Politecnico di Torino

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Gestionale

Il Project Management nel mondo dell'ICT

Applicazione della metodologia SCRUM al progetto CNHi Sales App

Relatrice:
Prof.ssa Anna Corinna Cagliano

Candidato:
Mauro Delli Noci

A.a. 2020/2021
Sessione di Laurea Luglio 2021

Indice

Introduzione

Il mondo attuale è caratterizzato da cambiamenti, incertezza e complessità crescenti, oltre che da estrema variabilità. Le aziende hanno difficoltà ad individuare vantaggi competitivi e metodologie atte al miglioramento continuo. Tali complessità ed incertezza sono accentuate nel mondo dell'Information Technology (IT) e dello sviluppo software.Una soluzione a queste problematiche potrebbe essere individuata nell'adozione di metodologie di lavoro e gestione dei processi "agili", che permettano alle aziende di avere maggiore flessibilità e capacità di adattamento.

Il presente elaborato di tesi si focalizza sull'effettivo utilizzo di pratiche di Agile Project Management (i.e., metodologia SCRUM) per lo sviluppo di un progetto IT di grandi dimensioni per un'azienda internazionale operante nel settore dell'automotive.

Nei capitoli componenti l'elaborato verranno dapprima introdotti i principi cardine e le teorie del Project Management classico e, a seguire, quelli relativi all'approccio Agile ed in particolare alla metodologia SCRUM. Verranno in seguito descritte le varie fasi di vita del progetto con focalizzazione sull'applicazione dei principi SCRUM alle stesse.

Lo scopo della tesi è valutare l'efficienza e la bontà dell'applicazione dell'Agile Project Management al progetto in esame, in contrapposizione ai risultati che sarebbe stato possibile ottenere tramite l'utilizzo del più classico approccio Waterfall per la gestione dei progetti.

1. Il Project Management

Nel presente capitolo verranno introdotti i principi base del Project Management, le sue caratteristiche principali e gli strumenti più diffusi. Saranno inoltre affrontati temi specifici quali il Project Management in organizzazioni che lavorano nel mondo dell'Information Technology e le metodologie Agile, oggetto di analisi in questo elaborato di tesi.

1.1 - Introduzione al Project Management

Citando il Project Management Institute, ente riconosciuto a livello internazionale come il più autorevole in materia di Project Management, si definisce Project Management l'applicazione di conoscenze, abilità, strumenti e tecniche alle attività di progetto, per soddisfare i requisiti dello stesso [1]. E' necessario dunque definire cosa si intenda per progetto.

1.1.1 Il Progetto

Come progetto si definisce la combinazione di risorse umane e non, riunite in un'organizzazione temporanea col fine di raggiungere un obiettivo definito con risorse limitate, in presenza di vincoli temporali, economici e qualitativi [1].

Come si evince dalla definizione fornita, un progetto deve essere svolto nel rispetto di vincoli specifici dati dalle seguenti variabili:

- Tempo, deve finire entro i limiti temporali previsti
- Costo, deve terminare rispettando i costi preventivati
- Ambito, deve produrre quanto previsto, nulla di più, nulla di meno
- Qualità, deve rispettare il livello qualitativo previsto
- Risorse, necessita di risorse adeguate e disponibili per essere completato
- Rischi, deve governare e gestire rischi e incertezze
- Benefici, deve creare valore [1]

Volendo rappresentare figurativamente un progetto, questo risulterebbe assumibile ad una piramide a base triangolare (Fig. 1.1), nella quale:

- I lati rappresentano i vincoli di qualità, tempi e costi del progetto
- La cuspide rappresenta il raggiungimento degli obiettivi di progetto
- La base rappresenta i rischi associati al progetto
- Il volume rappresenta l'ambito del progetto, il cosiddetto landscape, delimitato dai vincoli precedentemente citati

Figura 1.1 - La piramide di progetto [2]

Un progetto dunque, secondo lo Standard UNI ISO 21500:2013 è:

- Unico, per quanto ogni progetto possa essere classificato all'interno di una o più categorie (per tipologia, settore, importo, ecc.), ciascuno presenta condizioni di unicità e di non ripetitività. Un progetto che sia stato già realizzato in passato utilizzando le stesse risorse, coinvolgendo gli stessi soggetti (esecutori, fornitori e committenti) e producendo sostanzialmente lo stesso prodotto finale, non si qualifica come progetto, bensì come attività di produzione
- Temporaneo, deve avere una data d'inizio, una di fine e, di conseguenza, unadurata. Qualora non fosse possibile stimarne una durata, non si potrebbe parlare di progetto
- Inter-funzionale: spesso per arrivare al risultato finale è necessario utilizzare risorse e competenze proprie di differenti strutture funzionali
- Innovativo, un progetto introduce un cambiamento
- Rischioso, ad ogni cambiamento e innovazione è naturalmente legato un fattore dirischio ed incertezza.

Il progetto, infine, utilizza risorse umane, materiali e finanziari ed è caratterizzato da un'elaborazione progressiva: si comincia quasi sempre da zero (è presente solo un bisogno o un desiderio) e, progressivamente ci si avvicina e, infine, si consegna il prodotto o servizio finale in grado di soddisfare il bisogno o desiderio. (Cantamessa et al., 2007).

I progetti possono essere di due differenti tipologie:

- Progetti Esterni, i quali hanno come obiettivo la fornitura a terzi di prodotti o servizi
- Progetti Interni, i quali sono legati all'esigenza dell'azienda che li intraprende e non hanno un committente esterno

Tutti i progetti presentano un ciclo di vita che ne definisce l'inizio e la fine, ed è fondamentale per l'identificazione e l'organizzazione del lavoro.

Il ciclo di vita di un progetto risulta normalmente composto da:

- Aspetti gestionali, o processi di Project Management
- Aspetti tecnici, o processi di prodotto

Tali processi sono costanti per tutti i tipi di progetti e tra di essi paralleli, come mostrato in Figura 1.2 (Cantamessa et al., 2007).

Figura 1.2- Ciclo di vita del progetto (Cantamessa et al., 2007)

1.1.2 Il Project Management

Il Project Management viene utilizzato per progetti aventi le seguenti caratteristiche (Cantamessa et al., 2007):

- Complessità e difficoltà
- Obiettivi specifici
- Tempi e costi stabiliti

Gli obiettivi principali che le organizzazioni si propongono di raggiungere grazie all'uso del Project Management sono i seguenti:

- Obiettivi esterni, che hanno impatto sulle relazioni con il cliente ed il mercato e sono:
    - Il miglioramento continuo delle comunicazioni con i clienti, mediante ottimizzazione della definizione dei bisogni e delle richieste del cliente
    - La riduzione dei rischi di insuccesso
    - Il miglioramento della qualità, mediante un controllo più attento ed accurato dei processi ad opera del Project Manager
    - Il miglioramento del grado di soddisfazione dei clienti
    - Il miglioramento dell'immagine aziendale
- Obiettivi interni, che hanno impatto sui processi interni all'azienda e sono:
    - Il miglioramento delle comunicazioni del gruppo di progetto, mediante l'uso di una terminologia comunque, di una struttura organizzativa chiara e di processi predefiniti
    - La riduzione degli errori ed il miglioramento della qualità di processo

- La riduzione dei tempi di realizzazione, mediante tecniche specifiche di utilizzo delle risorse
- La riduzione dei costi di progetto, mediante l'aumento della produttività, il coordinamento e la cura del cliente
- La creazione di una base dati delle leziioni apprese dai progetti, che innesca nell'organizzazione un processo di miglioramento continuo

Il processo di gestione dei progetti, precedentemente definito, può essere suddiviso nelle seguenti macro fasi (Stewart, 2008), come mostrato in Figura 1.3:

- Concezione o identificazione del progetto
- Definizione del progetto
- Pianificazione del progetto
- Programmazione
- Esecuzione
- Monitoraggio e controllo del progetto
- Chiusura del progetto

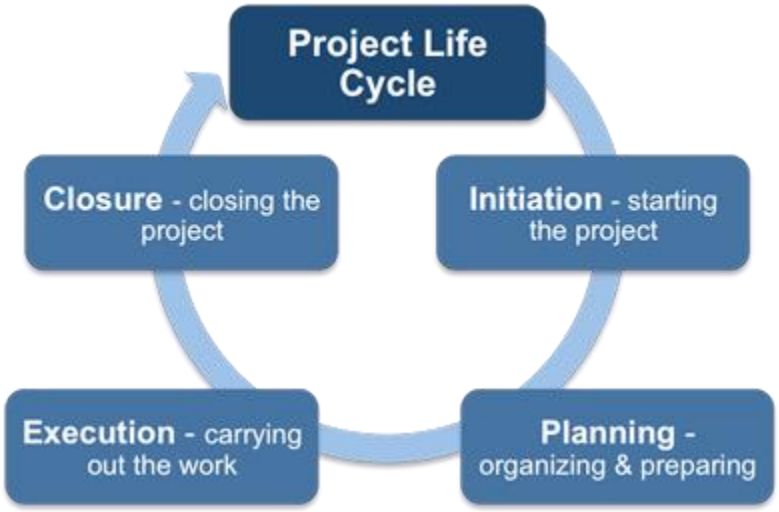

Figura 1.3 - Fasi di Project Management [3]

La fase di identificazione del progetto è quella fase nella quale, a partire da un bisogno o un problema, si sviluppa l'idea di un progetto interno o, partendo dalle richieste di un cliente, esterno. In questa fase si procede con uno studio di fattibilità del progetto e se ne valuta la convenineza economica.

Durante la fase di definizione del progetto si individuano i parametri caratterizzanti il nuovo progetto, partendo dalle valutazioni del mercato e dalle richieste del cliente. E' in questa fase che viene nominato il Project Manager e il Team di lavoro; vengono inoltre definiti gli obiettivi e le caratteristiche del prodotto da realizzare.

La fase di pianificazione del progetto è quella fase nella quale si individuano tulle le attività del progetto, l'ambito e le risorse necessarie. Si stimano inoltre i tempi e i costi del progetto e si

identificano e misurano i rischi. Il Project Manager e il Team di progetto devono stabilire, nel dettaglio, che cosa si deve realizzare, chi è responsabile di ciascuna attività operativa e come si intende operare. Tale fase prevede inoltre la pianificazione degli acquisti e delle forniture di progetto.

Il principale obiettivo della fase di programmazione del progetto è la schedulazione di tutte le attività componenti il progetto, individuate in fase di pianificazione, in modo da raggiungere gli obiettivi in un intervallo di tempo ragionevole. La programmazione comprende l'analisi della disponibilità delle risorse e l'applicazione di tecniche di schedulazione che verranno dettagliate successivamente.

