Ingegneria del Software

Processi di Ingegneria dei requisiti

Antonino Staiano
e-mail: **antonino.staiano@uniparthenope.it**

Ingegneria dei requisiti

- Un requisito è una caratteristica che il sistema deve avere o un vincolo che deve soddisfare perché sia accettato dal cliente
- L'ingegneria dei requisiti ha l'obiettivo di definire i requisiti del sistema da costruire
  - Si compone di due principali attività
    - Scoperta dei requisiti: che fornisce la specifica del sistema comprensibile per il cliente
    - Analisi: che fornisce un modello di analisi che gli sviluppatori possono interpretare in modo non ambiguo

Requisiti e stakeholder

- La raccolta dei requisiti è l'attività più difficile, perché richiede la collaborazione tra più gruppi di partecipanti con differenti background
  - **Stakeholders:** tutte le persone in qualche modo interessate alla messa in opera del sistema
  - Il cliente e gli utenti finali sono esperti nel loro dominio e hanno una idea generale (e in molti casi vaga) di cosa il sistema debba fare, e poca esperienza nello sviluppo del software
  - Gli sviluppatori hanno esperienza nel produrre sistemi software, ma hanno una conoscenza limitata del dominio di applicazione (ambiente degli utenti finali)

Errori nella raccolta dei requisiti

- Tutti gli stakeholder comunicano tra di loro per definire il sistema da realizzare
  - Una cattiva comunicazione (tra i diversi domini) porta a un sistema difficile da usare o che non supporta le funzionalità richieste
- Gli errori introdotti in questa fase sono difficili (e costosi) da risolvere perché sono scoperti nelle ultime fasi del processo di sviluppo del software
  - Errori possibili:
    - una funzionalità che il sistema dovrebbe supportare non è specificata
    - funzionalità incorrette o obsolete
    - interfacce utenti poco intuitive e difficili da usare

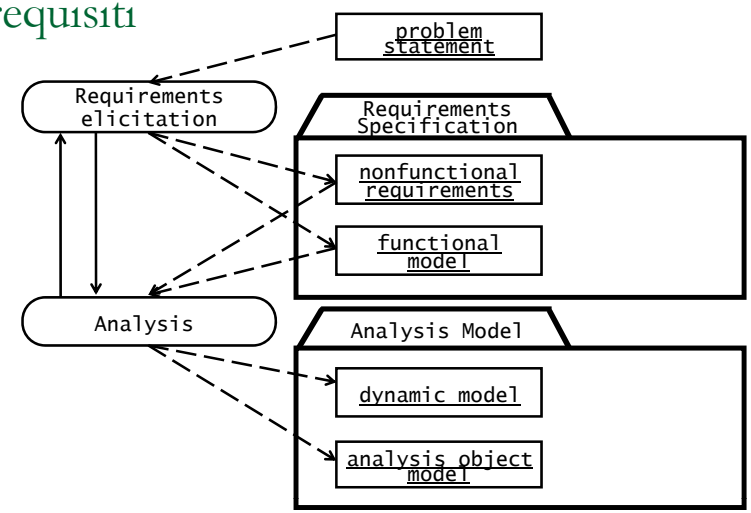
Strumenti per l'ingegneria dei requisiti

- Scenari e casi d'uso forniscono gli strumenti per colmare questo vuoto
  - Uno scenario descrive un esempio del sistema in uso in termini di una serie di interazioni tra l'utente e il sistema
  - Un caso d'uso è un'astrazione che descrive una classe di scenari
  - Entrambi sono scritti in linguaggio naturale, una forma comprensibile all'utente
- Non appena la definizione del sistema matura e si stabilizza, gli sviluppatori ed il cliente convergono su di una specifica del sistema nella forma di requisiti funzionali, non funzionali, casi d'uso e scenari

