obiettivo è quello di acquisire informazione attraverso esempi concreti. Tipicamente sotto forma narrativa. Le storyboard sono una sequenza di immagini che rappresentano una storia, che racconta lo step nella vita di un utente. Possono essere figure o in forma scritta. Possono essere passive o attive. Passive sono per la validazione, attive sono per elaborare la storyboard o completarne.

Gli scenari sono rappresentazioni tipicamente testuali che rappresentano interazioni con il sistema che possono essere utilizzate per capire come il sistema è o per esplorare soluzioni. Sono usate per elicitare informazioni, chiedendo il perché. Sono necessari per l'acceptance test.

Gli scenari possono essere:

- Positivi.
- Negativi.
- Normali.
- Anomali: situazione eccezionale.

Pro e contro: Sono esempi concreti e controesempi del sistema. Sono di tipo narrativo e sono ben recepiti dagli stakeholders. Possono essere tradotti in test di accettazione.

Sono intrinsecamente parziali potrei dimenticare degli scenari. Potremmo avere una esplosione combinatoria in caso di copertura completa. Rischiamo si specificare troppo il sistema. Prendere decisioni premature dal software e l'environment. Decisioni irrilevanti per il sistema.

Gli scenari concreti prima o poi arrivano opportuno utilizzarli.

2.1.6 Prototipazione e mockup

L'obiettivo è di fatto di raccogliere il feedback immediato con lo schizzo del software i nazione. Il prototipo potrebbe avere feature incomplete potrebbe essere funzionale per validare funzionalità oppure per interfacce grafiche

In molti casi la produzione di prototipi va a pari passo con i requisiti. Un mock up è tipicamente un prodotto che realizzo e poi butto, un prototipo posso raffinarlo. Può formulare feedback immediato.

Come pro abbiamo che il feedback è simile al software finito, per verificare requisiti e vincoli. Sono utili per la validazione e la verifica. Anche per sorgere mancanze e errori. Sono utili per la comunicazione con gli stakeholders. Sono utili anche per la formazione agli utenti, in modo da capire come usare il software finale, o usati come stub per integration testing.

Gli svantaggi sono che potrebbero essere che non coprono tutti i requisiti. Mancano funzionalità e potrebbero non catturare i requisiti non funzionali. legati alle performance. Possono essere forvianti, aspettative troppo alte. Potrebbero essere non del tutto implementabili e potrebbero esserci delle inconsistenze con il software finale.

2.1.7 Knowledge reuse

Prendiamo la conoscenza su altre esperienze di cose simile per avere una elicitazione di requisiti più spedita. Tipicamente si individua la conoscenza simile, la si traspone e poi si valida il risultato e lo si integra. La trasposizione avviene in 3 modi:

- Istanziazione:
- Specializzazione:
- Riformulazione:

Le tassonomie sono utili per trovare la conoscenza simile. Indipendenti dal dominio. Si usano anche i meta-modelli per trovare la conoscenza simile, istanziando ad esempio oggetto, goal, agenti e operazioni. Posso usare Informazioni specifiche su un dominio astratto e associarlo al dominio concreto specializzando entità.

In che modo posso personalizzare la conoscenza? Lo stesso dominio astratto può avere specializzazioni multiple. Alcuni domini concreti possono specializzare alcuni domini astratti.

Pro e contro: L'analista esperto può usare in maniera efficiente la conoscenza e valociza la conoscenza. Si eredita la struttura e la qualità del dominio, è abbastanza efficace nel caso si voglia completare il documento dei requisiti.

Negativo è che possiamo utilizzare conoscenza stratta solo se è abbastanza vicino. Preparare i domini astratti è complesso, lavoro preparatorio è complesso e tipicamente si usa quello in letteratura. Effort non banale da fare.

2.2 Coinvolgimento degli stakeholders

Attraverso interviste. Organizzare meeting, registrare il contenuto di interviste, attraverso i trascritti. Devono essere elaborati, cosa abbiamo capito va sempre linkato e questo report va condiviso con l'intervistato in modo da verificare la corretta interpretazione.

Se coinvolgiamo i gruppi alcuni potrebbero essere più influenti di altri. Quindi perdiamo alcune informazione, si va più valoce. A volte si preferisce coinvolgere una a due persone per volta.

L'efficacia dell'intervista è misurabile.

Le interviste possono essere strutturate, alcune domande chiuse o aperte. è opportuno avere la lista di domande per avere un traccia per rimanere su binari opportuni. Oppure potrebbero essere non strutturate, abbiamo quindi solo una lista di argomenti da trattare, raggiungiamo ogni argomento di volta in volta. I punti di forza sono che: le info che raccogliamo potrebbero essere difficili da raccogliere in altri modi, possiamo avere un feedback immediato, possiamo raffinare la metodologia al volo, grazie alla formulazione possiamo modificare le domande successive, personalizzare le informazioni successive.

Gli svantaggi sono che potrebbero essere specifiche sulla persona, ci pososno essere inconsistenze. L'efficacia delle interviste dipende dalla competenza dell'intervistatore. Per migliorare abbiamo linea guida: Identificare le persone da coinvolgere, arrivare preparati e focalizzarci del giusto problema e del giusto momento. Focalizzare sul lavoro che L'intervistato fa. Non lasciare l'intervistato solo, mantenendo il controllo dell'intervista. L'intervistato in una situazione di comfort. Formulare all'inizio domande semplici, considerare anche aspetti personali essendo anche empatici, apparire affidabili e amichevoli. Essere focalizzati è importante, essere flessibili con risposte che ci sorprendono. Formulare domande sul perché e cercare di non essere offensivi. Domande con bias vanno evitate. Domande ovvie, o difficili da rispondere. Elaborare gli sbobinamenti appena dopo annotando con reazioni personali. che possono essere utili per la comprensione. Fargli avere gli elaborati per verificare la correttezza.