La fase di esecuzione del progetto riguarda lo sviluppo e la realizzazione tecnica dei prodotti o servizi oggetto del sprogetto.

La fase di monitoraggio e controllo del progetto consiste nell'osservazione attenta dell'esecuzione dello stesso, con l'obiettivo di identificare dei potenziali problemi al fine di risolverli tempestivamente.

In tale fase:

- Si raccolgono ed elaborano le informazioni per capire lo stato del progetto ad una certa data
- Si definiscono gli indicatori di performance, in modo da misurare la performance del progetto confrontandola con gli obiettivi iniziali
- Si identificano le variazioni critiche fra performance attese e realizzate e se ne definiscone l'impatto sull'intero progetto e, se necessario, si prendono gli opportuni provvedimenti
- Si producono report continuativi sullo stato di avanzamento del progetto, in forma grafica o tabulare
- Si guidano ed ispirano i partecipanti al progetto a raggiungere gli obiettivi prefissati ad un livello che incontri o superi le aspettative

La fase di chiusura del progetto, infine, consiste nel passaggio di competenze e responsabilità da chi ha costruito l'opera a chi l'ha commissionata, nell'accettazione del progetto da parte del cliente e nella risoluzione di tutti i rapporti contrattuali che si sono instaurati con il progetto.

1.1.3 Il Project Manager

Volendo quindi definire definitivamente la figura del Project Manager, esso risulta essere la risorsa formalmente incaricata del raggiungimento degli obiettivi del progetto attraverso la corretta gestione dei processi di Project Management [5]. Esso gestisce e dirige le attività di progetto e ne è il responsabile ultimo del completamento.

Un progetto rappresenta una realtà molto diversa rispetto alle operazioni di routine di un ente e come tale necessita di un metodo gestionale diverso. Lavorare per progetti costituisce oggi sempre più la norma e non l'eccezione: negli ultimi anni il trend dominante è quello di organizzarsi sempre più per progetti, anziché secondo la più tradizionale programmazione "statica" di operazioni di routine produttiva.

Data, quindi, la crescente frequenza dei progetti e la ovvia crescente complessità che deriva da una realtà lavorativa sempre più in cambiamento, informatizzata e ora anche geograficamente diffusa, la riuscita di queste iniziative progettuali è data tutt'altro che per scontata, in particolare in assenza di una metodologia gestionale rigorosa. In sostanza, l'applicazione di una metodologia gestionale quale il Project Management serve a gestire al meglio la crescente complessità dei progetti, caratteristica della realtà attuale.

I vantaggi dell'uso del PM sono fondamentalmente due:

- Assumere impegni solo per obiettivi tecnici, di costi e di tempi definiti
- Pianificare, programmare e controllare ogni progetto in modo che gli impegni vengano effettivamente rispettati

L'adozione della metodologia del Project Management è dunque in generale consigliata in qualunque progetto il cui livello di complessità sia ritenuto non trascurabile, in quanto offre una maggiore sicurezza della buona riuscita dei progetti e una maggiore garanzia di successo (Lappe *et al.*, 2014).

1.2 IT Project Management

Il Project Management risulta essere una componente chiave per il successo delle imprese che operano nel settore dell'Information Technology (IT) (Stewart, 2008). Alcuni autori hanno affermato che, sotto determinate condizioni, l'applicazione delle pratiche di Project Management nel settore IT può risultare fonte di vantaggio competitivo per le organizzazioni (Basu and Muylle, 2007). L'IT Project Management può essere definito come un approccio strutturato alla pianificazione, organizzazione, direzione e controllo dei progetti di Information Technology. Esso riguarda dunque la gestione dell'effort totale necessario per l'implementazione di un progetto IT. Un progetto IT viene intrapreso con l'obiettivo di creare o implementare un prodotto o un sistema: esso comprende deliverables differenti quali software, hardware, data management, training e business process alignment (Hadaya et al., 2012).

1.2.1 Il Ciclo di Vita di un Progetto IT

Il ciclo di vita di un progetto IT delinea il lavoro tecnico e definisce i deliverables che sono richiesti per completare il progetto IT.

Basandosi sulla definizione fornita dal Project Management Institute (2008), esso include le seguenti fasi:

1. Requirements and analysis
2. Architecture
3. Design
4. Construction
5. Integration and test
6. Delivery

Il ciclo di vita di un progetto IT descritto risulta essere differente rispetto a quello relativo al Project Management di progetti standard (i.e., definizione, pianificazione, esecuzione, monitoring e chiusura). Tuttavia, essi risultano collegati e paralleli ed è dunque necessario l'utilizzo di entrambi per la gestione di un progetto IT: ciò perché il secondo include le fasi necessarie per l'organizzazione del progetto e la garanzia che esso proceda in maniera efficiente dall'inizio alla fine (Hadaya et al., 2012).

Fase 1: Requirements and analysis

La raccolta dei requisiti e la loro analisi compongono la prima fase richiesta per l'esecuzione di un progetto IT. I requisiti rappresentano il problema che deve essere risolto e gli obiettivi del progetto. L'analisi è condotta al fine di individuare le relazioni che intercorrono tra le componenti (i.e., funzionalità) del prodotto o del sistema relativo al quale sono stati raccolti i requisiti.

Durante tale fase, il Project Manager lavora a stretto contatto con il cliente e con gli esperti tecnici della materia in scopo del progetto, al fine di definire i requisiti provenienti dal business e determinare come essi possano essere soddisfatti. E' compito del Project Manager inoltre l'identificazione dei prodotti o sistemi che verranno consegnati al cliente, delle risorse necessarie alla realizzazione del progetto e delle skills richieste per il completamento dello stesso [4].

Fase 2: Architecture

La fase di definizione architetturale è richiesta per l'identificazione degli elementi che saranno inclusi nel prodotto o sistema che verrà consegnato al cliente. L'esperto di architettura rivede i requisiti provenienti dal committente ed identifica le possibili soluzioni da un punto di vista tecnico. Gli stakeholders di progetto analizzano le opzioni proposte e scelgono quella che ritengolo la soluzione più efficiente ed effetiva [4].

Fase 3: Design

La fase di design della soluzione da fornire al cliente si compone di due parti distinte: design high-level e design detailed.

Il design high-level è necessario per definire come le componenti del prodotto o sistema che verrà fornito funzioneranno da un punto di vista tecnico, quali saranno le interazioni di esse con gli altri componenti del sistema e come interagiranno con hardware e software.

Il design detailed è invece di tipo descrittivo e fornisce dettagli puntuali per ogni componente del prodotto o sistema. Esso identifica e definisce ogni modulo che appartiene ad ogni componente del prodotto o sistema, descrivendone la funzionalità, come lavorerà in dettaglio, e quali sono le relazioni di dettaglio che intercorrono tra moduli differenti [4].

Fase 4: Construction

La fase di costruzione del prodotto o sistema definito nella fase di design è l'attività di sviluppo vera e propria e coincide con la scrittura del codice e la costruzione delle componenti da parte del Team di sviluppatori [4].

In tale fase è in capo al Project Manager il processo di controllo del progetto in fase di sviluppo: esso coincide con la fase di monitoraggio e controllo caratteristica del ciclo di vita del Project Management applicato a progetti standard.

Il controllo del progetto include un insieme di attività e processi atti all'individuazione ed al monitoraggio del progresso e delle performances del progetto ad alla identificazione delle aree che richiedono un eventuale modifica. Esso può essere definito come un processo continuo che richiede al Project Manager capacità di osservazione, raccolta di informazioni e comprensione della necessità di cambiamento in caso di necessità.

Come definito nel capitolo di dettaglio, il processo di controllo ha come obiettivo la comparazione delle performance attuali di progetto con quelle pianificate, al fine di comprendere se il progetto è sotto controllo. Tale fase è inoltre necessaria per l'identificazione e il tracciamento di nuovi eventuali rischi e problematiche di progetto.

Un altro metodo utilizzato per determinare se il progetto sia sotto controllo è quello del project status reports. Il Project Manager pianifica degli incontri regolari con il Team di progetto al fine di raccogliere informazioni sullo status degli sviluppi da ogni gruppo di lavoro distinto che è coinvolto alla realizzazione del progetto. In questo modo il Project Manager è in grado di identificare eventuali problematiche insorte che potrebbero richiedere azioni correttive.

Fase 5: Integration and Test

La fase di integrazione consiste nella combinazione di tutti i moduli e componenti realizzati in un singolo prodotto o sistema funzionante in maniera tale che esso possa essere testato. La fase di testing è cruciale per lo sviluppo di progetti IT è atta alla determinazione della qualità del prodotto o sistema fornito.

Si possono individuare cinque distinte tipologie di test:

1. Unit and function test
2. System test
3. Integration test
4. No regression test (NRT)
5. User Acceptance test (UAT)

La fase di Unit test consiste nel testing dei moduli in maniera individuale ed è in capo agli sviluppatori che hanno prodotto quegli specifici moduli. In seguito alla scrittura del codice, il membro del Team che ha sviluppato il codice stesso lo testa, in modo da esser sicuro che il modulo funzioni a dovere. Dopo aver verificato che tutti i moduli sviluppati per un singolo componente siano testati ed abbiano superato con successo i test, il Team tutto conduce i Function test sul componente stesso, per verificare che funzioni.

In seguito al completamento della fase di Function test, si passa alla fase di System test sulle principali funzioni del prodotto o sistema. In questa fase il Development Team testa il codice prodotto utilizzando hardware o software differenti rispetto a quelli utilizzati per la produzione del codice stesso.

Terminata la fase di System test, si passa alla fase di Integration test. Tale fase prevede il testing di tutte le interfacce prodotte, testate come parte dell'intero sistema in scopo. Tutte le componenti vengono integrate nel nuovo sistema o in un sistema esistente (i.e., in caso di implementazioni su un prodotto già rilasciato), ed il sistema viene testato nella sua completezza. L'obiettivo è quello di verificare che tutto ciò che è stato sviluppato funzioni in maniera corretta all'interno del sistema completo che verrà fornito al cliente. Tale fase viene definita, per tale motivo, anche end-to-end testing, in quanto il sistema sviluppato è testato dall'inizio alla fine.

Parallelamente agli Intergation test si sviluppano i No regression test. Essi si effettuano solo nei casi in cui il nuovo codice sviluppato sia integrato in un'applicativo già esistente e hanno come obiettivo quello di verificare che le vecchie funzionalità disponibili non siano state danneggiate dal nuovo codice introdotto.