Panoramica sulla scoperta dei requisiti

- La scoperta dei requisiti si focalizza sulla descrizione dello scopo del sistema
- Il cliente, gli sviluppatori, e gli utenti identificano un'area del problema e definiscono un sistema che affronta il problema
- La descrizione risultante è la specifica dei requisiti che funge da contratto tra il cliente e gli sviluppatori
- La specifica dei requisiti viene strutturata e formalizzata durante l'analisi per ottenere il modello di analisi

Prodotti della scoperta ed analisi dei requisiti



Punto di vista della scoperta ed analisi dei requisiti

- La scoperta e l'analisi dei requisiti si concentrano solo sul punto di vista dell'utente del sistema
  - Fanno parte dei requisiti, ad esempio:
    - Le funzionalità del sistema
    - L'interazione tra utente e sistema
    - Errori che il sistema può individuare e la loro gestione
    - Le condizioni ambientali in cui opera il sistema
  - Non fanno parte dei requisiti
    - La struttura del sistema
    - Le tecnologie implementative selezionate per la costruzione del sistema
    - La progettazione del sistema
    - La metodologia di sviluppo

Attività della scoperta dei requisiti (I)

- Identificazione degli attori
  - Gli sviluppatori individuano i differenti tipi di utenti che il sistema supporterà
- Identificazione degli scenari
  - Gli sviluppatori osservano gli utenti e sviluppano un insieme di scenari dettagliati delle funzionalità tipiche che il sistema supporterà. Gli sviluppatori usano gli scenari per comunicare con gli utenti e comprendere in modo approfondito il dominio dell'applicazione
- Identificazione dei casi d'uso
  - Raggiunto un accordo su un insieme di scenari con gli utenti, gli sviluppatori derivano un insieme di casi d'uso che rappresentano completamente il sistema futuro

Attività della scoperta dei requisiti (II)

- Revisione dei casi d'uso
  - Gli sviluppatori verificano che la specifica dei requisiti sia completa dettagliando ogni caso d'uso e descrivendo il comportamento del sistema in presenza di errori e condizioni eccezionali
- Identificazione delle relazione tra i casi d'uso
  - Gli sviluppatori identificano le dipendenze tra i casi d'uso. Consolidano anche il modello dei casi d'uso esplicitando le funzionalità comuni
- Identificazione dei requisiti non funzionali
  - Sviluppatori, utenti e clienti concordano sugli aspetti visibili all'utente ma non direttamente legati alla funzionalità (vincoli sulle prestazioni, documentazione, risorse che consuma, sicurezza e qualità)

CONCETTI DELLA SCOPERTA DEI REQUISITI

Requisiti funzionali

- Descrivono le interazioni tra il sistema ed il suo ambiente indipendentemente dall'implementazione
- L'ambiente include l'utente e qualsiasi altro sistema esterno con cui il sistema interagisce

Requisiti funzionali per SatWatch

- SatWatch è un orologio da polso che visualizza l'ora in base della località corrente. SatWatch usa il GPS per determinare la propria locazione e strutture dati interne per convertire questa ubicazione in un fuso orario.
- Le informazioni memorizzate in SatWatch e la sua precisione sono tali per cui il proprietario non ha mai la necessità di resettare l'ora. SatWatch regola l'ora e la data visualizzata ogniqualvolta il proprietario attraversa fusi orari o confini politici. Per questa ragione, SatWatch non ha pulsanti o controlli per l'utente.
- SatWatch determina la posizione usando un satellite GPS e, come tale, soffre delle stesse limitazioni di tutti i dispositivi GPS (ad esempio, incapacità di determinare la posizione in certi orari del giorno in regioni montagnose). Durante i momenti di black-out, SatWatch assume di non attraversare fusi orari o confini politici. SatWatch corregge il proprio fuso orario non appena termina il momento di black-out

Requisiti funzionali per SatWatch (II)

