

6. PROGETTAZIONE DI SISTEMI INFORMATIVI BASATI SU WEB.....	2
6.1 LA PROGETTAZIONE COME PROCESSO	2
6.1.1 Sviluppo ad hoc	2
6.1.2 Il ciclo di vita dello sviluppo di un sistema	2
6.1.3 Sviluppo di un sistema informativo basato su web.....	4
6.2 PIANIFICAZIONE.....	6
6.2.1 Descrizione dettagliata delle fasi	6
6.2.2 Analisi dell'azienda e del problema	7
6.2.3 Analisi dei dominio	8
6.2.4 Documento di vision.....	8
6.2.5 Definizione del piano di progetto	9
6.2.6 Documenti prodotti nella fase di pianificazione.....	9
6.3 RACCOLTA E ANALISI DEI REQUISITI	10
6.3.1 Raccolta dei requisiti	10
6.3.2 Specifica dei requisiti	14
6.3.3 Casi d'uso	14
6.3.4 Esperienza dell'utente	21

6. Progettazione di Sistemi Informativi basati su Web

6.1 La progettazione come processo

6.1.1 Sviluppo ad hoc

Spesso nella realizzazione di un sito web, la progettazione assume un carattere non sistematico. Questo approccio è tipico nei casi in cui si vuole creare un sistema basato su web in tempi molto rapidi, ma può essere pericoloso in quanto la costruzione del sito procederà per aggiustamenti successivi sulla base di richieste degli utenti o dei committenti (fig. 6.1) e può risultare pericoloso se si vuole realizzare un'interazione con i sistemi informativi aziendali controllata, sicura e con una gestione coerente delle applicazioni.



Figura 6.1: Sviluppo ad hoc

Ovviamente tale approccio può andare bene per siti personali in cui le informazioni e i servizi non vengono gestiti da più persone e non sono condivise da più applicazioni. Problemi che si possono creare in sistemi pianificati e progettati in modo non sistematico vanno dal rischio di non disponibilità del sistema dovuto a interruzioni impreviste (crash), all'impossibilità di mantenere il sito, all'incapacità di soddisfare nuove esigenze o di gestire la crescita (scalabilità). Errori di contenuto tipici che ne derivano sono i seguenti: contenuto non appropriato, pubblicato al momento sbagliato, inaccurato, inconsistente, pubblicazione di informazioni confidenziali, non aggiornato, non autorizzato, con errori, errori di versioni. Ciò è tanto più grave in quanto l'informazione viene resa disponibile immediatamente: tali errori sono quindi visibili e a tutto il mondo e possono avere conseguenze anche gravi.

6.1.2 Il ciclo di vita dello sviluppo di un sistema

Per gestire la complessità di tutti gli aspetti legati alla fornitura di servizi informativi tramite web la soluzione è quella di seguire un processo di sviluppo sistematico. La complessità de sistema viene sintetizzata nella fig. 6.2.

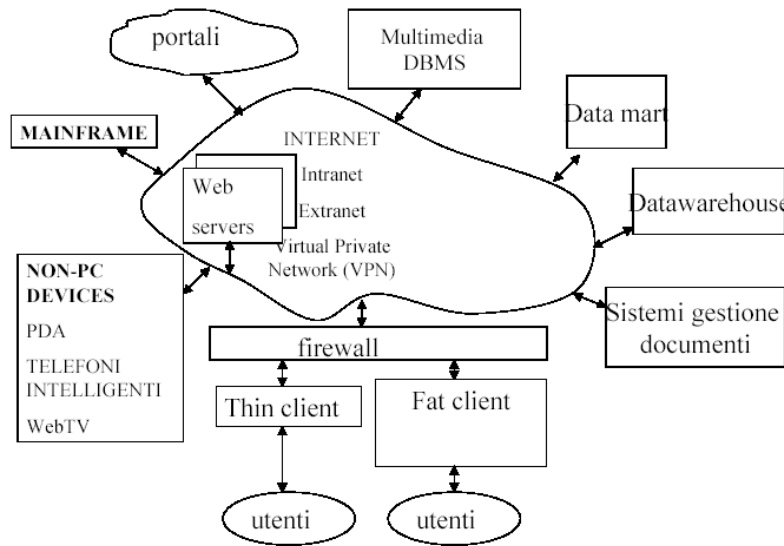


Figura 6.2: Fattori di complessità nella progettazione dei WIS

L'obiettivo è dunque quello di fornire un'indicazione sull'ordine delle attività, specificare quali elaborati (documenti di progetto) devono essere sviluppati in ogni fase, dirigere il lavoro dei singoli e del gruppo, fornire criteri di controllo e valutazione del sistema realizzato. Nella fig. 6.3 viene illustrato il ciclo di vita dello sviluppo di un sistema informativo in generale, che verrà adottato anche per la realizzazione di sistemi informativi basati su web.

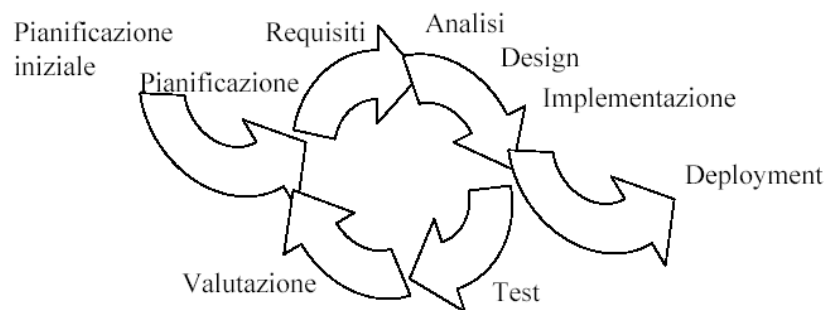


Figura 6.3: Il ciclo di vita dello sviluppo di un sistema informativo

Il primo aspetto da notare è che il ciclo di vita illustrato ha carattere iterativo. Dopo una fase iniziale, in cui viene effettuata la pianificazione del progetto, si prosegue con una serie di fasi che potranno essere eseguite anche più volte, fino a giungere alle fasi conclusive del progetto, in cui il sistema realizzato viene installato (deployment e successive fasi operative non indicate nella figura).

Nel seguito per ciascuna delle diverse fasi si illustrano in particolare alcuni aspetti rilevanti:

- i prodotti o risultati della fase: sono in generale i documenti di progetto o altri manufatti ottenuti al termine della fase;
- gli attori coinvolti nell'esecuzione della fase;
- il processo metodologico proposto per l'esecuzione della fase
- le relazioni con le altre fasi, e soprattutto le relazioni tra i risultati prodotti nelle diverse fasi.

La pianificazione del progetto viene descritta in dettaglio nella sezione 6.2. L'obiettivo di questa fase è di produrre un piano di progetto che fornirà le indicazioni generali che consentono l'esecuzione del progetto nell'ambito

di vincoli di tempo e risorse e obiettivi ben definiti. In ciascun ciclo iterativo verranno eseguite le fasi di raccolta dei requisiti, in cui si definiscono gli elementi che vengono successivamente analizzati nella successiva fase di analisi per definire con precisione le funzionalità e i limiti operativi del sistema, la progettazione dettagliata del sistema, la sua realizzazione, il testing per verificare che il sistema funzioni secondo quanto previsto inizialmente nei requisiti; successivamente si effettua una valutazione di quanto realizzato per verificare che il sistema effettivamente corrisponda a quanto richiesto.

In questa fase, ma anche nelle precedenti, potranno emergere nuovi requisiti, la necessità di gestire di nuovi scenari (ad esempio eccezioni non previste inizialmente, nuovi casi di esecuzione). E' per questo che lo sviluppo del sistema assume carattere iterativo, fino a giungere alla conclusione del progetto quando tutti i requisiti siano stati soddisfatti. Ogni ciclo di iterazione quindi si conclude con una valutazione con i referenti nel progetto e una eventuale ripianificazione dell'eventuale iterazione successiva.

E' da notare che lo schema operativo sopra illustrato potrebbe portare alla conclusione che le attività nel progetto si svolgano in maniera strettamente sequenziale. In realtà, in certi momenti queste attività possono essere svolte in parallelo allo scopo di ridurre i tempi complessivi, ad esempio iniziando a realizzare alcune delle funzionalità già definite mentre ancora sono in corso di progettazione altre funzionalità.

6.1.3 Sviluppo di un sistema informativo basato su web

Il ciclo di sviluppo sopra descritto ha caratteristiche generali nella realizzazione di progetti software per sistemi informativi. Nel caso di sistemi informativi basati su web vi sono inoltre alcune caratteristiche che portano in generale a attività di sviluppo continuo del sistema, con l'aggiunta di nuove funzionalità o l'integrazione con altri sistemi. Il contenuto del sistema in genere cambia continuamente con una crescita continua del sistema informativo. Il ciclo di vita dello sviluppo del sistema viene quindi iterato continuamente per soddisfare nuovi requisiti. Le professionalità coinvolte nello sviluppo del sistema non sono necessariamente informatiche, in quanto sono particolarmente rilevanti gli aspetti relativi alla presentazione grafica e alla redazione dei contenuti.

Vi sono quindi in genere tutte le caratteristiche presenti nella gestione di un processo complesso, a cui viene spesso data poca attenzione. Anche la responsabilità nell'organizzazione può essere ambigua, con il coinvolgimento di diverse unità operative nell'azienda.