Superate entrambe le precedenti fasi di test con esito positivo, il sistema è fornito al cliente, ancora in ambiente di certificazione, per permettere agli utenti final di effettuare dei test del nuovo sistema seguendo i propri processi di business. Tale fase viene definita di User Acceptance Test e l'esito positivo della stessa è fondamentale al fine di ottenere l'approvazione per il Go-Live [4].

Fase 6: Delivery

La fase di Delivery del prodotto o sistema sviluppato consiste di molteplici task ed attività chiave, necessarie affinchè la consegna dell'applicativo al cliente finale avvenga in maniera efficiente e con successo.

E' possibile distinguere tra due differenti fasi della stessa, caratterizzate da attività in capo al Project Manager che sono tra loro distinte: quelle necessarie prima del Go-Live dell'applicativo e quelle richieste in fase di Go-Live.

Le prime sono le seguenti:

- Creazione di un Deployment Plan dettagliato, necessaria per l'identificazione e il ricordo di tutti gli step critici in fase di delivery della funzionalità.
- Revisione del Deplyment Plan con i membri del Team di progetto, necessaria per l'individuazione di eventuali task che sono stati sottovalutati o addirittura dimenticati, evitando così costosi errori.
- Comunicazione delle milestone chiave del piano agli stakeholder di progetto, che consente al cliente di preparare adeguatamente e per tempo i propri Team alla nuova funzionalità.
- Formazione del Team di supporto, il quale deve essere informato sulla nuova funzionalità ed adeguatamente preparato per fornire il supporto necessario in caso di dubbi, domande o problematiche sollevate dagli utilizzatori finali in seguito al go-live.

Le seconde sono le seguenti:

- Monitoring del Go-Live, durante il quale il Project Manager ha il compito di controllare che il processo di rilascio dell'applicazione avvenga facilmente e senza intoppi. Attività tipiche di questa fase sono il restarting dei server, l'upload del codice e il deploy delle applicazioni mobile e sono in capo al Team di sviluppatori
- Follow-up con il Team di supporto, necessario per essere messo a conoscenza di eventuali bug e problematiche insorte in seguito al rilascio la cui risoluzione deve essere prioritizzata.

1.3 I principali strumenti del Project Management

Il Project Manager ed il Team di progetto utilizzano nell'esecuzione del proprio lavoro svariati strumenti e tecniche utili in fasi di progetto differenti. I principali di essi verranno dettagliati di seguito.

1.3.1 Work Breakdown Structure (WBS)

La Work Breakdown Structure è una rappresentazione analitica del progetto che suddivide le attività livello per livello, fino al grado di dettaglio necessario per una pianificazione ed un controllo adeguati. Come mostrato in Figura 1.4, essa rappresenta una struttura gerarchica a forma di albero del progetto, che comprende tutti gli elementi che costituiscono oggetto di fornitura al cliente, nonché i principali compiti funzionali necessari per la loro realizzazione. Risulta essere inoltre un importante strumento di comunicazione verso i vari stakeholder chiave di progetto (Miller, 2008).

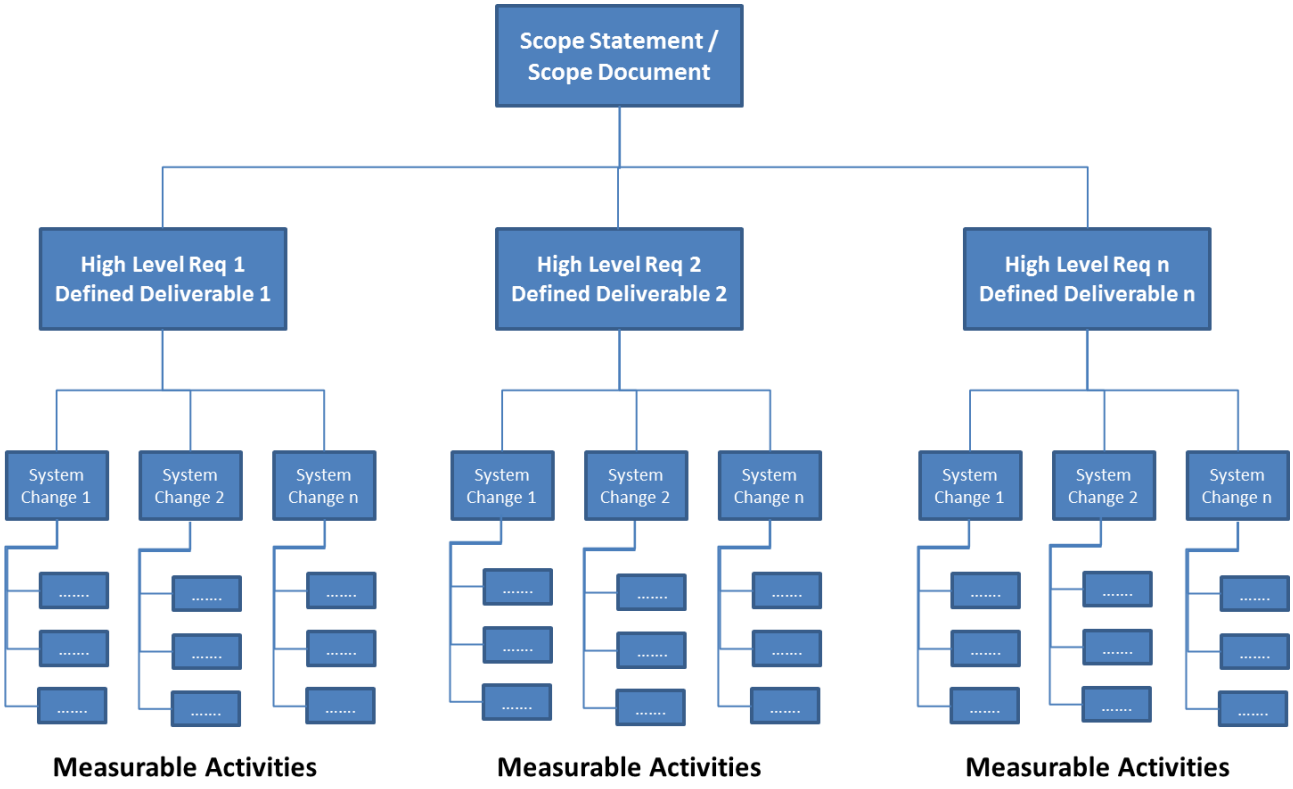

Figura 1.4 – WBS [6]

Tale documentazione, prodotta in fase di pianificazione del progetto, consente di definire in dettaglio che cosa debba essere realizzato e come . Essa inoltre permette di non trascurare alcuna attività, è un'ottima base per il calcolo dei costi di progetto e rende possibile un controllo agevole delle attività in corso di svolgimento.

Le attività principali necessarie alla realizzazione della WBS sono differenti. Il punto di partenza è la scomposizione e gerarchizzazione delle parti componenti il progetto (i.e., attività). In seguito è necessario individuare i compiti noti e ripetitivi, che saranno considerati come distinti work package.

Possono essere utilizzati differenti tipologie di logiche nella scomposizione delle attività:

- Logica funzionale, vengono identificati i sotto-sistemi dell'opera e dell'impianto che sono funzionalmente completi e collaudabili in modo indipendente. E' definita anche per deliverable o sotto-progetti
- Logica spaziale, la scomposizione si effettua in riferimento alla planimetria dell'opera o dell'impianto
- Logica dei processi di lavoro, il progetto viene disaggregato in funzione dei processi di lavoro necessari per la realizzazione dell'opera
- Logica della scomposizione fisica, la scomposizione si effettua in base alle parti costituenti, analogamente ad un'operazione di smontaggio di un prodotto
- Logica degli obiettivi, la scomposizione si esegue secondo milestones collocate nel tempo

Al termine della scomposizione gerarchica del progetto si ottengono i work package o compiti: essi sono dei "paccheti di lavoro" da realizzare chiaramente identificabili ed attribuibili univocamente ad una funzione o centro di lavoro aziendale.

I work package devono possedere un livello di dettaglio tale da consentire la definizione dei relativi costi, tempi, vincoli e avanzamenti, sia in termini di quantità che di costi.

Le informazioni associate ad ogni work package sono difatti le seguenti:

- Descrizione, cosa si deve fare
- Responsabile, chi deve farlo
- Committente, chi paga
- Costo, quanto costa
- Prodotti di input, cosa serve per eseguire l'attività
- Prodotti di output, cosa viene prodotto
- Tempi di realizzazione
- Attività necessarie per la realizzazione

1.3.2 Organizational Breackdown Structure (OBS)

L'Organizational Breakdown Structure è una scomposizione gerarchica delle responsabilità di progetto. Essa viene generata, in fase di pianificazione del progetto, allo scopo di individuare univocamente i responsabili o gli esecutori dei work package e di migliorare il flusso di comunicazioni di progetto (Velpuri *et al.*, 2011).

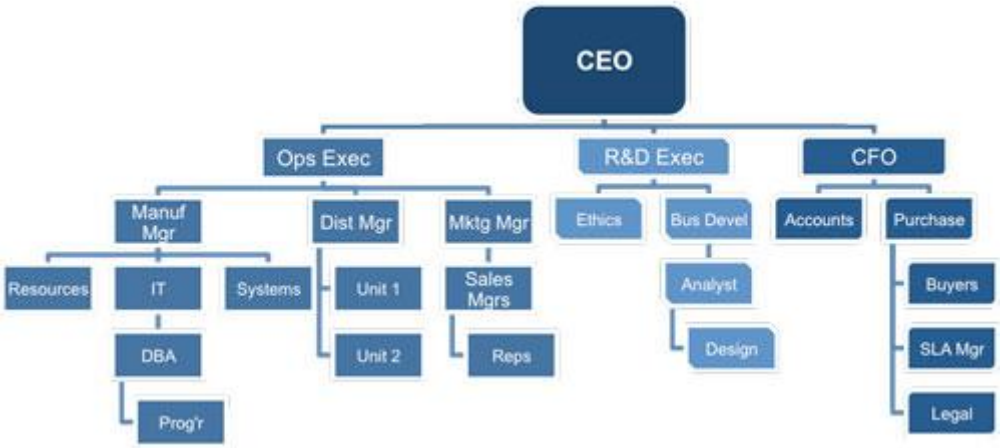

Figura 1.5 – OBS [6]

Ha lo scopo di agevolare il Project Manager nel suo lavoro di controllo e monitoraggio del progetto.