- SatWatch ha un display a due linee che mostra, sulla linea in alto, l'ora (ore, minuti, secondi, fuso orario) e sulla linee in basso, la data (giorno, mese, anno). La tecnologia del display usata è tale che il proprietario dell'orologio può guardare l'ora e la data anche al buio.
- Quando cambiano i confini politici, il proprietario dell'orologio può aggiornare il software dell'orologio usando il dispositivo WebifyWatch (fornito con l'orologio) e un pc connesso ad Internet

Requisiti non funzionali

- I requisiti non funzionali descrivono aspetti del sistema non direttamente legati al comportamento funzionale del sistema
- I requisiti non funzionali includono un'ampia gamma di requisiti che si applicano a differenti aspetti del sistema
- Il modello FURPS+ (Functional, Usability, Reliability, Perfomance, Supportability) usato dal Processo Unificato fornisce le seguenti categorie di requisiti non funzionali

Requisiti (di qualità) non funzionali FURPS+

- Usabilità
  - facilità per l'utente di imparare ad usare il sistema, e capire il suo funzionamento. Includono:
    - convenzioni adottate per le interfacce utenti
    - portata dell'Help in linea
    - livello della documentazione utente
- Affidabilità
  - capacità di un sistema o di una componente di fornire la funzione richiesta sotto certe condizioni e per un periodo di tempo. Includono:
    - un tempo medio di fallimento accettabile
    - l'abilità di scoprire specificati difetti
    - sostenere specifici attacchi alla sicurezza

Requisiti (di qualità) non funzionali FURPS+

- Prestazioni
  - riguardano attributi quantificabili del sistema come:
    - tempo di risposta, quanto velocemente il sistema reagisce a input utenti
    - throughput, quanto lavoro il sistema riesce a realizzare entro un tempo specificato
    - disponibilità, il grado di accessibilità di un componente o del sistema quando richiesti
- Supportabilità
  - Riguardano la semplicità di fare modifiche dopo il deployment. Includono
    - adattabilità, l'abilità di cambiare il sistema per trattare concetti aggiuntivi del dominio di applicazione
    - mantenibilità, l'abilità di cambiare il sistema per trattare nuove tecnologie e per far fronte a difetti

Pseudo requisiti (o vincoli) non funzionali FURPS+

- Chiamati anche vincoli, sono:
  - Implementazione
    - vincoli sull'implementazione del sistema, incluso l'uso di tool specifici, linguaggi di programmazione, o piattaforme hardware
  - Interfacce
    - vincoli imposti da sistemi esterni, incluso sistemi legacy e formati di scambio
  - Operazioni
    - vincoli sull'amministrazione e sulla gestione del sistema
  - Packaging
    - vincoli sulla consegna reale del sistema
  - Legali
    - riguardano licenze, regolamentazioni, e certificazioni

Requisiti di qualità per *SatWatch*

- Un utente che sa come leggere un orologio digitale e capire le abbreviazioni del fuso orario internazionale dovrebbe essere in grado di usare *SatWatch* senza il manuale utente [Usabilità]
- Poiché *SatWatch* non ha pulsanti, non dovrebbero verificarsi errori software che richiedono il reset del'orologio [Affidabilità]
- *SatWatch* dovrebbe visualizzare il fuso orario corretto dopo al più 5 minuti dal termine del periodo di black-out del GPS [Prestazioni]
- *SatWatch* dovrebbe visualizzare correttamente l'ora in tutti i 24 fusi [Prestazioni]
- *SatWatch* dovrebbe accettare aggiornamenti via interfaccia seriale Webify Watch [Supportabilità]

Vincoli per *SatWatch*

- Tutti i software associati con *SatWatch,* incluso il software proprietario, sarà scritto usando Java, per conformarsi con la corrente politica della compagnia [Implementazione]
- *SatWatch* è conforme con le interfacce fisiche, elettriche e software definite dalle API Webify Watch 2.0 [Interfaccia]