Le responsabilità legali, sociali, etiche sono anch'esse spesso sottovalutate nell'organizzazione del progetto. Inoltre, i sistemi informativi basati su web hanno una prevalenza, in generale, di utenti esterni all'organizzazione, che possono porre requisiti particolari nella realizzazione, dovuti a differenze regionali, culturali, linguistiche.

Nel presente capitolo si utilizza UML (Unified Modeling Language) come strumento di modellazione dei diversi aspetti considerati .

Attori nel processo di progettazione e realizzazione.

Nella fig. 6.4 vengono illustrati i principali partecipanti alle diverse fasi di progettazione e i prodotti a cui contribuiscono.

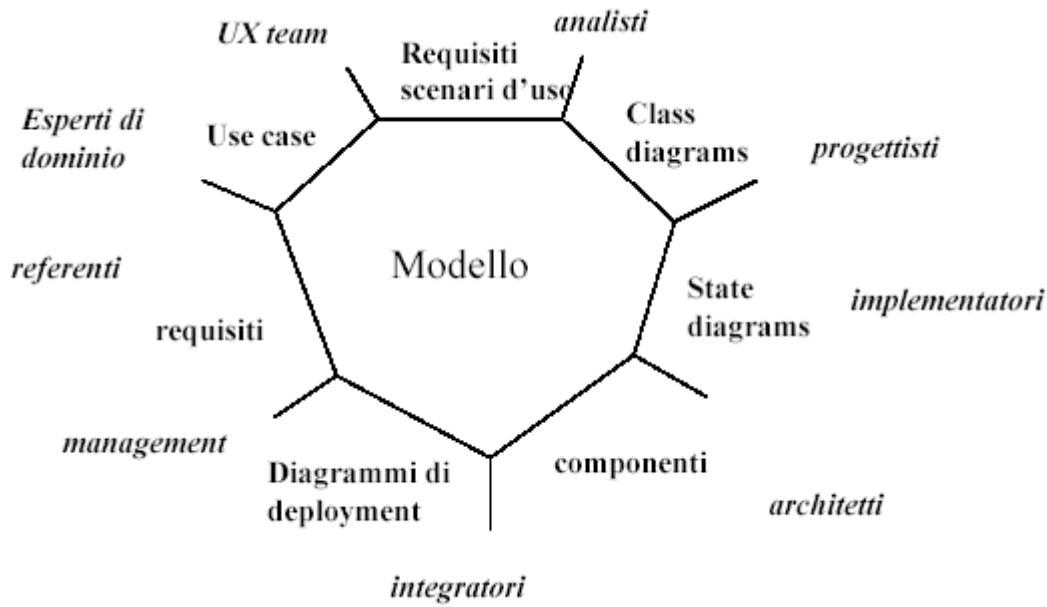


Figura 6.4: I partecipanti al processo e i prodotti

Il diagramma ripercorre il ciclo iterativo già presentato, indicando alcuni prodotti principali delle fasi progettuali. Iniziando dai requisiti, i principali attori coinvolti sono il management dell'organizzazione che ha commissionato il progetto e i referenti o stakeholder, interni o esterni all'organizzazione stessa, che costituiscono i principali azionisti nello sviluppo del progetto. Per definire i requisiti ci si avvale spesso di esperti del dominio per raggiungere velocemente la definizione dei principali aspetti del sistema e delle principali funzionalità, rappresentate tramite diagrammi di casi d'uso di UML e altri documenti collegati.

Per modellare adeguatamente l'interazione dell'utente con il sistema viene definita una squadra di utenti (o UX - User eXperience - Team), che è composta da esperti nella progettazione della navigazione nel sito, della presentazione grafica, nella comunicazione e da rappresentanti delle diverse tipologie di utenti del sistema. Le successive fasi di progettazione vengono svolte da figure tradizionali nello sviluppo di sistemi software e sistemi informativi: analisti, progettisti, che qui includono anche gli esperti nella progettazione di basi di dati, implementatori, architetti del software, integratori.

Caratteristiche di qualità

Nella realizzazione di un sistema informativo basato su web la progettazione ha alcune caratteristiche di qualità del sistema che guidano il progettista nell'effettuare alcune scelte. I principali indici che caratterizzano la qualità di un sistema informativo basato su web vengono descritti nel seguito.

- **Navigabilità.** Un sito web è caratterizzato dalla sequenza di presentazione delle informazioni e dei servizi all'utente. I collegamenti tra pagine devono essere progettati per rendere facile il passaggio da una pagina del sito alle altre a essa collegate per quanto riguarda le funzionalità da offrire all'utente. In particolare aspetti rilevanti sono la profondità dei livelli di navigazione e il numero di link per pagina. Tali aspetti possono essere considerati valutando tali valori in media su tutte le pagine o su un campione di pagine particolarmente significativo per il sito.
- **Accessibilità.** Le informazioni devono essere accessibili a tutti, anche, e soprattutto, alle persone diversamente abili. In particolare, il funzionamento del sito e i servizi forniti non dovrebbero dipendere dal browser utilizzato (indipendenza da browser). La Web Accessibility Initiative (WAI) del consorzio W3C ha definito alcune regole per la valutazione dell'accessibilità e reso disponibili alcuni strumenti per valutare l'accessibilità del sito.

- Leggibilità. Le informazioni devono essere proposte in modo adeguato, sia per quanto concerne la presentazione sia riguardo i contenuti. Per valutare la leggibilità di un sito è possibile condurre test on site o interviste on line (tramite questionari) agli utenti che utilizzano il sito.
- Affidabilità. Deve essere garantito il funzionamento del sistema.
- Manutenibilità. Poiché i contenuti e i servizi sui sistemi web sono in continua evoluzione, deve essere particolarmente curata la modificabilità del sito, in vista di una possibile evoluzione del sistema. La modificabilità può essere valutata in termini di costo della modifica.
- Usabilità. Indica la facilità per gli utenti nell'utilizzare il sito, ad esempio: la disposizione dei menu nella pagina, l'interpretazione dei bottoni, quante e quali immagini mettere (meglio un'immagine o un testo?), ...
- Compatibilità e interoperabilità. Interazione tra diversi sistemi che vengono utilizzati e integrati per fornire i servizi su web. La valutazione è in termini di costo dell'integrazione dei dati e dei servizi.
- Sicurezza. Protezione delle informazioni riservate. Può essere effettuata una valutazione economica dei costi e dei rischi.
- Efficienza. Uso delle risorse e comportamento del sistema nel tempo, valutata ad esempio come tempo di risposta medio o massimo

6.2 Pianificazione

6.2.1 Descrizione dettagliata delle fasi

I passi di progettazione della fase di pianificazione vengono illustrati nella fig. 6.5, in cui vengono poste in evidenza le fasi iniziali di progettazione. Le prime fasi comportano un'analisi dell'azienda e dei problemi da risolvere o delle opportunità che si rilevano nella realizzazione di un sistema informativo basato su web; una prima indagine sui sistemi già disponibili e sulle possibilità di integrazione nella realizzazione del sistema.

In parallelo vengono identificati i principali aspetti del dominio applicativo di interesse, sviluppando un modello di dominio che descrive i principali oggetti applicativi e i principali attori. Sulla base di queste analisi iniziali viene quindi sviluppato un documento di vision, in cui vengono delineati gli obiettivi del sistema, le principali funzionalità, le principali scelte realizzative.

Viene successivamente sviluppato un piano di progetto, prima di iniziare il ciclo di iterazione della progettazione già delineato nella sezione precedente, comprendente le fasi di raccolta e analisi dei requisiti, analisi dettagliata, design, implementazione e test nell'ambiente di sviluppo ed a seguire deployment dell'applicazione realizzata.

In parallelo vengono sempre svolte attività di gestione dei cambiamenti e gestione del progetto che non verranno discusse nel presente testo.

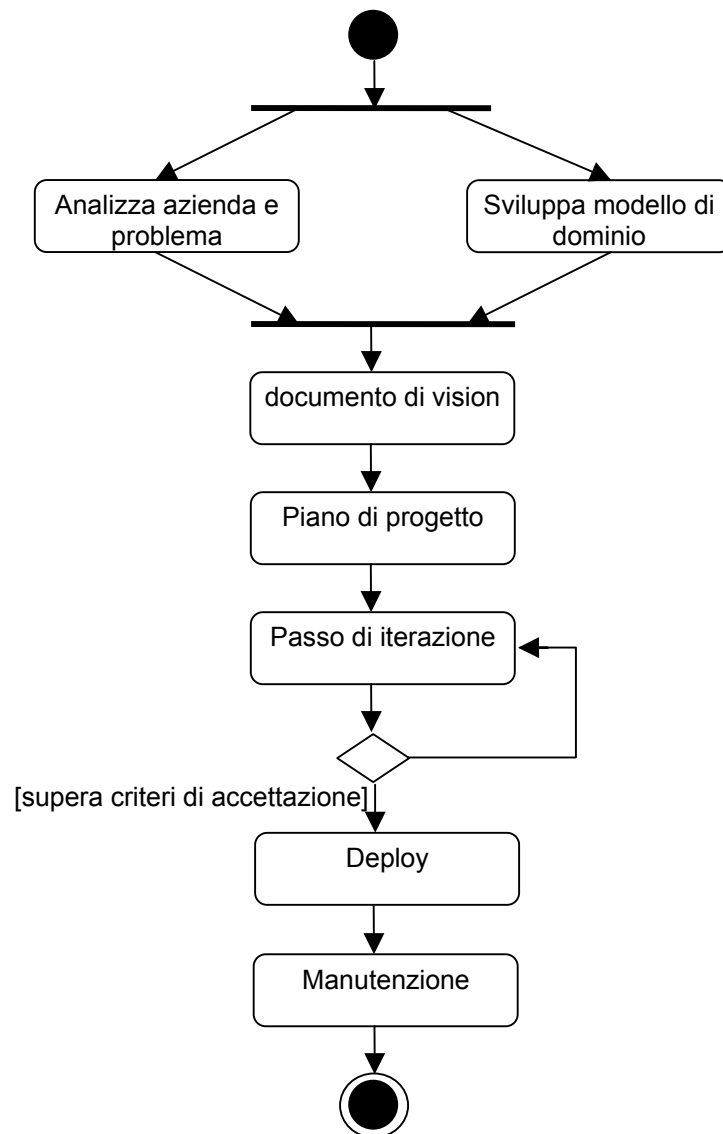


Figura 6.5: Il processo di sviluppo di un WIS: dettaglio delle fasi iniziali

6.2.2 Analisi dell'azienda e del problema

All'inizio del progetto è essenziale definire correttamente gli obiettivi del lavoro da svolgere e effettuare una pianificazione di massima del progetto.