1.3.3 WBS/OBS

L'incrocio tra la WBS e la OBS dà origine ad una matrice compiti/responsabilità la quale ha lo scopo di rispondere alla domanda "chi fa che cosa". I singoli work package sono difatti assegnati ai centri di responsabilità correti (i.e., risorsa definita nella OBS) ed ogni intersezione tra un workpackage ed un centro di responsabilità determina l'individuazione di un compito (i.e., attività di progetto) (Cantamessa et al., 2007).

Definita la WBS/OBS generale di progetto, ogni funzione responsabile svilupperà poi in dettaglio i work package assegnati, definendo così delle WBS di dettaglio. Le attività così definite sono poi utilizzate per la programmazione della commessa, individuando per ognuna di esse la risorse necessarie ed i tempi di esecuzione.

L'abbinamento di un codice ad ogni work package definito consente un agevole controllo delle attività in corso di svolgimento ed è utile per essere utilizzato su supporto informatico.

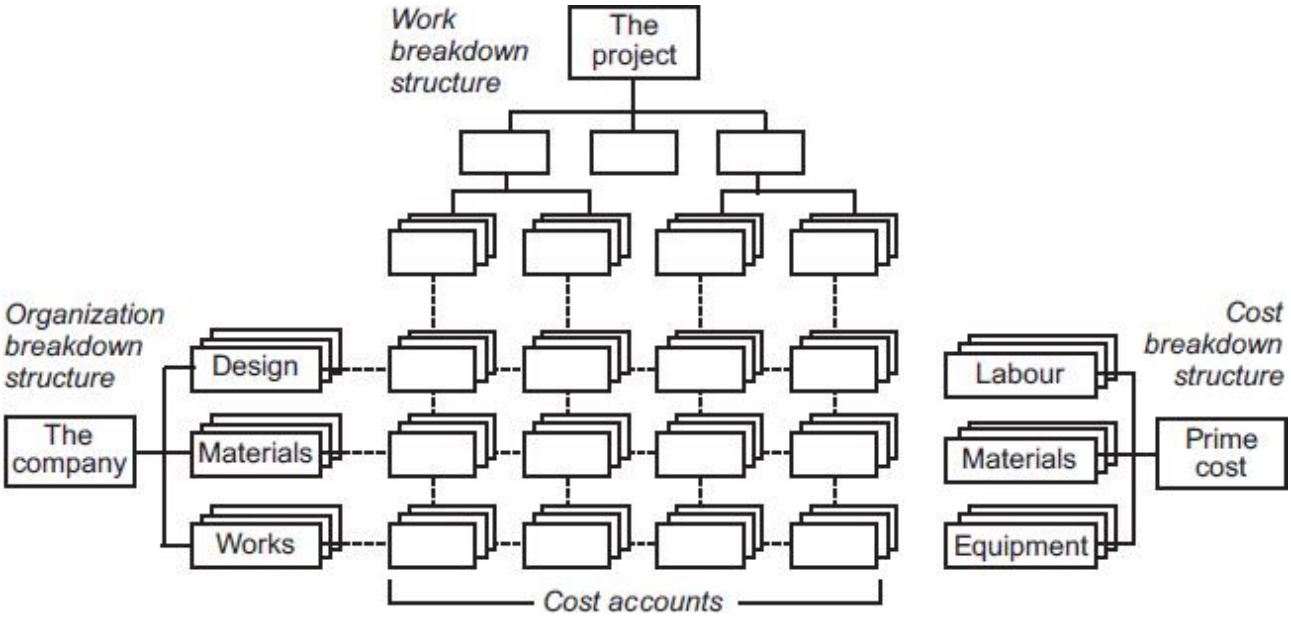

Figura 1.6 - WBS/OBS [8]

1.3.4 Cost Breakdown Structure (CBS)

La Cost Breakdown Structure è lo strumento utilizzato dal Project Manager per il calcolo dei costi di progetto. Essa ha come obiettivo la strutturazione dei costi in un schema dettagliato che consenta di attribuire ad ogni unità distinta il costo corretto (Cantamessa et al., 2007).

La scomposizione del progetto descritta nel paragrafo 1.3.3 si può estendere alla valorizzazione economica delle singole attività che consente di creare un'ulteriore struttura ad albero denominata, per l'appunto, CBS. Nella CBS vengono considerate le voci di costo direttamente legate al progetto o ad esso facilmente assimilabili.

Esistono due differenti approcci per stimare i costi di progetto:

- Approccio Top-Down
- Approccio Bottom-Up

Utilizzando l'approccio Top-Down si va a valutare il costo complessivo del progetto e lo si scompone attraverso iterazioni nelle varie componenti. Utilizzando stime parametriche e function point, il costo totale calcolato si ripartisce tra le fasi ed attività di progetto. Tale metodologia è quella di solito utilizzata per la preparazione di preventivi d'offerta.

L'approccio Bottom-Up è utilizzato invece per stabilire la baseline di costo. A partire dai costi delle singole attività previste per ogni work package, si costruiscono per aggregazione i costi globali di progetto.

Per determinare i costi di un singolo work package si determinano i costi delle risorse umane, nel dettaglio il costo orario di ogni risorsa umana impiegata viene moltiplicato per la durata dell'attività. Tali costi vanno sommmati così da determiinare i costi delle singole attività che costituiscono il work

package e, a questi ultimi, si aggiungono gli altri costi quali costo dei materiali, delle attrezzature, gli acquisti di beni e/o servizi e le trasferte.

In Figura 1.7 un esempio di CBS.

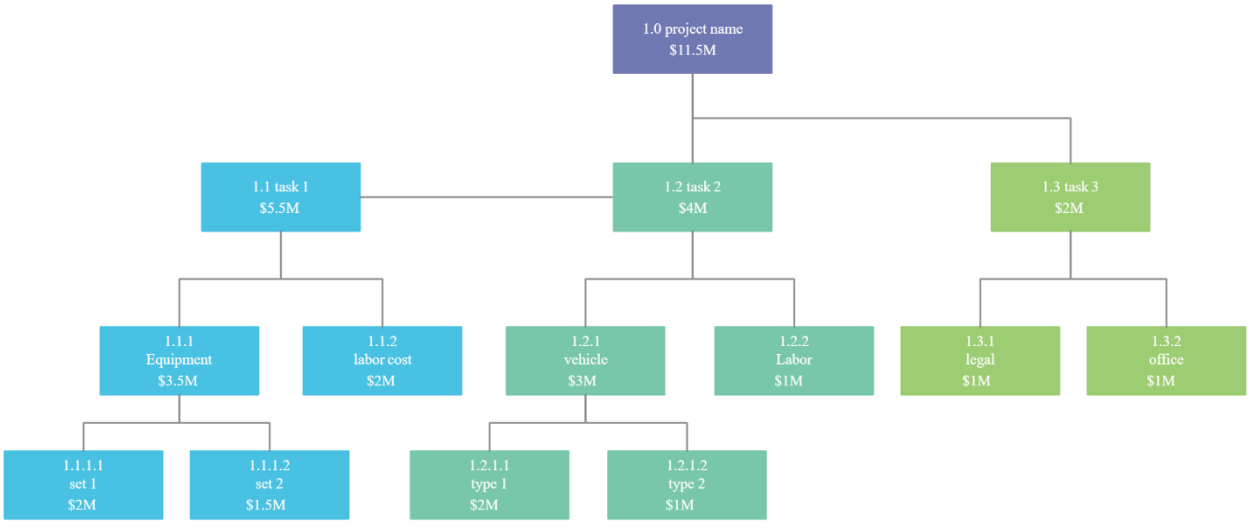

Figura 1.7 – CBS [9]

1.3.5 Strumenti di programmazione: Introduzione

In seguito alla fase di pianificazione del progetto, si passa alla programmazione dello stesso.

Tale fase consiste nella collocazione in un'unica scala temporale di tutti i pacchetti di lavoro componenti la WBS, tra loro logicamente correlati da relazioni di dipendenza operativa (Cantamessa et al., 2007).

I principali documenti di programmazione e reporting sono i seguenti:

- *Overall Master Schedule*, è il punto di contatto con il cliente, definito in fase d'offerta, presentato nel kick-off meeting e approvato in avvio di progetto. Rappresenta la baseline contrattuale e può essere aggiornato in presenza di varianti.
  Esso serve a fissare gli obiettivi di progetto, riportare le milestones contrattuali e fornire le linee guida per l'elaborazione dei programmi di dettaglio.
- *Project Schedule*, è redatto dal Project Manager nell'avvio del progetto assieme agli enti funzionali ed eventualmente al Project Office. Rappresenta la timeline operativa di progetto ed è costantemente aggiornata.
  Esso serve a redigere il programma mediante tecniche reticolari, riportare gli eventi di progetto, le priorità, le sequenze e la durata delle attività e fornire le date per l'elaborazione dei programmi di dettaglio sviluppati dai singoli enti aziendali.
- *Detailed Schedule*, sono le schedulazioni dettagliate conosciute solo dalle funzioni di competenza. Tra esse annoveriamo ad esempio l'engineering programme, il construction schedule e i piani di qualità

Gli elementi base per eseguire la programmazione del progetto sono i seguenti:

- Durata delle attività
- Relazioni di dipendenza tra attività
- Calendario temporale
- Milestones, intese come momenti significativi nel ciclo di vita del progetto

Esistono differenti tecniche di programmazione, ognuna delle quali costituisce uno strumento che permette al PM di coordinare in modo valido e consapevole le attività.

Le tecniche di programmazione possono essere *semplificate*, utilizzate nei progetti semplici o a breve durata o di tipo *reticolare*, utilizzate nei progetti complessi o di lunga durata.

Di seguito verranno descritte le principali tecniche di programmazione appartenenti a tali tipologie.

1.3.6 Tecniche di programmazione semplificata

Come precedentemente definito, le tecniche di programmazione semplificata vengono utilizzate per progetti semplici o di breve durata.

Le principali sono le seguenti:

- Elenchi di attività
- Diagrammi lineari o di Gantt

Elenchi di attività

E' la tecnica di programmazione utilizzata più semplice. Consiste nel riprendere le attività definite nella WBS e, in base alle stime delle risorse utilizzate derivanti dalla OBS, si quantificano i tempi di esecuzione di ogni attività. E' possibile definire solo le durate in termini di FTE (Full Time Equivalent) o inserire riferimenti al calendario. Le durate vengono sottoposte a verifiche periodiche per individuare eventuali scostamenti positivi o negativi di quanto preventivato.

Diagrammi lineari o a barre di Gantt

Il diagramma di Gantt è uno schema grafico che permette di visualizzare la durata delle attività nel tempo. In Figura 1.8 è mostrato un diagramma di Gantt.