Validazione dei requisiti

- E' un passo critico nel processo di sviluppo
- I requisiti sono continuamente validati da cliente e utenti
  - Di solito dopo l'analisi dei requisiti
- La validazione dei requisiti richiede di controllare
  - Correttezza: una specifica è corretta se rappresenta accuratamente il sistema che il cliente richiede e che gli sviluppatori intendono sviluppare
  - Completezza: una specifica è completa se tutti i possibili scenari per il sistema sono descritti, incluso i comportamenti eccezionali
  - Coerenza: se i requisiti non si contraddicono tra di loro
  - Chiarezza: una specifica è chiara se non è possibile interpretare la specifica in due modi diversi

Criteri per la validazione dei requisiti

- Realismo:
  - La specifica dei requisiti è realistica se può essere implementata tenendo conto dei vincoli
  - Verificabilità:
    - La specifica dei requisiti è verificabile se, una volta che il sistema è stato costruito, possono essere delineati test ripetuti per dimostrare che il sistema soddisfa i requisiti
      - Esempio di non verificabilità: Il prodotto dovrebbe avere una buona interfaccia – Buono non definito
- Tracciabilità:
  - Ogni funzione del sistema può essere individuata e ricondotta al corrispondente requisito funzionale
  - Include anche l'abilità di tracciare le dipendenze tra i requisiti, le funzioni del sistema, gli artefatti, incluso componenti, classi, metodi e attributi di oggetti
  - È cruciale per lo sviluppo di test e per valutare i cambiamenti

Tipi di raccolta dei requisiti

- Classificati sulla base della sorgente dei requisiti:
- Greenfield Engineering
  - Lo sviluppo parte da zero, nessun sistema esiste in precedenza, così i requisiti sono "estratti" dagli utenti e dal cliente
  - E' guidata dalla necessità dell'utente o dalle esigenze del mercato
- Re-engineering
  - Un sistema esistente viene riprogettato a causa della disponibilità di nuove tecnologie o per estendere funzionalità del sistema
- Interface Engineering
  - Per fornire i servizi di un sistema esistente in un nuovo ambiente
  - Vengono riprogettate le interfacce
  - Sistemi legacy lasciati inalterati, eccetto che per le interfacce
  - È guidata dall'introduzione di nuove tecnologie, e da nuove necessità del mercato

ATTIVITÀ DI SCOPERTA DEI REQUISITI

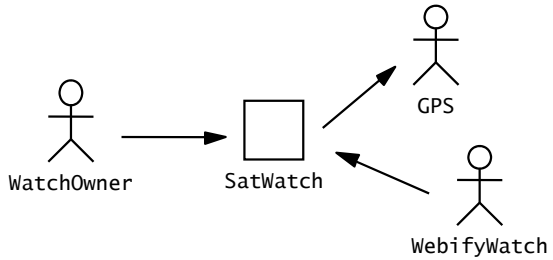
Attività di scoperta dei requisiti

- Le attività di scoperta dei requisiti fanno corrispondere la definizione di un problema ad una specifica dei requisiti, che può essere presentata come un insieme di
  - Attori
  - Scenari
  - Casi d'uso
- Le attività di scoperta dei requisiti includono
  - Identificazione degli attori
  - Identificazione degli scenari
  - Identificazione dei casi d'uso
  - Identificazione delle relazioni tra attori e casi d'uso
  - Identificazione degli oggetti iniziali dell'analisi
  - Identificazione dei requisiti non funzionali

Identificazione degli attori

- Gli attori rappresentano entità esterne che interagiscono con il sistema
  - Può essere un utente o un sistema esterno
- Nell'esempio *SatWatch*, il proprietario dell'orologio, il satellite GPS e il dispositivo seriale Webify Watch sono attori
  - Tutti scambiano informazioni con *SatWatch*
  - E' da osservare che tutti hanno interazioni specifiche con *SatWatch*:
    - Il proprietario porta e guarda l'orologio
    - L'orologio monitora il segnale dal satellite GPS
    - Webify Watch scarica nuovi dati nell'orologio