Il primo passo da effettuare, preliminare alle attività successive, è comprendere le attività dell'azienda in relazione al progetto di sistema informativo su web da realizzare. In tale fase il gruppo di analisi e pianificazione del sistema interagirà con i referenti (stakeholder) del committente del progetto e analizzerà tutta la documentazione disponibile relativa a regolamenti, leggi, organizzazione dell'azienda utile a definire i confini del sistema e eventuali interventi organizzativi necessari contestualmente alla realizzazione del sistema informatico. Nel seguito ci si focalizza sulle attività che portano alla realizzazione del sistema.

6.2.3 Analisi dei dominio

Nel corso dell'analisi dell'azienda si rappresentano in forma diagrammatici i principali concetti rilevanti nel dominio applicativo trattato. Tramite un diagramma entità-relazione (ER) vengono indicati i principali concetti di dominio, le relazioni tra essi e gli attori a essi collegati. Vengono inoltre rappresentati i principali processi aziendali allo stato attuale, utilizzando diagrammi di attività di alto livello, in cui non vengono illustrati i dettagli, ma solo le attività principali in ciascun processo. Viene anche redatto un glossario, contenente l'elenco dei termini contenuti nei diagrammi (entità, relazioni, attributi, attività) e le loro definizioni.

6.2.4 Documento di vision

Il documento di vision viene tratto dall'analisi dei risultati dei passi precedenti e esprime il contesto e gli obiettivi dell'intero progetto. Spesso tale documento serve a come base della decisione di finanziare il progetto e pertanto deve anche contenere una valutazione comparativa delle possibili scelte realizzative. Il documento è strutturato come segue:

- Introduzione: perché si realizza il sistema, identificazione referenti, possibili alternative con valutazione costi e benefici, priorità
- Background: motivazioni di supporto, contesto aziendale, sistemi preesistenti • Requisiti e funzionalità generali: breve descrizione di cosa deve fare il sistema
- Definizione dell'architettura generale del sistema (software architecture document)

Un esempio di documento di vision, molto sintetizzato, è illustrato in fig. 6.6.

Introduzione L'obiettivo del progetto è quello di offrire ai clienti della ditta COMP i servizi di vendita su canali alternativi a quelli tradizionali. In particolare in questo progetto si esamina l'offerta di servizi su internet, ma nello sviluppo del progetto si porrà particolare attenzione alla possibilità di offrire gli stessi servizi in futuro su altri canali (call center, su smartphone) estendendo il presente progetto. Si proporrà quindi un'architettura flessibile basata sull'utilizzo di web-services. Alternative possibili I referenti nel corso del progetto saranno l'amministratore delegato della ditta COMP, il responsabile dei sistemi informativi, e verrà selezionato un gruppo di grandi clienti a cui verranno presentate le caratteristiche del progetto.

Background. Attualmente le vendite avvengono presso i punti di vendita. Tutti i punti di vendita sono collegati al sistema informativo della ditta COMP su cui sono registrati i componenti disponibili e le possibili configurazioni. E' possibile effettuare i pagamenti degli ordini al momento dell'acquisto anche con carta di credito e su richiesta emettere fatture

Requisiti generali e funzionalità. L'azienda COMP offre la possibilità di acquistare computer via Internet. Il cliente può selezionare un computer sulla pagina web dell'azienda. Il cliente può selezionare una configurazione già predisposta oppure comporre il proprio sistema assemblando componenti, con l'assistenza del sistema. Per ogni configurazione il cliente può valutare il costo. Per ordinare il cliente deve fornire le informazioni per la consegna e il pagamento. Il pagamento deve avvenire con carta di credito. Una volta inviato l'ordine, viene inviata via mail una conferma al cliente. Il cliente in attesa di consegna può controllare lo stato dell'ordine in ogni momento. Nel back-end viene controllata la solvibilità del cliente, viene richiesta al magazzino la configurazione ordinata, viene stampata la fattura e contattato un servizio di spedizione per effettuare la consegna.

Requisiti architetturali Il sistema dovrà consentire di svolgere via web le attuali operazioni di predisposizione degli ordini. Dovrà pertanto utilizzare i sistemi già disponibili per le operazioni di controllo dell'ordine, contabili e di magazzino (sistema di tipo web delivery). Il sistema deve poter essere utilizzato su ogni tipo di browser (thin client).

Figura 6.6: Esempio di parte di documento di vision

6.2.5 Definizione del piano di progetto

Il piano di progetto indica la durata del progetto, le risorse necessarie e le principali scadenze nella realizzazione.

Tramite un Gantt (vedi fig. 6.7) è possibile definire la Attività e le sottoattività (numerate) nel progetto, i risultati di ciascuna attività e associare a ciascuna di esse inizio e durata e le risorse (mesi uomo, costi) necessarie. In genere vengono definite le scadenze (milestone) periodiche o in momenti significativi del progetto, in cui è prevista una verifica dell'effettivo stato di avanzamento del progetto sulla base dei risultati ottenuti. Vengono anche indicati i costi di ciascuna attività, con riferimento alle risorse utilizzate.

6.2.6 Documenti prodotti nella fase di pianificazione

Nella fase di pianificazione saranno quindi prodotti i seguenti documenti progettuali:

- schema dei concetti nel dominio (schema ER)
- schemi dei principali processi aziendali (diagrammi delle attività)
- glossario dei termini (documento testuale)
- documento di vision, strutturato in introduzione, background, requisiti e funzionalità generali, definizione dell'architettura generale del sistema (documento testuale)
- piano di progetto (Gantt)

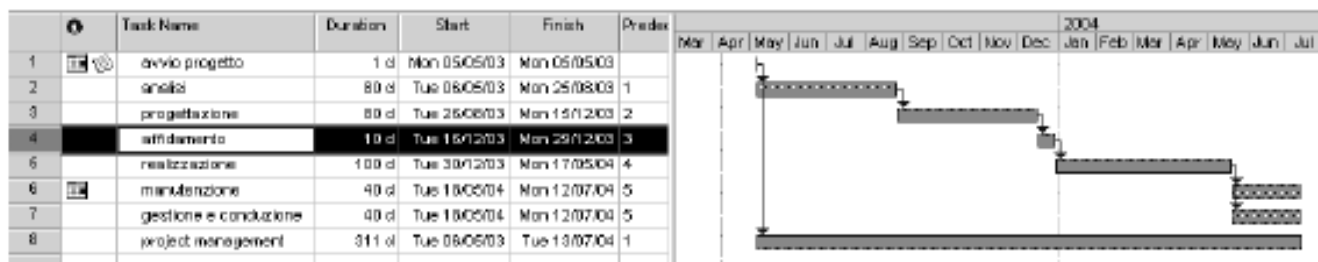


Figura 6.7: Esempio di Gantt

6.3 Raccolta e analisi dei requisiti

Nella fase di raccolta e analisi dei requisiti l'obiettivo è quello di giungere a una descrizione del sistema non ambigua, che consenta di procedere con le fasi successive di progettazione del sistema. Il risultato di questa fase sarà un insieme di documenti di progetto che descrivono i requisiti. La raccolta dei requisiti sarà necessariamente incompleta, in quanto non sarà possibile in questa fase specificare tutti i possibili requisiti in dettaglio, e prevedere tutti i possibili comportamenti del sistema. Infatti in questa fase, che è opportuno mantenere limitata nel tempo rispetto alla durata del progetto, un criterio sarà anche quello di raccogliere un insieme di requisiti che sia possibile discutere con i referenti del progetto. Saranno quindi da bilanciare le esigenze di dettaglio necessarie alle fasi successive e quelle di non fornire ai referenti una documentazione troppo onerosa da esaminare e incomprensibile e di contenere i costi dovuti alla raccolta dei requisiti.

In questa sezione verranno pertanto illustrati alcuni criteri metodologici per svolgere la fase di raccolta dei requisiti e i modelli che consentono di rappresentare i requisiti per discutere e concordare le caratteristiche del sistema con i referenti individuati nella fase iniziale di pianificazione.