2.2.1 Etnografia

Osservare persone quando lavorano, system as is senza intrgire direttamente con loro. Passiva da parte dell'osservatore. Oppure coinvolto diventando anche membro del team.

I pro e i contro: Questo tipo di indagine danno conoscenza implicita difficile da cogliere. Rivelare modi articolati e complessi e permette di avere aspetti specifici della cultura aziendale quando raccogliamo i requisiti e riusciamo a contestualizzare i requisiti.

Lo svantaggio è che può essere lungo da fare, costa molto. Potrebbe essere inaccurato, sapendo di essere osservati potrebbero comportarsi in modo diverso. Sarebbe interessante vedere errori. Ci si concentra sul sistema attuale.

2.2.2 Sessioni di gruppo

Sessioni in cui si discute in gruppo dei requisiti in maniera strutturata. Opportuno che sia gestito con sessioni strutturate, con il moderatore. Focus grop con temi specifici

Le sessioni possono essere meno strutturate, con brainstorming. Si discute a ruota libera e si raccoglie tutto. Tipicamente con due fasi, nella prima ognuno lancia idee che vengono in mente a ruota libera, offrendo anche spunto dalle idee date. Nella seconda fare si converge o di prioritizzare le idee.

Pro e contro: SOno meno formali delle interviste quindi. Possono nascere delle sinnergie tra utenti per risolvere problemi. La composizione del gruppo è critico, a volte molti sono impegnati. è importante avere un moderatore con alte skill. A volte emergono persone che dominano la discussione, divergendo quindi la discussione su loro idee. Il rischio è che si perda il focus, in questo caso si rischia di perdere tempo. Aspetti superficiali non rilevanti.

2.2.3 Combinare le tecniche

Di solito di combinano tali tecniche con un mix compensando gli svantaggi di alcune tecniche con i vantaggi di altre.

Capitolo 3

Specifica dei requisiti e documentazione

Potrei avere errori con il linguaggio naturale, perché è intrinsescamente ambiguo. Ci sono linea guida per scrivere requisiti, riducendo l'ambiguità per raggiungere una certa completezza. Abbiamo delle guide stilistiche per scrivere i requisiti.

- Identificare a chi è rivolto il requisito.
- Brevi abstract per ogni requisito, per capire subito di cosa si tratta.
- Le cose vanno motivate prima di spiegare il requisito.
- Prima di usare un concetto è bene definirlo.
- Chiedersi se è sufficiente, comprensibile, rilevante.
- Non cercare di spiegare troppo in una frase, meglio scrivere più frasi per spiegare meglio.
- Usare "deve" e "non deve" per esprimere i requisiti o "dovrebbe" e "non dovrebbe" per esprimere i desiderata.
- Evitare gergo o acronimi.
- Usare esempi per spiegare meglio.
- Usare diagrammi per spiegare meglio relazioni complesse.

3.1 Regole locali

Con condizioni complesse è bene usare una tabella di verità per spiegare meglio il concetto. In questo caso l'interpretazione di `AND` e `OR` è ben definita. Il problema è che le combinazioni possono essere molte e quindi la tabella può diventare molto grande. In questo caso sarà possibile collassare degli elementi. Con una tabella di questo tipo è possibile avere i test di accettazione da far girare sul sistema non appena sarà disponibile.

3.2 Formare le frasi

- Requisiti enumerati e gerarchici.
- Requisiti divisi per categorie.
- Seguire regole stilistiche.
- Regole di fit.
- Tracciare le sorgenti dei requisiti.
- Motivo.
- Interazioni di conflitto o dipendenza.
- priorità.

Importante avere un criterio di fit per capire se la specifica è soddisfatta o meno dal sistema. Requisiti quantitativi attraverso la soglia percentuale. Anche attraverso interviste successive.

È opportuno ordinare i requisiti per sezioni, in modo da avere requisiti comuni vicini.

3.3 Diagrammi

Un altro modo per raccogliere i requisiti è attraverso i diagrammi. Che sono però semi-formali.

3.3.1 Context diagram

Si tratta di un grafo che contiene i componenti del sistema (*vertici*) e le relazioni tra i componenti (*archi*). Ad esempio il guidatore e il sistema di frenata sono due vertici, mentre l'arco tra i due rappresenta il fatto che il guidatore può azionare il sistema di frenata.

3.3.2 Problem diagram

È simile al context diagram, ma è più dettagliato, perché contiene chi fa cosa. Contiene anche notazioni in linguaggio naturale. Ragiono i ntermini del problema e quali requisiti fanno riferimento a quali compinenti del sistema.

3.3.3 Frame diagram

Questo diagramma è più informativo perché permette di tipare i componenti e i fenomeni. I componenti possono essere: Causali, Evento, Simbolico. I fenomeni possono essere: Causale, Biddable, Lessicale. Si va nella direzione del riutilizzo della conoscenza.

Ci sono altri diagrammi per ragionare in termini di strutturazione del sistema.

3.3.4 Entity-Relationship diagram

Diagramma visto anche in basi di dati. è possibile specializzare le classi per aggiungere più proprietà.

Ci sono diagrammi che permettono di parlare del sistema

3.3.5 SADT