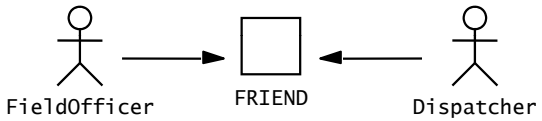
Attori per il sistema `SatWatch`. `WatchOwner` sposta l'orologio (eventualmente attraverso diversi fusi) e lo consulta per conoscere l'ora. `SatWatch` interagisce con il *GPS* per calcolare la posizione. `WebifyWatch` aggiorna i dati contenuti nell'orologio per riflettere le modifiche dell'orario



Identificazione degli attori in FRIEND

- Consideriamo l'esempio del sistema FRIEND, un sistema informativo distribuito per la gestione degli incidenti
- Include molti attori
  - FieldOfficer, gli ufficiali di polizia e dei vigili del fuoco che rispondono ad un incidente
  - Dispatcher, l'ufficiale di polizia responsabile di rispondere alle chiamate del 911 e di allocare le risorse ad un incidente
- FRIEND supporta ambo gli attori tenendo traccia degli incidenti, delle risorse e dei piani delle attività
  - Inoltre, supporta l'accesso a database multipli, come db dei materiali nocivi e delle procedure delle operazioni di emergenza
  - Gli attori FieldOfficer e Dispatcher interagiscono attraverso varie interfacce
    - FieldOfficer accedono a FRIEND attraverso un laptop
    - Dispatcher accedono a FRIEND attraverso una workstation

Attori del sistema *FRIEND*. I *FieldOfficer* non solo hanno accesso a differenti funzionalità, ma usano anche differenti computer per accedere al sistema



Identificazione degli attori

- Attori e ruoli sono astrazioni e non necessariamente corrispondono a persone
  - La stessa persona può ricoprire il ruolo di FieldOfficer e Dispatcher in tempi diversi
  - Le funzionalità a cui accedono sono sostanzialmente differenti
    - Quindi, i due attori sono modellati con due attori differenti

Scoperta dei requisiti e identificazione attori

- Il primo passo nella scoperta dei requisiti è l'identificazione degli attori
  - Serve per definire i confini del sistema e per trovare tutte le prospettive da cui gli sviluppatori necessitano di considerare il sistema
- Quando il sistema è inserito in un'organizzazione esistente, molti attori solitamente esistono prima che il sistema si sviluppato: essi corrispondono a ruoli nell'organizzazione

Scoperta dei requisiti e identificazione attori

- Durante la fase iniziale dell'identificazione degli attori, è difficile distinguere gli attori dagli oggetti
  - Ad esempio, un sottosistema di DB può essere un attore in un frangente e parte del sistema in un altro
- Quando il confine del sistema viene definito, non ci sono più problemi nella distinzione
  - Gli attori sono fuori dal confine del sistema, sono esterni
  - I sottosistemi e gli oggetti sono all'interno del confine del sistema, sono interni
  - Ogni sistema software esterno che usa il sistema da sviluppare è un attore

Domande per identificare gli attori

- Quali gruppi di utenti sono supportati dal sistema per eseguire il proprio lavoro?
- Quali gruppi di utenti eseguono le funzioni principali del sistema?
- Quali gruppi di utenti eseguono le funzioni secondarie, come manutenzione e amministrazione?
- Con quali sistemi software e hardware esterni interagirà il sistema?