Verranno illustrati due attività progettuali svolte in questa fase:

- Raccolta dei requisiti funzionali, non funzionali, architetturali, in forma testuale (6.3.1)
- Specifica dei requisiti tramite diagrammi di casi d'uso UML, schemi ER (Entità Relazione), diagrammi User eXperience (UX) (6.3.2)

6.3.1 Raccolta dei requisiti

Considerazioni generali

La raccolta dei requisiti procede tramite l'analisi della documentazione disponibile e effettuando interviste.

La documentazione utile a identificare le caratteristiche del sistema sono tutti i documenti rilevanti, modelli di processi e di dati, l'analisi di dati quali record nelle basi di dati e log dei sistemi. Un documento che guida le attività di raccolta dei requisiti è il documento iniziale di vision, che consente di valutare correttamente le priorità di intervento richieste. Vengono anche utilizzati come base di partenza i modelli di dominio creati nella fase di pianificazione iniziale.

Le interviste andranno effettuate con gli stakeholder (azionisti o referenti) del sistema, che forniranno le indicazioni principali sulle funzionalità desiderate e con gruppi di utenti selezionati (UX team).

I principali referenti del progetto, inclusi i rappresentanti degli utenti, sono stati identificati nella fase di pianificazione. Il UX (User eXperience) team è particolarmente rilevante per la definizione dei requisiti riguardanti l'interazione degli utenti con il sistema che verrà realizzato. I requisiti dovranno indicare i comportamenti previsti nel sistema. Si possono perciò considerare come un insieme di vincoli che ne descrivono le caratteristiche, ad esempio con frasi quali: il sistema dovrà....

I requisiti non dovranno esprimere proprietà generali (ad esempio, un sistema con un tempo di risposta accettabile), ma piuttosto comportamenti con proprietà verificabili in fase di test. Sarà quindi opportuno arrivare a definire criteri misurabili e casi di test sui quali verificare il comportamento del sistema (ad esempio, il sistema, con 10 utenti collegati, dovrà avere un tempo di risposta inferiore ai 6 secondi).

I casi di test verranno utilizzati per verificare anche formalmente se il sistema realizzato corrisponde a quanto richiesto in questa fase.

Documentazione dei requisiti iniziali

I requisiti, che verranno inizialmente espressi in forma prevalentemente testuale, verranno numerati secondo una numerazione gerarchica (ad esempio 1.3.1), raggruppando allo stesso livello requisiti correlati allo stesso livello di dettaglio.

La numerazione faciliterà la fase di analisi dei requisiti, in cui esigenze diverse e a volte contrastanti dovranno essere ricomposte, e faciliterà inoltre la tracciabilità dei requisiti nelle fasi successive di progettazione.

Infatti, nella fasi successive si passerà a rappresentazioni sempre più formali, tramite modelli o notazioni in genere di tipo semiformale. Sarà sempre importante collegare i modelli ai requisiti raccolti: ogni elemento nei modelli deve corrispondere almeno a un requisito. Infatti questa corrispondenza serve a verificare se sono davvero utili le funzionalità fornite dal sistema. A volte nei sistemi informatici vengono inserite nelle fasi di analisi e progettazione dettagliata funzionalità aggiuntive di tipo generale che potrebbero non risultare utili e ostacolare una realizzazione rapida del sistema.

Inoltre la numerazione dei requisiti consente di analizzare più facilmente l'impatto di eventuali cambiamenti nei requisiti stessi. Difatti una caratteristica nella realizzazione di sistemi informativi è la scarsa stabilità nei requisiti, che tendono a evolvere nel tempo, durante lo sviluppo del sistema, a fronte di nuove esigenze o della valutazione di parti del sistema realizzate.

La tracciabilità dei requisiti consente di mettere in corrispondenza le modifiche richieste con gli schemi da modificare per soddisfare tali modifiche.

Req1. L'azienda COMP fra la possibilità di acquistare computer via Internet. Il cliente può selezionare un computer sulla pagina web dell'azienda. Il cliente può selezionare una configurazione già predisposta oppure comporre il proprio sistema assemblando componenti, con l'assistenza del sistema. Per ogni configurazione il cliente può valutare il costo. Per ordinare il cliente deve fornire le informazioni per la consegna e il pagamento.

Req2. Il pagamento deve avvenire con carta di credito. Una volta inviato l'ordine, viene inviata via mail una conferma al cliente. Il cliente in attesa di consegna può controllare lo stato dell'ordine in ogni momento.

Req3. Si deve controllare la solvibilità del cliente, viene richiesta al magazzino la configurazione ordinata, viene stampata la fattura e contattato un servizio di spedizione per effettuare la consegna.

Req4. Il sistema mostra report sintetici sugli acquisti

Figura 6.8: Esempio di requisiti funzionali generali

E' utile inoltre assegnare priorità ai requisiti. Tali priorità aiuteranno nella fasi successive di progettazione, in particolare a risolvere eventuali conflitti generati da requisiti contrastanti.

Requisiti funzionali

I requisiti funzionali descrivono le funzionalità che dovranno essere inserite nel sistema. Lo scopo è quello di descrivere, per ora in forma testuale, le operazioni che sarà possibile effettuare con il sistema, chi le effettua e in che momento, eventuali interazioni con altri sistemi di back-end.

Le descrizioni potranno essere molto generali, ma in alcuni casi andrà fornito l'opportuno dettaglio qualora si vogliano dare indicazioni specifiche ai progettisti.

Nella figura 6.8 viene riportato un esempio di requisiti funzionali generali per il sistema di vendite on line che varrà utilizzato come esempio nel seguito.

I requisiti funzionali di un sistema informativo basato su web in Internet saranno principalmente volti a definire le funzionalità del sistema per gli utenti esterni all'azienda.

Altri requisiti funzionali forniranno eventuali dettagli per funzionalità a uso interno dell'azienda, come riportato nell'esempio di figura 6.9

Negli esempi indicati si può notare che i requisiti definiscono vincoli di funzionamento del sistema, con frasi di tipo dichiarativo (il cliente usa, il sistema deve). Queste frasi andranno raffinate e precisate nella loro formulazione per evitare ambiguità di interpretazione.

1.1 Il cliente usa la pagina web d'acquisti on line del produttore per selezionare una configurazione standard del server, desktop o computer portatile che potrebbe interessargli. Viene mostrato il prezzo.

4.1 Il sistema deve produrre un sommario delle vendite settimanali

Figura 6.9: Esempio di descrizione di dettaglio dei requisiti generali 1 e 4

Ad esempio, nel requisito n. 4.1, non viene evidenziato chi utilizzerà la funzionalità prevista. Alla formulazione "il sistema deve produrre un sommario" è preferibile sostituire "il personale dell'azienda addetto alle vendite potrà visualizzare un sommario", indicando chi utilizzerà la funzionalità prevista.

Requisiti non funzionali

Oltre ai requisiti relativi alle funzionalità del sistema, nella fase di raccolta dei requisiti si devono indicare le caratteristiche generali che il sistema dovrà avere riguardo a aspetti di qualità del WIS.

Le principali qualità da considerare sono le seguenti:

- *Usabilità*
(es: massimo 4 click per raggiungere una funzionalità, non usare frame, browser qualunque che supporti le tabelle)
- *Prestazioni*
Es: Tempo massimo per caricare una pagina, almeno 150 sessioni simultanee
- *Robustezza/affidabilità*
(rispetto a 24/7/52) 0,9999, oppure down 1 ora alla settimana per manutenzione
- *Sicurezza*
A chi è accessibile (ruoli, matrice funzioni/ruoli) Meccanismi: controllo accessi, autenticazione crittografia, audit, intrusion detection
- *Deployment*
Come l'applicazione viene consegnata al cliente: installazione, manutenzione, scalabilità.
- *Architettura*
Tra i requisiti è necessario anche porre i vincoli architetturali alla base del progetto (design e implementazione). I vincoli riguardanti l'architettura possono essere di diverse tipologie, quali quelli riguardanti la piattaforma realizzativa, i set di caratteri ammissibili. La formulazione dei requisiti architetturali verrà discussa più in dettaglio nella sezione seguente, in cui verranno discussi i vari punti di vista a partire dai quali si formulano tali requisiti.

Requisiti architetturali

L'obiettivo della raccolta dei requisiti architetturali è quello di produrre un documento, il SAR (System Architectural Requirements), che contiene la descrizione dei principali vincoli architetturali e delle loro motivazioni.

Il termine architettura assume un significato diverso secondo i punti di vista di alcuni attori coinvolti nel processo di progettazione, e sarà necessario coinvolgere in questa fase tutti coloro che possono fornire indicazioni a questo proposito. Come per gli altri requisiti, l'obiettivo è quello di fornire gli elementi per le fasi di progettazione successive e per la verifica finale del sistema realizzato.

Nel SAR consideriamo i seguenti punti di vista:

- *Punto di vista dei requisiti generali*
In questa categoria vengono specificati i requisiti relativi alle funzionalità osservabili del sistema, quali ad esempio i set di caratteri da utilizzare, le ipotesi sulle caratteristiche dei browser utilizzati dagli utenti (utilizzabilità dei cookie, browser speciali).
- *Punto di vista del design*
Nel progetto dell'architettura dal punto di vista del design verranno definiti i requisiti e i vincoli relativi alla realizzazione del sistema software, le piattaforme disponibili, il middleware utilizzato e eventuali componenti di terze parti che verranno riutilizzate o con cui il sistema interagirà.
- *Punto di vista della realizzazione*
Vengono fornite indicazioni relative all'architettura hardware del sistema. Vengono definiti i requisiti minimi hardware per la realizzazione rispetto all'architettura delineata nel documento di vision.
- *Punto di vista del testing*
I requisiti dal punto di vista del testing devono definire indicazioni per il gruppo di sviluppo e i referenti per la fase di valutazione del sistema realizzato. vengono definite misure di qualità e prestazioni del sistema. Nella prima fase di sviluppo viene definito un piano di test delle funzionalità e dell'architettura.