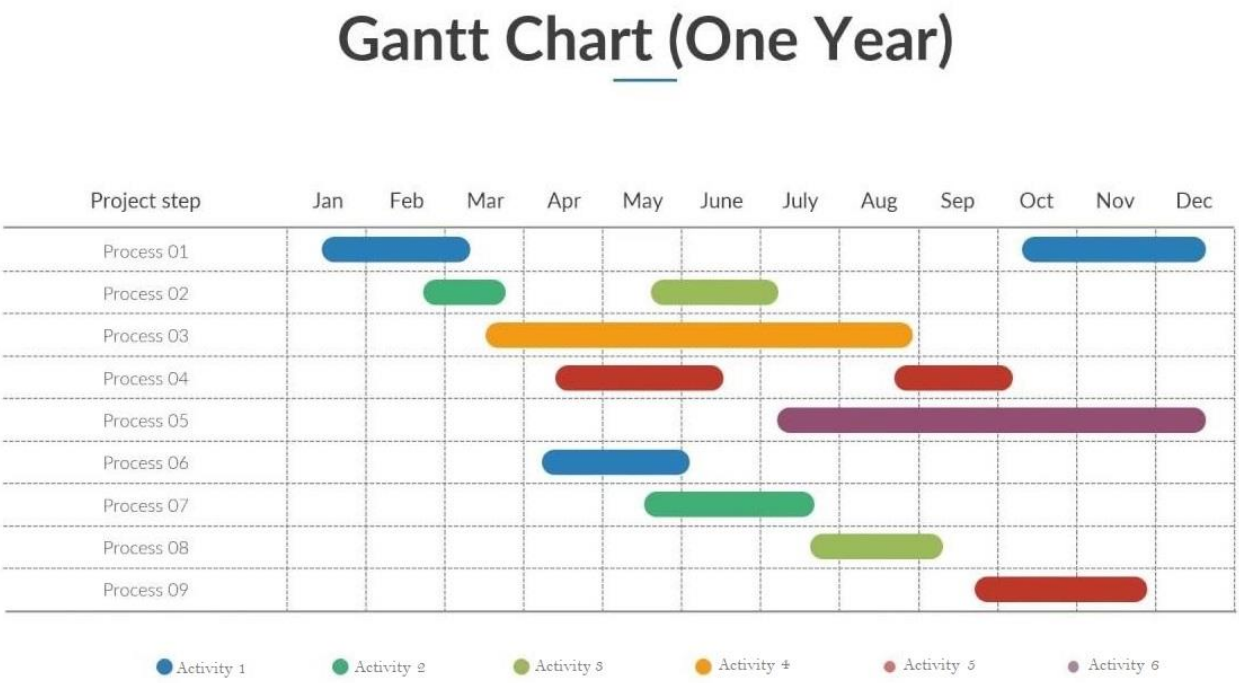

Figura 1.8 - Gantt Chart [10]

Per costruire il diagramma di Gantt si svolgono le seguenti attività:

- Si effettua l'analisi del progetto e lo si scompone nelle operazioni fondamentali corrispondenti alle attività della WBS
- Si assegna ad ogni attività la durata prevista inizialmente
- Si rappresentano le operazioni con dei segmenti a barre. Ogni attività viene rappresentata mediante segmenti di lunghesìzza proporzionale alla durata, in modo tale che la sequanza rispetti il reale sviluppo dei lavori nel tempo

1.3.7 Tecniche di programmazione reticolari

Le tecniche di programmazione reticolare vengono come detto utilizzate per progetti complessi o di durata considerevole. Esse si basano sulla teoria dei grafi e presentano quindi due elementi principali:

- Nodi, rappresentati con un punto o un cerchio
- Archi, rappresentati la relazione binaria tra i nodi

Esistono due differenti approcci per la rappresentazione dei reticoli di progetto, mostrati in Figura 1.9:

- Metodo Activity On Arrows (AOA), con il quale le attività sono rappresentate da frecce e gli eventi da nodi. Il verso della freccia rappresenta il verso positivo dello sviluppo temporale ed ogni attività è delimitata dagli eventi di inizio e di fine
- Metodo Activity on Node (AON), con il quale le attività vengono rappresentate da nodi, a cui vengono associate le rispettive durate e le frecce indicano i vincoli di precedenza temporale

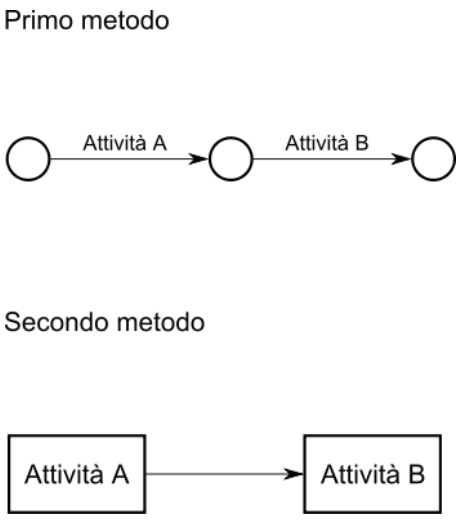

Figura 1.9 - AOA e AON [11]

I metodi reticolari permetto di definire il cosiddetto Cammino Critico di progetto o Critical Path. Esso è il tempo minimo di realizzazione del progetto ed è definito da tutte quelle attività sequenziali che presentano la durata maggiore. Di seguito verranno dettagliate le principali tecniche di programmazione reticolari.

Pert

Il PERT o Program Evaluation and Review Technique è un metodo creato nel 1958 dall'US Navy.

Lo scopo del PERT è:

- Stabilire un ordinamento delle attività
- Determinare il minor tempo possibile di realizzazione dell'obiettivo
- Individuare le operazioni critiche

Esistono due differenti tipologie di PERT:

- PERT probabilistico
- PERT-Costi

Il PERT probabilistico tiene conto dell'incertezza previsionale relativa ai tempi realizzativi delle attività.

Esso prevede dunque che per ogni attività non sia disponibile un unico valore della durata, ma che lo si debba ricavare da una distribuzione probabilistica. Le distribuzioni di probabilità più utilizzate sono quella normale, triangolare e la distribuzione Beta. La distribuzione Beta è quella che maggiormente rispecchia l'andamento dei tempi delle attività, ed è per questo quella più utilizzata.

Per ogni attività si individuano tre durate significative caratteristihce della distribuzione:

- Durata ottimistica (a), che corrisponde al tempominimo richiesto per l'esecuzione delle attività, nell'ipotesi che tutto vada per il meglio

- Durata modale (m), che corrisponde al tempo verificatosi con la massima frequenza per quell'attività
- Durata pessimistica (b), che corrisponde al maggior tempo possibile richiesto per l'esecuzione dell'attività

Il PERT-Costi si utilizza per valutare la possibilità di ridurre la durata del progetto mediante un impiengo più intensivo delle risorse, valutandone l'incidenza in termini di costo.

Le tre grandezze significative utilizzate per la stima del costo di ogni attività sono:

- Costo ottimistico (Ca)
- Costo modale (Cm)
- Costo pessimistico (Cb)

Tali grandezze sono assimilabili a quelle precedentemente descritte per il PERT classico.

In Figura 1.10 è mostrato un esempio di PERT.

Figura 1.10 - PERT [12]

Critical Path Method (CPM)

Il Critical Path Metod, creato nel 1958 da Dupont considera le relazioni tempo-costo di ogni attività, individuando il costo totale minimo di progetto e la relativa durata ottima.

Il CPM si articola in due diversi tipi di analisi:

- Prima analisi, di tipo reticolare, che consente di individuare le durate delle attività del progetto, similmente a quanto previsto col PERT. La differenza risiede nel fatto che le durate risultano essere deterministiche e non probabilistiche
- Seconda analisi, anche detta CPM Costi che consiste nell'approfondimento dell'analisi precedente mirato all'unificazione dei tempi e dei costi al fine di ottenere un migliore controllo del progetto

Per attuare il CPM bisogna dunque eseguire le seguenti attività:

- Costruzione del reticolo utilizzando i tempi normali, individuando per ogni attività il costo minimo diretto e totale (diretto più indiretto)
- Individuazione delle attività critiche con costi diretti unitari minori
- Riduzione dei tempi delle attività individuate precedentemente
- Ricavazione del costo minimo totale, attraverso iterazioni successive

In Figura 1.11 è mostrato un esempio di CPM.

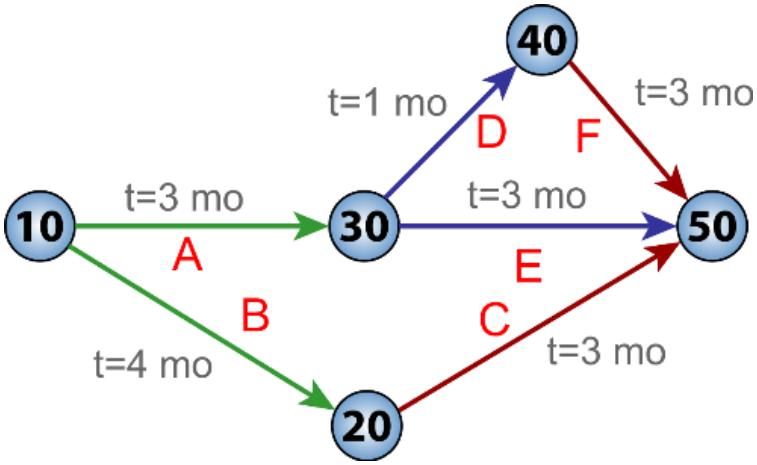

Figura 1.11 – CPM [13]

Precedence Diagramming Method (PDM)

Il metodo PDM o Precedence Diagramming Method risulta essere una evoluzione del CPM che ammette l'utilizzo di legami tra le attività diversi dal solo tipo Finish-to-Start (FS). Esso difatti presuppone l'esistenza di differenti tipologie di legami:

- Legame FS (Finish-to-Start), per il quale la fine di un'attività precedente si lega all'inizio di quella successiva
- Legame SS (Start-to-Start), per il quale l'attivitò seguente non può iniziare se non è già iniziata la precedente
- Legame FF (Finish-to-Finish), per il quale la fine di un'attività seguente dipende dal comportamento della precedente
- Legame SF (Start-to-Finish), per il quale la fine dell'attività seguente dipende dall'inizio della precedente

Il metodo PDM consente dunque di creare reticoli nei quali le attività siano eseguite in parallelo, completamente o parzialmente, invece che completamente in serie. In questo modo è possibile sovrapporre l'esecuzione di più attività e di comprimere la durata globale del progetto.