Attori in FRIEND

- Nell'esempio FRIEND, queste domande hanno portato ad una lunga lista di potenziali attori:
  - Vigile del fuoco, ufficiale di polizia, dispatcher, investigatore, sindaco, governatore, db per materiali pericolosi, amministratore di sistema ecc.
  - E' necessario poi consolidare questa lista in un piccolo numero di attori, che sono differenti dal punto di vista dell'uso del sistema
    - Ad esempio, un vigile del fuoco ed un ufficiale di polizia possono condividere la stessa interfaccia al sistema, poiché sono entrambi coinvolti con un singolo incidente
    - Un Dispatcher, gestisce più incidenti concorrenti e richiede l'accesso ad un maggiore volume di informazioni
    - Il governatore ed il sindaco non interagiscono direttamente col sistema ma usufruiscono dei servizi di un operatore addestrato

Identificazione degli scenari

- Uno scenario è una descrizione concreta, mirata e informale di una singola funzionalità del sistema dal punto di vista di un singolo attore
- Gli scenari non possono (e non sono concepiti per) sostituire i casi d'uso, in quanto mirano su specifiche istanze ed eventi concreti
- Gli scenari, comunque, potenziano la scoperta dei requisiti fornendo uno strumento comprensibile agli utenti ed ai clienti

Nome Scenario	warehouseOnFire
Istanze attori partecipanti	bob, alice: FieldOfficer john: Dispatcher
Flusso di eventi	1. Bob, guidando lungo la strada principale nella sua patrol, osserva del fumo proveniente da un magazzino. Il suo partner, Alice, attiva la funzione "Report Emergency" dal laptotp del FRIEND. 2. Alice immette l'indirizzo dell'edificio, una breve descrizione della sua ubicazione (angolo nord-ovest) ed un livello di emergenza. In aggiunta ad un'unità dei vigili del fuoco, richiede diverse unità paramediche poiché la zona non è molto trafficata. Conferma l'input ed attende l'accettazione. 3. John, il Dispatcher, è allertato per l'emergenza da un beep dalla sua stazione di lavoro. Rivede le informazioni sottomesse da Alice e accetta il rapporto. Alloca un'unità dei vigili del fuoco e due unità paramediche sul sito dell'Incidente ed invia la stima del tempo di arrivo (ETA) ad Alice. 4. Alice riceve l'accettazione e l'ETA.

Tipologia di scenari (I)

- Gli scenari possono avere molti usi differenti durante la scoperta dei requisiti e durante altre attività del ciclo di vita:
  - Scenari as-is
    - descrivono una situazione corrente
  - Scenari visionari
    - descrivono un sistema futuro. Sono usati come un punto nello spazio di modellazione dagli sviluppatori per rifinire le idee sul sistema futuro e come mezzo di comunicazione per scoprire i requisiti dagli utenti

Tipologia di scenari (II)

- Scenari valutativi
  - Descrivono i compiti utente rispetto ai quali il sistema deve essere valutato. Lo sviluppo collaborativo degli scenari valutativi da parte di utenti e sviluppatori migliora anche la definizione delle funzionalità testate da questi scenari
- Scenari di addestramento
  - sono tutorial usati per introdurre i nuovi utenti al sistema. Sono istruzioni *passo per passo* progettate per aiutare l'utente attraverso i compiti comuni

Stesura degli scenari

- Nella scoperta dei requisiti, sviluppatori e utenti scrivono e rifiniscono una serie di scenari con lo scopo di migliorare la comprensione condivisa di cosa il sistema dovrebbe essere
- Inizialmente, ogni scenario può essere ad alto livello ed incompleto come nel caso del *warehouseOnFire*

Domande per identificare gli scenari

- Quali sono i task che l'attore vuole che il sistema esegua?
- A quali informazioni l'attore accede? Chi crea i dati? Possono essere modificati o rimossi? Da chi?
- Quali modifiche esterne l'attore deve notificare al sistema? Quanto spesso? Quando?
- Di quali eventi il sistema deve informare l'attore? Con quale latenza?