Come nel caso degli altri requisiti, i requisiti architetturali verranno analizzati e verrà assegnata una priorità per individuare requisiti architetturali significativi (SAR). A partire dai requisiti verrà definita una architettura candidata, che potrà essere predisposta per preparare e valutare prototipi. Sempre nell'ambito della definizione dell'architettura verrà definita una strategia di riuso dei componenti realizzati.

Esempio

Il sistema chiede il nome e l'indirizzo. L'utente compila un modulo sullo schermo. Quando inserisce il CAP, il sistema riempie automaticamente il nome della città se non è già stato riempito. Quando l'indirizzo è completo il cliente preme il tasto Avanti per proseguire nel completamento dell'ordine

Figura 6.10: Esempio di requisito con implicazioni architetturali

Un esempio di funzionalità a cui possono essere associati requisiti architetturali è presentata in fig. 6.10. In questo caso verranno identificati i seguenti requisiti architetturali:

- *Punto di vista dei requisiti generali:* CAP significa solo clienti italiani? Quale ipotesi lato client? Client ad hoc per l'applicazione disponibile per ogni utente? In Internet: in generale non si può ipotizzare che vi sia un software caricato sul client

- *Punto di vista del design*: quali pattern architetturali per il presentation tier vengono adottati? thin o fat web client? quali sono le funzionalità che si suppongono disponibili nel browser? i cookie possono essere utilizzati o sono disabilitabili? La connessione è tramite http o con https? Sono presenti firewall che possono ostacolare l'interazione di un fat client con componenti applicativi lato server (può utilizzare non solo HTTP, ma anche IIOP e DCOM)?

6.3.2 Specifica dei requisiti

I primi passi di progettazione sono volti a definire le caratteristiche generali del sistema informativo basato su web, e portano alla definizione delle principali funzionalità e a una prima progettazione dell'interazione dell'utente con il sistema. L'interazione dell'utente di un sistema informativo basato su web verrà modellata rappresentando i principali percorsi di navigazione nel sistema, visto dall'utente come collezione di pagine web tra loro collegate.

I requisiti raccolti potranno presentare anche inconsistenze e livelli di dettaglio molto diversi. L'obiettivo della specifica dei requisiti è anche quello di chiarire e confrontare i requisiti raccolti e integrarli in un insieme di requisiti coerenti, che formeranno la base per le fasi di progettazione e realizzazione successive.

Nella fase di specifica dei requisiti funzionali verranno utilizzate diverse tecniche di modellazione messe a disposizione da UML. Due saranno le principali notazioni utilizzate in questo capitolo: *i casi d'uso*, che consentono di modellare le interazioni degli utenti con il sistema informativo e i principali servizi, e i *diagrammi delle classi* che verranno utilizzati come base per modellare la navigazione tra le pagine e il contenuto delle pagine stesse.

I modelli così ottenuti, come vedremo nel seguito, potranno essere accompagnati da ulteriori modelli, che, con altre notazioni di UML, consentiranno di documentare i casi d'uso specificando a grandi linee la logica applicativa dei servizi.

Come rappresentare i requisiti del sistema:

- Come già discusso nella sezione 6.2 è essenziale l'adozione di modelli per rappresentare in modo preciso i risultati di ciascuna fase di progettazione e per garantire la tracciabilità tra i diversi documenti di progetto. Nella specifica dei requisiti verranno adottati modelli concettuali per la rappresentazione dei requisiti, come strumento per rappresentare in modo sintetico e preciso i requisiti.
- Nel seguito, nella sez. 6.3.3 viene illustrato come gli Use case diagram di UML consentono di rappresentare gli attori, la loro interazione con il sistema, e come i concetti del dominio possono essere rappresentati tramite un modello delle classi.
- L'interazione degli utenti con il sistema viene rappresentata tramite particolari diagrammi delle classi, denominati User eXperience (UX model), che consentono di rappresentare sinteticamente la navigazione nel WIS e gli elementi principali nelle pagine del sito. Il modello UX verrà illustrato nella sezione 6.3.5

6.3.3 Casi d'uso

In questa sezione si esaminano i casi d'uso, che vengono utilizzati per specificare le funzionalità di un sistema informativo basato su web. Nel specificare l'interazione con il sistema tramite casi d'uso si seguono i classici passi di progettazione utilizzati per definire modelli di casi d'uso per applicazioni software. I casi d'uso contengono una rappresentazione di come il sistema informativo si comporta rispetto a interazioni di utenti o in generale a input esterni.

Nel caso di un sistema informativo web based si vogliono modellare come *attori* sia utenti del sistema intesi come persone che utilizzeranno le funzionalità del sistema, sia altri sistemi informativi con cui il sistema dovrà interagire. Si rappresentano sia interazioni volte alla fornitura di servizi di *front office*,

per ricevere richieste da utenti o altri sistemi, sia interazioni di *back office* (con utenti o sistemi) necessarie per fornire i servizi stessi.

Si percorreranno i seguenti passi progettuali:

- Passo 1: identificare attori;
- Passo 2: identificare casi d'uso;
- Passo 3: disegnare un diagramma dei casi d'uso;
- Passo 4: documentare i casi d'uso.

Passo 1: identificare attori

Gli attori vengono identificati nei requisiti, elencando per ciascun requisito gli utenti che interagiscono con il sistema e altri sistemi informativi con cui il sistema deve interagire. Si identificano sia gli utenti che richiedono servizi forniti dal sistema, che quelli che interagiscono con il sistema al fine di consentire di fornire i servizi richiesti. In questo secondo caso si parla di utenti che svolgono attività di back-end.

Alcuni attori che interagiscono con un sistema possono essere altri sistemi informativi e si potrà parlare in questo caso di attori automatizzati. Un esempio di attore automatizzato è un sistema informativo per la gestione di conti dei clienti sui quali il sistema potrà effettuare operazioni di verifica, addebito o accredito.

Non si considerano invece attori i sistemi a cui si accede in qualità di risorse utilizzate per la realizzazione del sistema. Un esempio di sistema di questo tipo è un sistema informativo legacy, che fornisce una interfaccia che verrà realizzata all'interno della piattaforma J2EE tramite un protocollo proprietario utilizzato da un modulo connettore. Anche le basi di dati che contengono dati che verranno utilizzati dall'applicazione come risorsa a cui si accede tramite collegamenti quali ad esempio JDBC vengono considerate interne al sistema informativo e non vengono visualizzate come interfaccia esterna.

Nell'esempio citato si identificano i seguenti attori:

- *Cliente*: interagisce con il sistema per effettuare ordinazioni
- *Sistema verifica conti*: ha accesso al sistema per effettuare operazioni di verifica dei pagamenti. Si noti che il sistema di verifica conti viene considerato un attore se ha proprie funzionalità di accesso al sistema, mentre non verrebbe rappresentato se considerato solo come sistema di back-end. Va altresì sottolineato che, come evidenziato dallo use case diagram, il verso della freccia che lega lo use case "richiedi pagamento cliente" e l'attore "sistema verifica conti" sottolinea il fatto che il coinvolgimento dell'attore è dettato dal sistema come una sorta di interrupt che richiede l'intervento dell'attore per far evolvere il processo.

Altri attori che potranno accedere al sistema potranno essere ad esempio il Servizio spedizione, che però non verrà preso in esame nel seguito.

Passo 2: identificare i casi d'uso

I casi d'uso rappresentano i servizi forniti dal sistema informativo agli attori che interagiscono con esso. E' possibile identificare i servizi a partire dai requisiti funzionali del sistema tramite un'analisi delle descrizioni testuali. Nelle descrizioni dei requisiti i casi d'uso corrispondono frequentemente a verbi nel testo che corrispondono a azioni nelle descrizioni. Alcuni esempi di verbi che corrispondono a azioni sono ordinare, spedire, pagare. Ad esempio, nella figura 6.9 si può identificare un servizio che consente di selezionare la configurazione di un computer standard. Si utilizzerà come convenzione per i nomi dei servizi un verbo all'infinito seguito dall'oggetto del servizio.

Per ogni requisito raccolto si indicheranno gli attori coinvolti e la funzionalità da realizzare. Alle funzionalità alla fine di questa fase risulteranno associati i nomi dei principali use case da progettare nelle fasi successive (Tabella. 6.1).