In Figura 1.12 è mostrato un esempio di PDM.

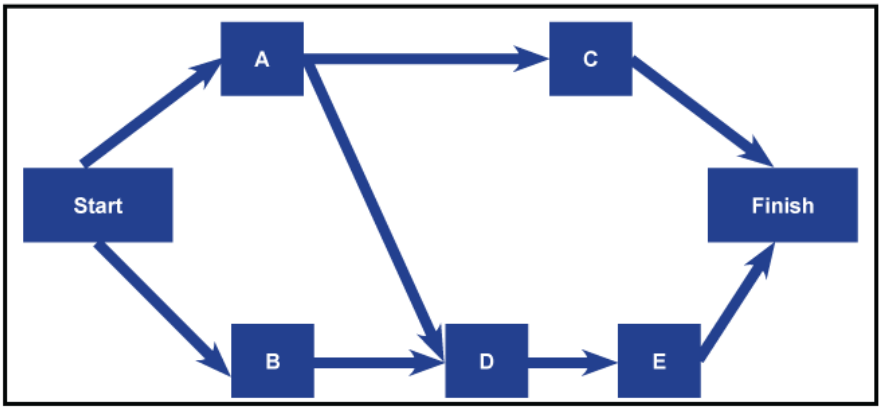

Figura 1.12 - PDM [14]

Critical Resource Diagram (CRD)

Il CRD o Critical Resource Diagram è un estensione del metodo CPM che consente di individuare la distribuzione delle risorse di progetto nel tempo, sviluppando dunque un diagramma a barre rappresentante la suddivisione delle stesse.

Le caratteristiche principali di un reticolo CRD, rappresentato secondo la convenzione AOA, sono le seguenti:

- Ogni arco identifica un'attività eseguita da una risorsa. Se più compiti devono essere svolti dalla stessa risorsa, allora compariranno nel reticolo più archi con lo stesso nome
- I nodi indicano i rispettivi eventi di inizio e di fine dell'attività svolta da un determinato tipo di risorsa

In un reticolo CRD inoltre possono essere indicate le durate temporali di ogni attività, in modo tale da fornire un ulteriore informazione sulle relazioni esistenti tra le risorse. Il cammino critico sche si ottiene da questo reticolo riguarda le singole risorse che possono dover eseguire più attività.

Critical Chain Method (CCM)

Il CCM o Critical Chain Method è un metodo di porgrammazione che tiene conto dell'influenza umana nello sviluppo dei progetti. Esso si basa sull'idea che si possa ottenere un miglioramento continuo delle prestazioni attraverso l'analisi e comprensione delle relazioni di causa-effetto che governano la realtà in esame (Goldratt, 1994).

Egli sostiene che nella programmazione delle attività si debba tener conto del comportamento umano delle persone che si dedicano al progetto. Tra i vari aspetti della psicologia delle persone egli ne individua quattro:

- Il calcolo delle stime: se richiesto ad una persona di indicare il tempo di esecuzione di un'attività sulla base della sua esperienza, essa esegue una stima che possiede un valore reale ed uno nascosto, aggiunto dalla stessa per cautelarsi
- La sindrome dello studente: pur avendo molto tempo a disposizione, si decide di svolgere una determinata attività quando rimane poco tempo, correndo il rischio che si verifichi un problema, costringendo così la persona a lavorare ad un ritmo maggiore per recuperare
- Legge di Parkinson: un'attività difficilmente verrà svolta in un tempo inferiore a quello assegnato, in quanto le persone tendono ad occupare completamente il tempo disponibile per il completamento, perché vanno ad un ritmo inferiore o per non evidenziare il fatto che per eseguire quell'attività fosse necessario minor tempo
- Multi-tasking: se una risorsa lavora a più progetti contemporaneamente potrebbe essere costretta a "spezzettare" le attività, passando da un porgetto all'altro, riducendo così l'efficienza

I passi metodologici per l'applicazione del CCM sono i seguenti:

- Scheduling Backwards: nel CCM il programma viene sviluppato a ritroso, a partire da una data di fine assegnata. Operando su un reticolo già costruito per l'assegnazione dei tempi alle attività, i soggetti sono meno propensi ad inserire riserve di tempo. Al termine di tale operazione si ottiene automaticamente una data di inizio progetto "as late as possible", ovvero l'ultima data disponibile per avviare il progetto senza ritardi
- Stima delle durate: consiste nel chiedere alle risorse di considerare nella stima della durata delle attività solo fattori positivi, ignorando quelli negativi, in modo da non inserire nella programmazione delle durate di riserva
- Identificazione della catena critica: la catena critica è la sequenza di attività con durata maggiore, considerando sia i vincoli di risorse che quelli di precedenza tra le attività. Il CCM riconosce che un'indisponibilità di una risorsa comunque tra più attività può causare un ritardo nell'esecuzione delle attività coinvolte

- Inserimento dei Buffer: per compensare la riduzione delle sicurezze nella programmazione delle singole attività si prevede l'inserimento di buffer temporali in alcune posizioni strategiche lungo il calendario di progetto.
  Esistono due tipologie di Buffer:
  - Buffer di progetto: inserito al termine della catena critica per proteggere la commessa da ritardi
  - Feeding buffer: inseriti per ogni cammino non critico nel punto in cui essi si inseriscono nella catena critica, per definire una protezione anche per le singole attività appartenenti alla stessa.

1.3.8 Aspetti economico-finanziari

Tempificazione dei costi

Una volta eseguita la pianificazione delle risorse occorrenti al progetto, quantificato il relativo fabbisogno economico e allocato il costo ad ogni singolo pacchetto di lavoro, si procede a spalmare nel tempo i costi definiti. Si utilizza come strumento una specifica curva ad S di progetto, anche definita baseline dei costi, la quale rappresenta l'aggregazione dei costi stimati delle singole attività o pacchetti di lavoro nell'arco temporale di vita del progetto.

Tale strumento consente in fase di monitoraggio di controllare i costi sostenuti effettivamente rispetto a quelli preventivati per ogni singola attività componenete il progetto.

Piano finanziario del progetto

Eseguita l'attività di tempificazione dei costi, è necessario individuare il fabbisogno finanziario del progetto.

L'obiettivo è l'individuazione delle spese di ogni singolo periodo, in confronto con i ricavi attesi, al fine di stabilire il fabbisogno di fondi. La pianificazione deve dunque tener conto del piano di fatturazione attiva e dei relativi incassi, e del piano di fatturazione passiva con i relativi esborsi legati alle spese dirette di progetto. Gli esborsi vengono calcolati sulla base della baseline di progetto precedentemente definita. L'uscita monetaria avviene in genere con una certa dilazione di tempo rispetto all'esecuzione delle attività, per tale ragione la curva delle uscite risulta essere spostata nel tempo rispetto alla baseline dei costi.

Relativamente alle entrate, bisogna considerare le milestone di fatturazione, ovvero quegli eventi ai quali si lega la possibilità di emissione di una fattura. Ad ognuno di tali eventi viene attribuita una percentuale del costo complessivo del progetto (e.g. il 20% all'accettazione, 30% alla consegna del primo prototipo, etc. etc.).

Per determinare il flusso di cassa del progetto ed il relativo fabbisogno finanziario, si porcede al confronto tra la curva del flusso di uscite di cassa e la curva rappresentate le entrate, che è generalmente una spezzata. Le aree al di sopra la spezzata degli incassi rappresentano il fabbisogno di fondi. Le aree al di sopra la curva delle uscite e sotto la spezzata degli incassi individuano invece le disponibilità finanziarie.

1.3.9 Approcci di controllo avanzamento della commessa

Il monitoraggio ed il controllo sono processi gestionali che intervengono durante l'esecuzione delle attività operative di progetto e che si sviluppano in maniera parallela ad esse.

L'obiettivo di tali operazioni è la verifica della qualità, dei tempi e dei costi delle attività che costituiscono il progetto.

Il monitoraggio prevede le seguenti attività:

- Misurazione dell'andamento del progetto
- Definizione degli indicatori di performance opportuni
- Misurazione delle performance di progetto in relazione ai costi sostenuti, ai tempi trascorsi ed alla qualità ottenuta
- Confronto delle misure affettive con quelle programmate
- Calcolo degli scostamenti
- Identificazione degli scostamenti critici o inaccettabili
- Definizione dell'impatto delle variazioni osservate sul progetto

Il controllo prevede le seguenti attività:

- Contenimento del progetto all'interno degli obiettivi iniziali
- Analisi degli scostamenti e comprensione delle cause
- Esecuzione di azioni correttive e preventive e valutazione dell'impatto delle stesse sul progetto
- Quantificazione delle modifiche
- Riprogrammazione del progetto e ridefinizione degli obiettivi
- Comunicazione agli stakeholder delle variazioni effettuate

L'attività di controllo, eseguite dal PM e dal suo Team con cadenza regolare, possono risultare *On/Off*, ovvero il PM può decider di bloccare il progetto per effettuare delle determinate verifiche di controllo, *Feedback*, ovvero l'oggetto del controllo è un'attività già conclusasi, o *Feed-Forward,* l'oggetto del controllo è un'attività in corso di realizzazione, al fine di correggere eventuali scostamenti rispetto alla pianificazione

Relativamente al controllo dei costi e delle performance di progetto, le tecniche di cost engineering utilizzate nella fase di valutazione degli scostamenti sono l'*analisi del planned value e dell'actual cost*, il *metodo dell'earned value*, l'*analisi degli scostamenti* e la *performance analysis*.

Negli ambienti lavorativi non si distingue tra valutazioni di monitoraggio ed azioni di controllo, ma si definisce un unico processo di Project Control. Il Project Control è un processo ciclico che consente di prevedere le conseguenze di avvenimenti passati ed intraprendere azioni correttive prima che tali conseguenze si verifichino. Esso si occupa di qualità (WBS/OBS e piani operativi), costi (CBS/BUDGET), tempi (GANTT/CPM/PERT) e del cash flow. Il Project Control è intrapreso sia dal

committente che dall'appaltatore, secondo prospettive differenti, in tutte le fasi operative di progetto (Cantamessa et al., 2007).

La curva ad S

Per misurare e controllare gli scostamenti dei valori effettivi delle variabili di progetto rispetto a quelli preventivati si possono utilizzare le curve a S.