Sorgenti di informazioni per gli scenari

- Gli sviluppatori usano i documenti esistenti sul dominio applicativo per rispondere a tali questioni
- Questi documenti includono manuali utente di sistemi precedenti, manuali delle procedure, standard della compagnia, note utente, interviste con utenti e clienti
- Gli sviluppatori dovrebbero sempre scrivere gli scenari usando la terminologia del dominio applicativo

FRIEND e scenari

- Nel sistema FRIEND, identifichiamo quattro scenari che abbracciano i tipi di task che ci si aspetta il sistema supporti
  - *warehouseOnFire*: E' individuato un incendio in un magazzino; due ufficiali arrivano sulla scena e richiedono le risorse
  - *fenderBender*: si verifica un incidente automobilistico sull'autostrada senza vittime. Gli ufficiali di polizia documentano l'incidente e gestiscono il traffico mentre i veicoli danneggiati sono portati via dal carro attrezzi
  - *catInATree*: un gatto è bloccato su un albero. Un automezzo dei vigile del fuoco è richiamato per recuperare il gatto. Poiché l'incidente ha una bassa priorità, il mezzo impiega del tempo per arrivare sul luogo. Nel frattempo, il proprietario del gatto, impaziente, si arrampica sull'albero, cade e si spezza una gamba, richiedendo l'intervento di un'ambulanza
  - *earthQuake*: un terremoto senza precedenti danneggia seriamente edifici e strade, provocando incidenti multipli e abilitando l'attivazione di un piano di emergenza nazionale. Il governatore è informato. Le strade danneggiate ostacolano i soccorsi

Identificazione dei casi d'uso

- Uno scenario è un'istanza di un caso d'uso
  - Un caso d'uso specifica tutti i possibili scenari per una data parte di funzionalità
- Un caso d'uso è iniziato da un attore. Dopo l'iniziazione, un caso d'uso può interagire anche con altri attori
- Un caso d'uso rappresenta un flusso di eventi completo attraverso il sistema
  - Descrive una serie di interazioni collegate che sono il risultato dell'iniziazione

Nome	ReportEmergency
Attori Partecipanti	Iniziato dal FieldOfficer Comunica con il Dispatcher
Flusso eventi:	1. Il FieldOfficer attiva la funzione "Report Emergency del terminale 2. FRIEND risponde presentando una form al FieldOfficer 3. Il FieldOfficer riempie la form selezionando il livello di emergenza, tipo, località, breve descrizione della situazione. Il FieldOfficer descrive anche possibili risposte alla situazione di emergenza. Appena finito Il FieldOfficer invia la form 4. FRIEND riceve la form e notifica al Dispatcher 5. Il Dispatcher rivede le informazioni sottomesse e crea un Incidente nel DB invocando il caso d'uso OpenIncident. Il Dispatcher seleziona una risposta e accetta il rapporto. 6. FRIEND visualizza l'accettazione e la risposta selezionata al FieldOfficer
Condizioni di entrata	Il FieldOfficer è loggato in FRIEND
Condizioni di uscita	Il FieldOfficer ha ricevuto un'accettazione e una risposta selezionata dal Dispatcher, OR Il FieldOfficer ha ricevuto una spiegazione indicante perché la transazione non è stata eseguita
Requisiti di qualità	Il rapporto del FieldOfficer è accettato entro 30 secondi La risposta selezionata arriva non più tardi di 30 secondi dopo che è stata inviata dal Dispatcher

Organizzare i casi d'uso

- Generalizzare gli scenari e identificare i casi d'uso di alto livello che il sistema deve supportare consente agli sviluppatori di definire lo scopo del sistema
- Inizialmente, gli sviluppatori danno i nomi ai casi d'uso, li allegano agli attori che li iniziano e forniscono una descrizione ad alto livello del caso d'uso
- Il nome di un caso d'uso dovrebbe essere una frase verbale che denota l'attore cosa sta cercando di realizzare
  - **ReportEmergency** indica che un attore sta cercando di fornire una rapporto di emergenza al sistema
  - Non è chiamato **RecordEmergency** poiché il nome dovrebbe riflettere la prospettiva dell'attore, non del sistema
  - Non è nemmeno chiamato **AttemptToReportanEmergency** poiché il nome dovrebbe riflettere l'obiettivo del caso d'uso e non l'attività corrente