Requisito	Attore	Caso d'uso
1.1 - Il cliente usa la pagina web d'acquisti on line del produttore per selezionare una configurazione standard del server, desktop o computer portatile che potrebbe interessargli. Viene mostrato il prezzo.	Cliente	Mostrare configurazione computer standard
3 - Nel back-end viene controllata la solvibilità del cliente, ...	Sistema verifica conti	Richiedi pagamento cliente

Tabella 6.1: Corrispondenza tra requisiti, attori e use case

Per facilitare la tracciabilità dei requisiti e per verificare che le funzionalità previste corrispondano effettivamente a requisiti raccolti nella fase di raccolta dei requisiti, si predisporrà nella documentazione una tabella riassuntiva che indica, per ogni caso d'uso, i requisiti a cui si riferisce (indicati con la numerazione utilizzata nella raccolta) e gli attori che utilizzano il servizio (Tabella. 6.2).

caso	attore	requisiti
mostrare configurazione computer standard	cliente	Req1
costruire configurazione computer	cliente	Req1
ordinare computer	cliente	Req1
richiedi pagamento cliente	sistema verifica conti	Req3
controlla stato ordine	cliente	Req2

Tabella 6.2: Relazione tra use case e requisiti di fig 6.8

Passo 3: disegnare un diagramma dei casi d'uso

La funzionalità principali del sistema vengono rappresentate utilizzando la notazione UML dei *casi d'uso* (use case). Un caso d'uso, è rappresentato tramite una ellisse. Rappresenta una funzionalità del sistema con cui l'utente interagisce. Nei diagrammi d'uso si rappresentano solo le interazioni, mentre lo scopo non è quello di rappresentare una scomposizione funzionale del sistema, intesa come scomposizione di componenti applicativi in grado di realizzare una particolare funzionalità. Il diagramma dei casi d'uso rappresenta ad alto livello tutte le funzionalità del sistema. I casi d'uso identificati verranno collegati agli attori corrispondenti e verranno identificate le relazioni di inclusione e estensione tra i casi d'uso rappresentati.

Il diagramma dei casi rappresenta così l'insieme dei servizi forniti dal sistema informativo e le relazioni funzionali tra di essi. Ad esempio nella Fig. 6.12 si illustrano in un diagramma i casi d'uso per il sistema di vendite on-line della ditta COMP, illustrando la possibilità di ordinare sia prodotti preconfigurati, sia di configurare i prodotti all'interno dell'ordinazione. Viene anche illustrata l'interazione del sistema per la verifica dei conti con il sistema delle vendite on-line.

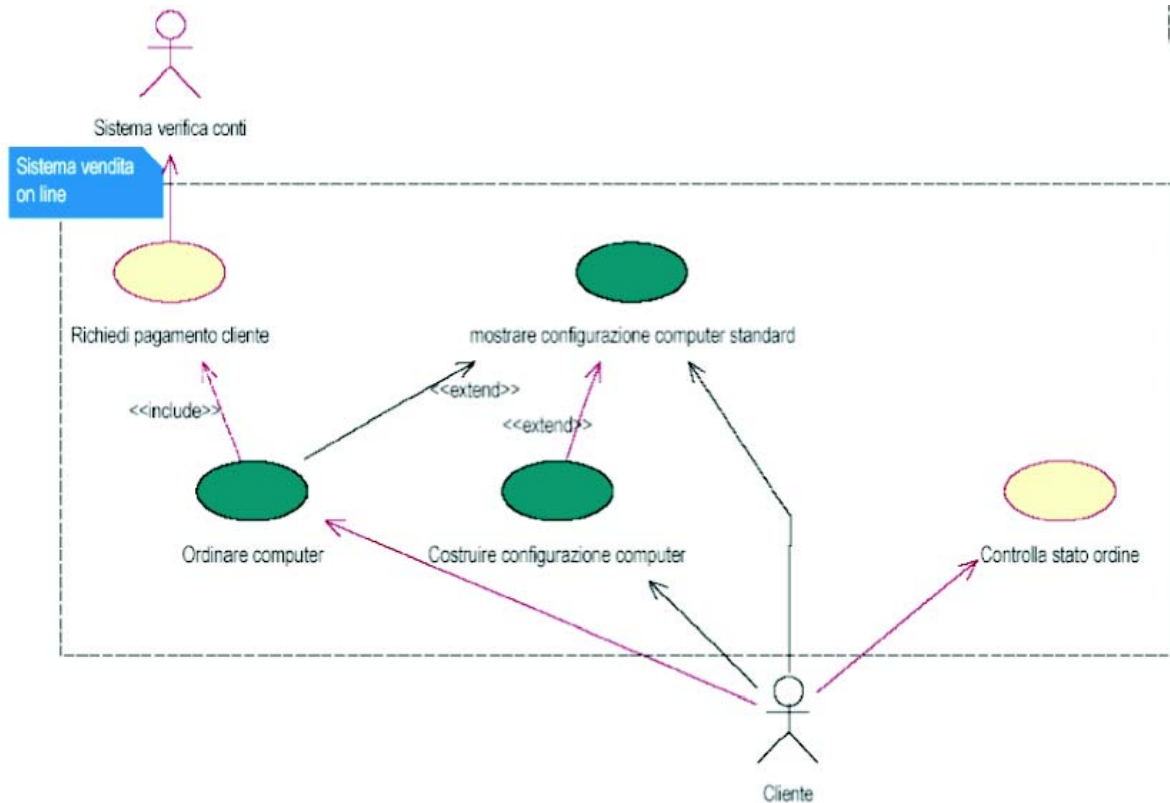


Figura 6.11: Use case per il sistema di vendita on line della ditta COMP

Nella costruzione del diagramma complessivo una difficoltà può essere rappresentata dai collegamenti stereotipati disponibili per collegare i casi d'uso. Vengono utilizzati usualmente i due stereotipi include e extend:

- **<<include>>**: un caso d'uso include un altro se necessita per funzionare di tutte le funzionalità fornite dall'altro caso d'uso.
- **<<extends>>**: un caso d'uso ne estende un altro se fornisce funzionalità che possono essere utilizzate in modo opzionale all'interno dello stesso.

La semantica associata a essi è differente. Nel caso di inclusione, si vuole rappresentare il caso in cui lo stesso servizio viene utilizzato da più servizi, e quindi la sua descrizione si ripete più volte. Rappresenta quindi un caso di riuso di servizio che può essere disponibile anche per più sistemi informativi.

Nel caso dell'estensione, si rappresentano funzionalità che l'attore può decidere di attivare, in aggiunta a quelle già fornite dal caso d'uso di partenza.

Ad esempio, nel diagramma di 6.11, il cliente può decidere di acquistare il computer in una data configurazione (caso d'uso "ordinare computer").

Nel predisporre i casi d'uso bisogna inoltre tenere presente che essi rappresentano solo relazioni di tipo strutturale tra i servizi, ma non forniscono indicazioni sulla sequenza delle operazioni svolte, o sull'ordine dei servizi stessi. Questi andranno rappresentati con altre notazioni associate ai casi d'uso, come viene discusso nella sezione successiva. Inoltre non si devono utilizzare le relazioni tra casi d'uso per rappresentare scomposizioni di tipo funzionale. Una scomposizione di tipo funzionale indica quali funzionalità compongono un dato servizio. Le relazioni tra casi d'uso possono rappresentare relazioni di inclusione o estensione, ma non forniscono indicazioni su come i servizi offerti vengono forniti da funzionalità componenti. Utilizzare i diagramma dei casi d'uso per rappresentare una scomposizione funzionale è quindi contrario alle finalità di rappresentazione del modello.

Passo 4: documentare i casi d'uso

I casi d'uso devono essere associati a una precisa descrizione delle funzionalità che si vogliono fornire. Tale descrizione potrà essere fornita solo in testo libero, oppure si potranno associare al testo anche altri diagrammi di UML, quali diagrammi delle classi, diagrammi di sequenza, diagrammi delle attività. L'obiettivo è quello di specificare per ogni servizio quali sono le principali funzionalità che svolge e l'interazione con il servizio stesso. La documentazione servirà nei passi successivi di analisi per identificare le classi per la realizzazione del servizio e le loro proprietà e interfacce. Tali descrizioni dovranno essere pertanto sufficientemente dettagliate per consentire agli analisti di progettare il sistema senza dover consultare nuovamente chi ha fornito i requisiti stessi.

In genere in un progetto la descrizione dettagliata e articolata dei casi d'uso sarà di circa 10 pagine per caso d'uso, e conterrà i seguenti elementi principali:

- *nome*
- *breve descrizione*
- *attori*
- *casi d'uso collegati*
- *precondizioni*: esprimono le condizioni che devono essere verificate al momento dell'interazione con il sistema all'interno del caso d'uso (ad esempio, l'utente deve essere un utente registrato per accedere a queste funzionalità)
- *postcondizioni*: esprimono le condizioni che devono essere verificate al termine dell'esecuzione del caso d'uso. Spesso tali condizioni rappresentano l'effetto dell'esecuzione del caso d'uso nella logica di business (ad esempio, il prodotto è stato configurato correttamente, il prodotto non è stato configurato).
- *descrizione dettagliata*

Come tutti i documenti UML, la documentazione associata ai casi d'uso evolverà durante il progetto, partendo da una sintetica descrizione iniziale, fino a arrivare a una documentazione dettagliata del caso d'uso che costituirà la documentazione delle funzionalità fornite dal servizio e potrà anche essere utilizzata in fase di testing del sistema.

La documentazione del caso d'uso potrà quindi comprendere inizialmente altri diagrammi, predisposti utilizzando le notazioni di UML, o fare riferimento a diagrammi più dettagliati prodotti nelle fasi successive del progetto. In particolare, sono spesso utilizzati:

- un *UML Activity Diagram* volto a descrivere le attività principali che costituiscono uno use case
- alcuni *UML Sequence Diagram* ognuno dei quali rappresenta un possibile scenario di utilizzo dello use case in termini di attività così come esse sono definite nell'activity diagram.