Misurare gli avanzamenti significa attivare un processo di raccolta periodica delle informazioni. L'avanzamento percentuale di un'attività (progress) è definito come il rapporto tra la quantità eseguita e la quantità totale preventivata, valutata secondo la migliore stima condotta alla data attuale di:

- Ore lavorate/Ore totali
- Kg, m, mq realizzati/Kg, m, mq totali
- Quantità realizzate per costi unitari/Costi totali

Una volta valutato lo stato di avanzamento di tutte le attività della WBS, è possibile risalire alla misura dell'avanzamento complessivo di progetto per somma pesata. Il valore dei pesi percentuali, assegnato a ogni livello della WBS e calcolato in sede di pianificazione originaria, deve rimanere invariato. Per controllare gli scostamenti di tempi e costi, si prende come variabile il costo del progetto.

La curva ad S in questo scenario può rappresentare i costi cumulati di progetto nel tempo, in diverse situazioni:

- Preventivo, quanto preventivato di spendere
- Consuntivo, quanto effettivamente speso
- Earned Value, il valore effettivamente prodotto

Per realizzare tali curve è necessario tracciare un istogramma dove l'ascissa riporta le unità temporali e l'ordinata la variabile che si vuole tenere sotto controllo. Eseguendo un'integrazione si ottengono le curve a S a consuntivo e a preventivo.

Una volta costruite tali curve è possibile eseguire il controllo dei tempi, monitorando la differenza orizzontale tra le due curve (i.e., anticipo o ritardo nei lavori), e il controllo dei costi, monitorando la differenza verticale tra le due curve (i.e., eccesso o risparmio di spesa) (Maravas *et al.*, 2012).

In Figura 1.13 è mostrato un esempio di Curve ad S.

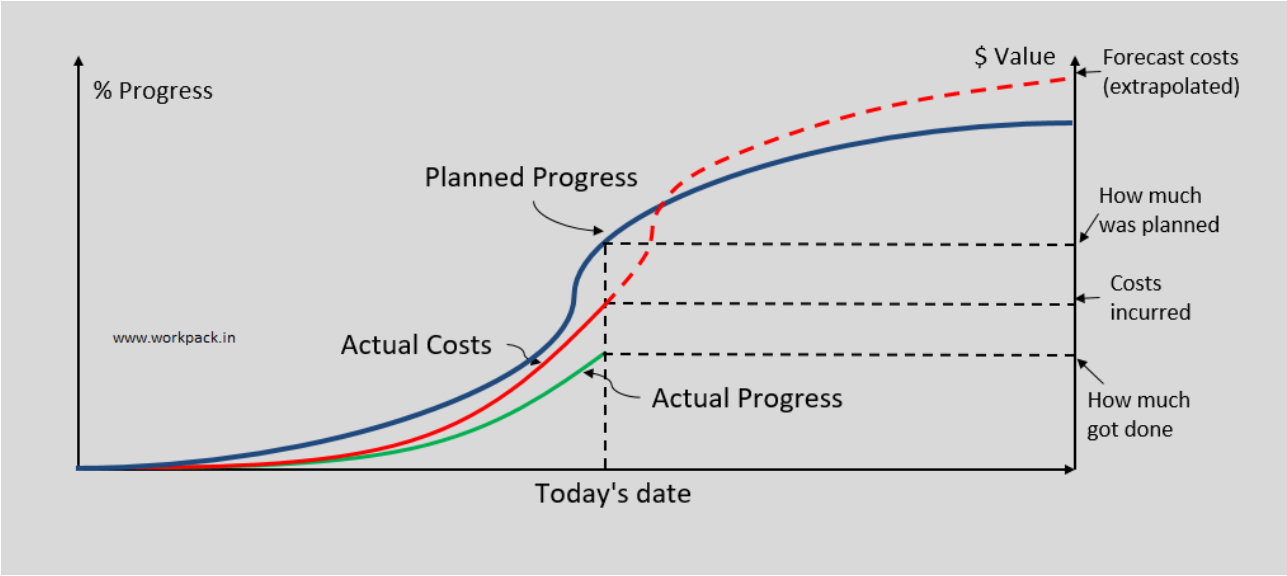

Figura 1.13 - Curve a S [15]

Il Planned Value e l'Actual Cost

Una delle analisi dello stato di avanzamento di progetto è quella basata sui valori Planned Value ed Actual Cost.

Si definiscono i seguenti paramentri:

- Planned Value (PV): rappresenta il costo previsto da programmazione per un determinato periodo, la quantità di prodotto che era stato ipotizzato di realizzare

$$PV = \sum_i Rp_{0,i} * CPU_i$$

Dove:

  - $Rp_{0,i}$ = risorse o lavoro previsti in fase di programmazione per le attività in analisi
  - CPU_i = costo unitario preventivato dell'attività i-esima

- Actual Cost AC: rappresenta il costo effettivo sostenuto in un determinato periodo, la quantità di prodotto effettivamente realizzata

$$AC = \sum_i Ru_{t,i} * CUS_i$$

Dove:

  - $Ru_{t,i}$ = risorse utilizzate o lavoro realizzato
  - CUS_i = costo unitario sostenuto per l'attività i-esima

Lo scostamento tra costo a consuntivo e costo a preventivo è definito come:

$$\Delta costi = AC - PV$$

Effettuando periodicamente una somma dei costi secondo questi due parametri, si ottengono gli andamenti dei costi di periodo, che consentono di costruire le curve a S dei costi a preventivo e a consuntivo ed effettuare l'analisi di confronto (Figura 1.14).

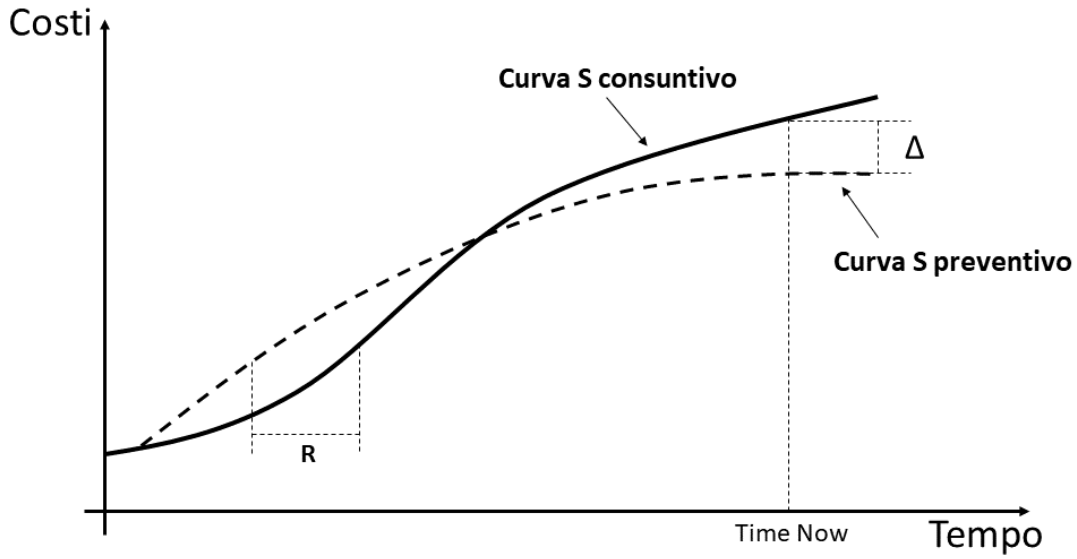

Figura 1.14 – Curva dei Costi (Cantamessa et al., 2007)

Si noti come un valore di $\Delta costi$ negativo non indichi necessariamente un risparmio sui costi unitari di produzione, in quanto potrebbe derivare dall'aver realizzato una quantità di prodotto inferiore rispetto a quella preventivata.

Tale metodo difatti non considera la rlazione tra i costi effettivamente sostenuti e il lavoro effettivamente svolto (Maravas *et al.*, 2012).

Il metodo dell'Earned Value

Il metodo dell'Earned Value è un metodo contabile che consente di rappresentare gli scostamenti con riferimento alle previsioni in termini di:

- Ritardo/Anticipo nei tempi
- Maggiori/Minori costi unitari del prodotto

Si definiscono I seguenti parametri:

- Earned Value (EV): rappresenta il valore a budget del lavoro realmente eseguito, ovvero quanto si sarebbe dovuto spendere se il lavoro effettivamente eseguito avesse avuto i costi definiti a budget

$$EV = Ru_{t,i} * CPU_i$$

Lo scostamento tra costo a consuntivo e costo a preventivo è definito come:

$$\Delta costi = AC - EV$$

Anche in questo caso effettuando periodicamente una somma dei costi secondo questi due parametri, si ottengono gli andamenti dei costi di periodo, che consentono di costruire le curve a S dei costi a preventivo e a consuntivo ed effettuare l'analisi di confronto. Tale confronto sarà effettuato tra termini che risultano tra loro omogenei e si potranno dunque ottenere indicazioni corrette circa il reale andamento economico del progetto.

Il confronto tra Planned Value, Actual Value e Earned Value fornisce informazioni di valore rispetto allo stato di avanzamento del progetto in termini economici, indicando se si stia spendendo di più di quanto rpeventivato per svolgere il lavoro effettivamente realizzato o se si stia eseguendo meno lavoro e dunque si stia procedendo in ritardo (Maravas *et al.*, 2012).

Riportando sullo stesso grafico l'andamento dei tre parametri descritti si può rappresentare il reale andamento del progetto fino al time now (i.e., momento presente) (Figura 1.15).

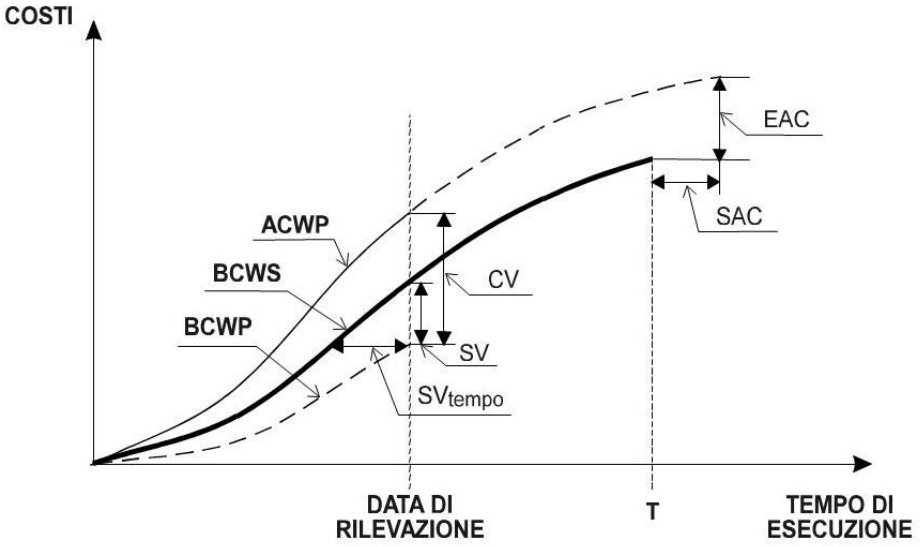

Figura 1.15 - Analisi degli scostamenti [16]

Sull'asse x del grafico viene riportato il tempo, sull'asse y è invece riportata la grandezza cumulata in analisi.