Descrizione dei casi d'uso

- Allegare i casi d'uso agli attori che li iniziano consente agli sviluppatori di chiarire i ruoli dei diversi utenti
  - Spesso, focalizzandosi su chi inizia ogni caso d'uso, gli sviluppatori identificano nuovi attori che precedentemente sono sfuggiti
- La descrizione di un caso d'uso comporta la specifica di quattro campi:
  - Descrizione delle condizioni di entrata e di uscita
  - Descrizione del flusso di eventi
  - Descrizione dei requisiti di qualità

Esempio: Allocate a Resource

- Attori:
  - *Field Supervisor:* E' l'ufficiale sul luogo dell'emergenza
  - *Resource Allocator:* E' responsabile per l'impegno ed il disimpegno delle Risorse gestite dal sistema FRIEND
  - *Dispatcher:* Immette, aggiorna, e rimuove Incidenti, Azioni, e Richieste nel sistema. Provvede anche a chiudere le pratiche sugli incidenti
  - *Field Officer:* Produce i rapporti sul lugo dell'emergenza

Esempio: Allocate a Resource

- *Nome caso d'uso:* AllocateResources
- *Attori partecipanti:*
  - Field Officer (Bob e Alice nello Scenario)
  - Dispatcher (John nello Scenario)
  - Resource Allocator
  - Field Supervisor
- *Condizione di ingresso*
  - Il Resource Allocator ha selezionato una risorsa disponibile
  - La risorsa correntemente non è allocata
- *Flusso eventi*
  - Il Resource Allocator seleziona un incidente
  - La risorsa è impegnata sull'incidente
- *Condizione di uscita*
  - Il caso d'uso termina quando ls risorsa è impegnata
  - La risorsa selezionata ora è non disponibile per altri incidenti o richieste di risorsa
- *Requisiti speciali*
  - Il Field Supervisor è responsabile della gestione delle risorse

Guida alla scrittura dei casi d'uso

- Scrivere i casi d'uso è un'arte. Un analista impara con l'esperienza
- Analisti diversi tendono a sviluppare stili diversi
  - Risulta difficile produrre una specifica dei requisiti consistente
- E' possibile usare una guida alla stesura dei casi d'uso

Guida alla stesura dei casi d'uso (I)

- *I casi d'uso dovrebbero essere definiti con frasi verbali. Il nome del caso d'uso dovrebbe indicare cosa sta cercando di realizzare l'utente (ReportEmergency, OpenIncident, ecc.)*
- *Gli attori dovrebbero essere nominati con dei sostantivi (FieldOfficer, Dispatcher,ecc.)*
- *Il confine del sistema dovrebbe essere chiaro. I passi realizzati dall'attore ed i passi realizzati dal sistema dovrebbero essere chiaramente distinguibili*
- *I passi dei casi d'uso nel flusso di eventi dovrebbero essere descritti con voce attiva. Ciò rende esplicito chi esegue il passo*
- *La relazione di causalità tra i passi successivi dovrebbe essere chiara*

Guida alla stesura dei casi d'uso (II)

- *Un caso d'uso dovrebbe descrivere una transazione utente completa (ad esempio, il caso d'uso ReportEmergency descrive tutti i passi tra l'avvio del rapporto di emergenza e la ricezione della notifica avvenuta)*
- *Le eccezioni dovrebbero essere descritte separatamente*
- *Un caso d'uso non dovrebbe descrivere l'interfaccia utente del sistema. Ciò distoglie l'attenzione dai passi effettivi realizzati dall'utente*
- *Un caso d'uso non dovrebbe superare una lunghezza di due o tre pagine. Altrimenti, si devono usare le relazioni include e extend per decomporlo in casi d'uso più piccoli*