Activity Diagram

Attraverso l'Activity Diagram si vuole scomporre uno use case in una serie di attività in grado di rispecchiare, ad alto livello, i passi che il sistema deve compiere per dialogare con gli attori e per portare a compimento l'obiettivo del caso d'uso stesso. Un diagramma delle attività (vedere come esempio le figg. 6.12 e 6.5) può includere i seguenti elementi fondamentali (non verranno qui discussi i possibili elementi secondari proposti in UML, quali ad esempio associazioni di documenti alle attività svolte):

- *attività*: rappresentano azioni significative per il cambiamento dello stato degli oggetti nel sistema o per la gestione dello stato dell'interazione con l'utente (ad esempio, "aggiungi prodotto al carrello" nella figura 6.12). Le attività vengono rappresentate da rettangoli (con i bordi arrotondati);

- *collegamento tra attività*: rappresentano flussi di controllo tra attività. Il collegamento con una freccia da una attività A a una attività B rappresenta il fatto che terminata l'attività A potrà iniziare l'attività B;
- *barra di sincronizzazione*: la barra di sincronizzazione serve a indicare che più flussi di controllo vengono attivati in parallelo (ad esempio l'"analisi dell'azienda" e l'analisi del dominio nella Fig. 6.5). Analogamente la barra di sincronizzazione con più flussi in ingresso e uno in uscita indica che l'attività che segue la barra di sincronizzazione può iniziare solo quando tutti i flussi di controllo precedenti la barra sono stati terminati;
- *alternativa*: l'alternativa, indicata con un rombo e con più flussi in uscita, rappresenta il fatto che i flussi sono percorsi alternativi; è possibile etichettare con una condizione il flusso (indicata tra parentesi quadre) per indicare che il flusso viene attivato solo se la condizione è verificata.

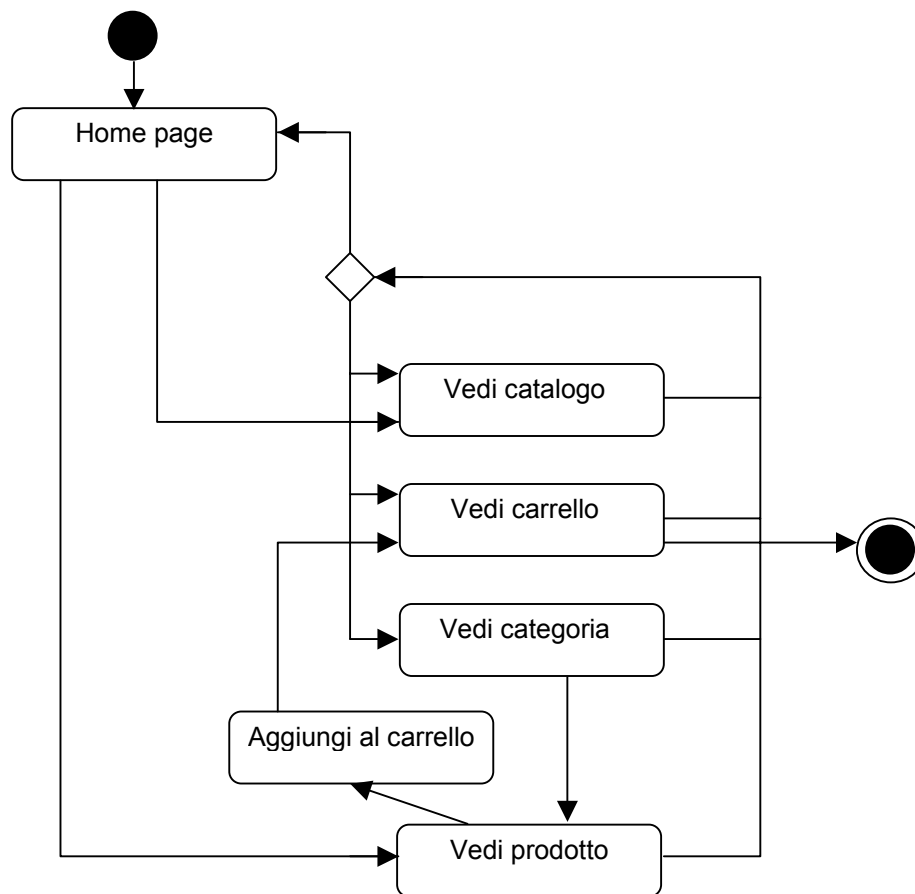


Figura 6.12: Esempio di diagramma delle attività
(questo diagramma mostra quali sono le (possibili) attività per aggiungere un prodotto nel carrello)

Lo Use Case Diagram ha messo in luce le relazioni che sussistono tra diversi use case principalmente in termini di stereotipi quali include ed extend.

Queste relazioni sono catturate anche all'interno dell'Activity Diagram. Infatti, in un Activity Diagram associato ad un particolare use case sono incluse particolari attività che si riferiscono agli use case collegati. Naturalmente, gli stereotipi usati per la definizione delle relazioni tra use case, influenzano anche la definizione dell'Activity Diagram. Riprendendo le definizioni degli stereotipi, quando uno use case legato ad un altro attraverso lo stereotipo include, significa che l'esecuzione del primo richiede

necessariamente anche l'esecuzione del secondo use case. Di riflesso nell'Activity Diagram del primo use case, indipendentemente dalle scelte dell'utente o dal comportamento dell'applicazione, l'attività collegata allo use case include deve essere sempre eseguita.

Al contrario, se uno use case è collegato ad un altro use case attraverso lo stereotipo extend, nell'Activity Diagram del primo use case deve esistere almeno una traccia di esecuzione che includa l'attività dello use case associato.

Sequence Diagram

Gli scenari di esecuzione vengono associati a ciascun caso d'uso per descrivere la dinamica dell'esecuzione del caso d'uso nei principali casi possibili. In genere vi sarà uno scenario di funzionamento normale, in cui il caso d'uso viene svolto completamente e giunge a buon fine e una serie di scenari che descrivono le possibili anomalie nel comportamento dell'utente o del sistema nel fornire i servizi richiesti.

I diagrammi di sequenza (chiamati anche diagrammi di interazione nel seguito) in UML sono un tipo di diagramma che consentono di rappresentare le interazioni tra oggetti nel sistema, collegando gli oggetti tramite le richieste di servizio tra gli oggetti stessi. Gli oggetti vengono rappresentati, come nell'esempio di fig. 1.16, con barre verticali a cui viene associato un nome (di classe o di singola istanza di una classe, e cioè un oggetto).

Nel caso degli scenari associati ai casi d'uso l'interazione è tra gli utenti esterni (attori) e il sistema. Gli attori potranno essere classificati in più classi di appartenenza.

Il diagramma di interazione va quindi letto dall'alto in basso, e riporta le interazioni (indicate come messaggi di richieste di servizio inviate da un attore all'altro) in ordine temporale. E' possibile, ma non obbligatorio, indicare le risposte ottenute a fronte delle richieste.

Nell'esempio di fig. 1.16 si illustra lo scenario di consultazione del catalogo dei prodotti (all'interno di mostrare configurazione computer standard) da parte del cliente. Il cliente prima sceglie di esaminare il catalogo(esamina-catalogo), quindi seleziona una categoria di prodotto (seleziona-categoria). Nello scenario vengono indicati anche eventuali messaggi relativi alla consultazione di più pagine del catalogo (per la sequenza delle pagine i messaggi sono: next, previous, first, last), infine è possibile effettuare la selezione di un prodotto (seleziona-prodotto).

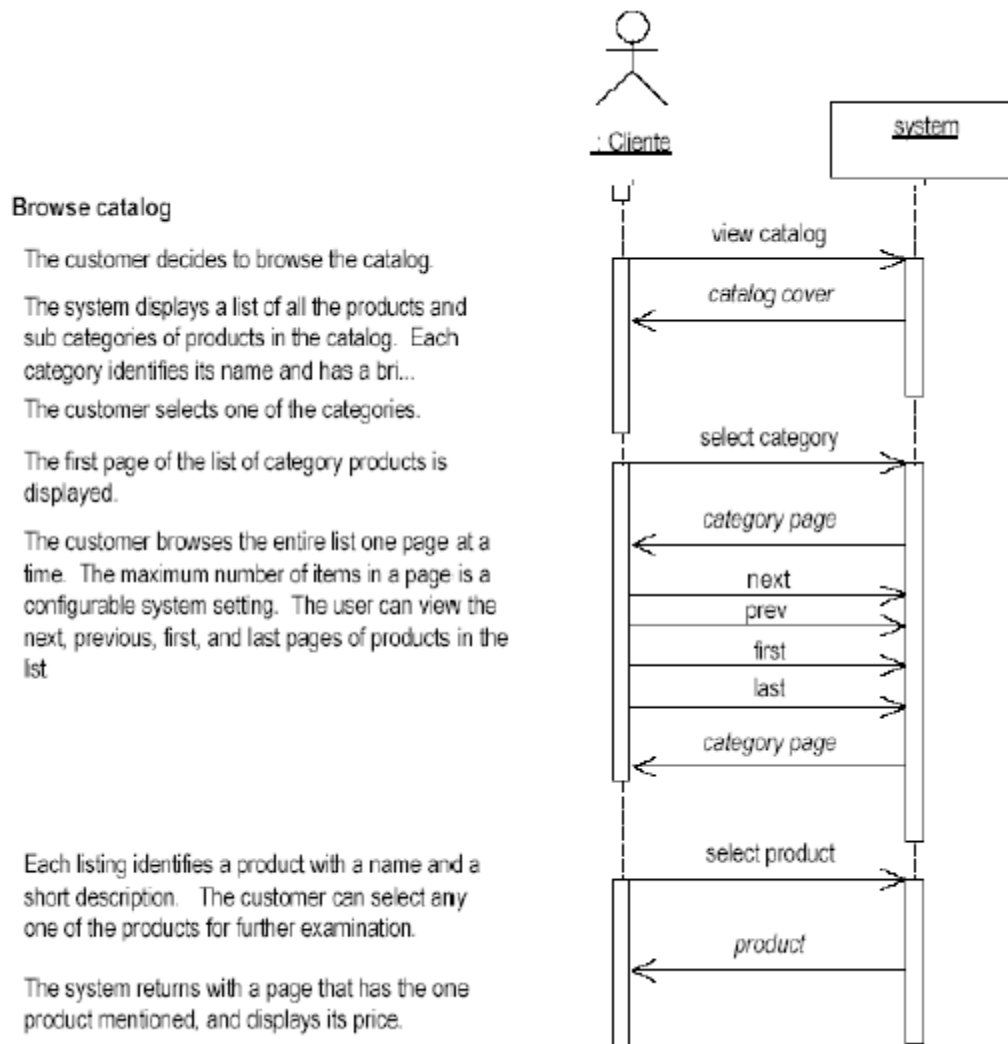


Figura 1.16: Esempio di diagramma di interazione

6.3.4 Esperienza dell'utente

La documentazione iniziale prodotta a seguito dell'analisi dei requisiti comprende anche la rappresentazione dell'interazione degli utenti con il sito e in particolare per i sistemi informativi basati su web questo consiste nel rappresentare in questa fase la navigazione nel sito e le pagine principali, con l'indicazione della sequenza delle interazioni previste. Si vuole quindi rappresentare, ad alto livello, come l'applicazione dovrà essere vista dall'utente finale in termini di schermate, maschere e percorsi di navigazione.

A questo scopo, si adotterà nel seguito il modello UX (user experience) o modello della navigazione. Tale modello è un diagramma delle classi che viene utilizzato per descrivere percorsi di navigazione tra schermate tramite le associazioni tra classi. Rispetto ad un classico diagramma delle classi, sono definiti tre stereotipi:

- *Schermata* (<< screen >>);
- *Parti di schermo* (<< screencompartment >>) [una schermata può essere composta da più parti di schermo, quali ad esempio banner o parti fisse; parti comuni a più pagine si incorporano in classi a parte.- per semplicità in seguito non tratteremo questo stereotipo];

- *Input form*(*<< inputform >>*).

screen

Una classe con lo stereotipo screen rappresenta l'astrazione di una pagina così come è vista dall'utente. I suoi attributi rappresentano i dati dinamici forniti all'utente. I dati statici non sono rappresentati. I suoi metodi rappresentano le operazioni di navigazione in grado di cambiare lo stato dell'applicazione. Nella figura vengono adottate alcune convenzioni di rappresentazione:

- “nomi classi”: con prima lettera in maiuscolo, indicano lo scopo della pagina;
- “\$”: significa pagina a cui si può ritornare da ogni pagina, tipicamente la home page. Sono le pagine che nella grafica del sito saranno sempre raggiungibili tramite una barra di navigazione generale. La home page spesso sarà raggiungibile tramite un logo che sarà posizionato in alto a sinistra nel sito o con un link con riferimento esplicito a tale pagina.
- “+” (scrollable): indica una sequenza di pagine che hanno la stessa struttura. Viene adottata per pagine che sarebbe troppo lunghe come contenuti e vengono spezzate per migliorarne i tempi di caricamento e la leggibilità (si pensi alle pagine restituite come risultato dai motori di ricerca). Per loro natura, queste pagine devono avere tra i propri metodi le funzionalità next e previous che permettono la navigazione all'interno della lista di informazioni che sono state spezzate in più pagine.

Successivamente si introducono dettagli relativi alle classi. Vengono indicati gli attributi che contribuiranno alla pagina con informazioni fornite dinamicamente. Ad esempio, nella schermata Product vengono fornite informazioni sul singolo prodotto, quale identificatore, nome, descrizione del prodotto e così via. Non vengono invece indicate informazioni relative al contenuto fisso della schermata, come figure o testo fissi.

Come detto in precedenza, gli attributi di una schermata rappresentano i dati dinamici. Questi dati, quindi, sono ottenuti da elaborazioni e attinti dalla base dati a cui l'applicativo fa riferimento ed è pertanto possibile esplicitare questa relazione. Partendo dai dati rappresentati nello schema E-R possibile derivare delle classi in grado di rappresentare i dati a disposizione. Collegando attraverso una relazione di aggregazione una schermata con una classe di questo tipo, si esplicita il fatto che i dati presenti nella classe sono automaticamente inclusi all'interno della schermata.

Passando ai metodi di una schermata, per ogni operazione eseguibile dall'utente, deve essere previsto un metodo. Solitamente, a fronte dell'esecuzione di un metodo, come si vedrà nella discussione sul modello UX, il controllo passerà ad un'altra schermata.

input form

Indica una pagina che consente l'interazione da parte dell'utente, inserendo informazioni che consentono di effettuare operazioni nel sito.

Gli attributi di una input form rappresentano i campi della form, mentre nessun metodo deve essere presente.

UX-Model

Come per tutti i diagrammi, il livello di definizione dei dettagli è progressivo nel corso della progettazione nelle diverse fasi e nell'iterazione del ciclo di sviluppo. Come indicato in fig. 1.18, vengono inizialmente definite le principali classi che consentono la navigazione nel sito. Nell'esempio, viene mostrato come si giunge alla selezione dei prodotti selezionandoli tra quelli nel catalogo tra quelli di una data categoria.

La costruzione dello UX Model parte dalle schermate e dalle input form che sono state individuate, e introduce i cammini di navigazione tra di esse in termini di associazioni tra schermate o input form. In particolare, si individuano due tipi di cammini: statici e dinamici. Un cammino statico è un semplice link, senza nome, tra pagine che non modifica lo stato dell'applicazione.

Va ricordato che la notazione \$ consente di non mostrare tutti i link a questa tipologia di pagine.

I cammini dinamici, al contrario, sono percorsi ogniqualvolta che l'utente invoca uno dei metodi definiti in una schermata. Data una schermata, per ogni metodo definito deve esistere una associazione, cui è assegnato un nome pari al nome del metodo, che parte dalla schermata stessa e raggiunge la schermata a cui viene passato il controllo. Può anche accadere che il link punti alla schermata stessa: si pensi ad esempio all'invocazione dei metodi next() e previous().

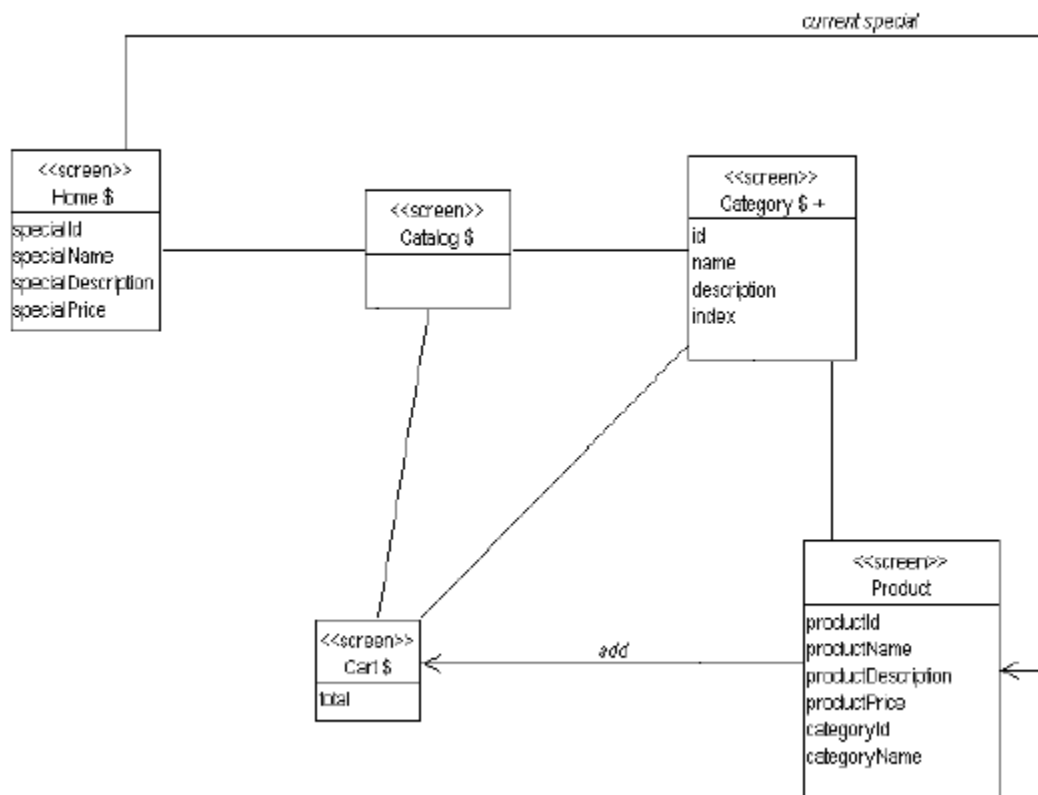


Figura 1.18: Esempio di UX model - selezione dei prodotti