Analizzando il grafico, è possibile definire due ulteriori indicatori di performance:

- Cost Variance CV: indica se si è speso più o meno rispetto al budget preventivato

$$CV = EV - AC$$

- Schedule Variance SV: indica se si è in anticipo o in ritardo rispetto alla pianificazione iniziale

$$SV = EV - PV$$

Performance Analysis

L'analisi delle performance di progetto può essere effettuata attraverso il monitoraggio dei seguenti parametri (Maravas *et al.*, 2012):

- Time Now T: rappresenta la data di rilevamento dei dati
- Budget At Completation BAC: rappresenta il totale dei costi di progetto stimati in sede di preventivo iniziale
- Budget at Completion Time BT: rappresenta la data di termine progetto stimata in sede di preventivo
- Cost Performance Index CPI: rappresenta l'efficienza sui costi.

$$CPI = \frac{EV}{AC}$$

Se:

  - CPI > 1, si sta spendendo meno del previsto
  - CPI = 1, si sta spendendo quanto previsto
  - CPI < 1, si sta spendendo più del previsto
- Schedule Performance Index SPI: rappresenta l'efficienza sul tempo.

$$SPI = \frac{EV}{PV}$$

Se:

  - SPI > 1, si sta facendo più del previsto
  - SPI = 1, si sta facendo quanto previsto
  - SPI < 1, si sta facendo meno del previsto
- Costo residuo a finire ETC: rappresenta il costo che si prevede di sostenere per finire le attività di progetto rimanenti

$$ETC = \frac{BAC - EV}{CPI}$$

- Preventivo Totale Aggiornato EAC: rappresenta la previsione del costo complessivo di realizzazione del progetto ad un dato momento

$$EAC = AC + ETC$$

- Stima totale dei tempi EAC_t: rappresenta la previsione del tempo complessivo di realizzazione del progetto ad un dato momento

$$EAC_t = \frac{BT}{SPI}$$

2. Agile Project Management nel settore IT

Negli ultimi anni si fa un gran parlare di quanto è divenuto più impegnativo per le imprese gestire il proprio business in ambienti fortemente turbolenti, caratterizzati da iper-competizione e dalla necessità di essere più produttivi, veloci, aperti, innovativi, di qualità. Al mutare del contesto è cambiata anche la natura dei progetti da gestire. I progetti, sebbene sempre più brevi, sono generalmente più complessi e i project manager si trovano a operare in ambienti altamente caotici. Il mercato ed i clienti oggi si aspettano prodotti più veloci, più economici e di migliore qualità che richiedono cambiamenti costanti. Il cambiamento come regola mette in discussione i principi teorici del management tradizionale, specialmente quelli legati alla stabilità dei requisiti o dei presupposti iniziali di progetto. Nel tentativo di aiutare i PM e le organizzazioni aziendali nella gestione dei "progetti estremi", un pool di consulenti e studiosi di fama mondiale ha elaborato un insieme di metodologie e tecniche evolute rispetto al project management tradizionale. Si tratta dei "metodi agili" di project management (APM = Agile Project Management) [17].

In questo paragrafo verranno introdotte alcune nozioni fondamentali relative all'APM e ne verranno descritte in dettaglio le principali metodologie.

1.4 Le origini dell'Agile Project Management

I metodi agili (Agile o Lightweight methodologies) si affermano in ambito IT, a metà degli anni '90, in contrapposizione ai cosiddetti metodi Heavyweight o Waterfall. Il metodo Waterfall è un modello classico utilizzato, nel project management, nello sviluppo del ciclo di vita del progetto per creare un sistema con un approccio lineare e sequenziale. Questo modello è diviso in diverse fasi e l'output di una fase viene utilizzato come input per la fase successiva. In poche parole, ogni fase deve essere completata prima dell'inizio della fase successiva e non vi è alcuna sovrapposizione delle fasi [18].

Nel momento in cui un IT manager è sempre più sotto pressione per ottenere risultati – in termini di applicazioni che apportino miglioramenti alla bottom line dell'impresa- i bilanci destinati all'IT vengono significativamente ridotti. Queste modifiche portano a un maggiore interesse per lo sviluppo di software con metodologie agili, che promettono rapida consegna e flessibilità, pur mantenendo la qualità.

A valle del moltiplicarsi di modelli agili, nel 2001 alcuni degli ideatori di tali metodi hanno formulato un Manifesto per lo sviluppo "agile" con i seguenti "precetti":

- Persone ed interazione più che processi e tools.
- Software che funziona più che documentazione esaustiva.
- Collaborazione con il cliente più che negoziazione contrattuale.
- Rispondere al cambiamento più che seguire un piano prestabilito.

In pratica, nonostante si riconosca l'importanza degli elementi a destra nelle frasi riportate, si ritengono più importanti quelli a sinistra.

1.5 Principi chiave APM

I concetti chiave alla base delle metodologie agili di project management sono i seguenti [16]:

- Il valore del cliente innanzitutto: fare in modo che il progetto, il prodotto e i valori del gruppo di lavoro siano allineati per rilasciare prodotti di qualità in modo più veloce ed economico.
- Piccole funzionalità: creare un flusso di funzionalità complete, rilasciat.e al cliente in modo incrementale nel corso della vita del progetto.
- Gruppi di lavoro piccoli e integrati: il lavoro di gruppo è fatto di collaborazione intensa attraverso la localizzazione ravvicinata delle risorse e la comunicazione faccia a faccia; i ruoli sono diversificati all'interno di gruppi integrati, auto-organizzati e auto-disciplinati.
- Miglioramenti piccoli e continui: i gruppi di lavoro riflettono, imparano e si adattano al cambiamento; il lavoro influenza il piano.

1.6 Project management tradizionale vs Agile Project management

Nella Tabella 2.1 è riportato un parallelo sintetico che Sanjiv Augustine ha presentato a Milano nel settembre 2007 per il Project Management Institute (PMI) Northern Italy Chapter al fine di contrapporre le caratteristiche tipiche dell'approccio di project management tradizionale e quello agile.

Tabella 2.1 - Agile PM vs Traditional PM [16]

Agile Project Management	**Traditional Project Management**
Interazione e soddisfazione del cliente al centro	Programmi e prodotti al centro
Risposta adattativa al cambiamento	Controllo correttivo del cambiamento
Elaborazione progressiva, pianificazione dinamica	Pianificazione a monte e rigida
Priorità alle scadenze temporali del cliente	Negoziazione tra i manager del rilascio in base allo scopo
Gestione della committenza attraverso la Feature Beakdown Structure	Gestione delle attività attraverso la Work Breakdown Structure
Collaborazione basata su gruppi di lavoro auto-organizzati ed auto-disciplinati	Controllo dall'alto
Repertorio minimo di procedure flessibili al mutamento di contesto	Metodi prescrittivi, pesanti
Metriche essenziali centrate sul valore	Controlli senza aggiunta di valore

Da questa tabella diventa ancor più chiaro come la risposta agile sia un filosofia guida per poter affrontare al meglio ambienti e contesti turbolenti, con alta instabilità dei requisiti, calati in progetti di durata breve, con gruppi di lavoro di piccole dimensioni e dove la richiesta dei clienti non è quella di vedere dei prototipi, ma funzionalità complete, rilasciate in modo incrementale nel corso della vita del progetto, rispettando le scadenze (progetti time-boxed) senza alcuna deroga.

Quanto definito in Tabella 2.1 è confermato da uno studio scientifico (Eder *et al.*, 2015) secondo il quale è possibile identificare l'utilizzo o meno dell'approccio Agile di project management in un'azienda osservando sei caratteristiche specifiche delle pratiche adottate dall'organizzazione:

- Il modo in cui il progetto viene elaborato;
- Il modo in cui è descritto l'obiettivo del progetto;
- Il livello di dettaglio e standardizzazione con cui è definita ogni attività del progetto;
- L'orizzonte di pianificazione delle attività di progetto;
- La strategia utilizzata per controllare la durata del progetto;
- La strategia utilizzata per garantire il conseguimento dell'obiettivo del progetto.

I valori osservati per un'azienda per ognuna di queste voci (e.g., definizione dell'obiettivo del progetto pari alla soddisfazione del cliente) definiscono l'approccio utilizzato dalla stessa per lo sviluppo del progetto coerentemente con quanto definito nella Tabella 2.1.

1.7 Metodi agili e sviluppo software

La gran parte dei metodi agili tentano di diminuire il pericolo di insuccesso nello sviluppo software mediante la suddivisione temporale del progetto in iterazioni, ovvero in finestre di tempo brevi della durata di poche settimane. Ogni iterazione è un piccolo progetto autonomo e deve avere al suo interno tutto ciò che è indispensabile per rilasciare un piccolo avanzamento nelle funzionalità del software: pianificazione (planning), analisi dei requisiti, analisi, implementazione, test e documentazione.

Seppure ciò che viene rilasciato come risultato di una singola iterazione non contiene tutte le funzionalità per considerarsi completo, esso deve essere rilasciato. Le varie iterazioni, rilasciate una dopo l'altra, consentono di avvicinarsi sempre di più alle esigenze del cliente. Il rilascio progressivo delle iterazioni consente inoltre la rivisitazione in corso d'opera delle priorità di progetto.

I metodi agili preferiscono la comunicazione in diretta, faccia a faccia, rispetto a quella asincrona che passa solitamente attraverso la documentazione scritta di progetto. Il Team agile ingloba non solo chi si occupa di programmazione, ma anche tutti i portatori di conoscenza, tutte le persone che sono in grado di definire come il prodotto dovrà essere fatto. Nel Team, quindi, ci devono stare sia i product manager, che gli analisti di business che i clienti stessi.

Quello che conta per chi gestisce un progetto non è onorare il contratto, ma soddisfare il cliente.

Con le metodologie agili si riduce al minimo la parte più dispendiosa (la progettazione), abbattendo significativamente i costi di sviluppo del software.

Dicendo addio ai vecchi modelli di software engineering, basati sulla raccolta formale delle specifiche e su un processo strutturato di sviluppo software, i metodi agili interpretano le specifiche come un elemento dinamico, che cambia in linea con le necessità degli utenti durante l'avanzamento del progetto.

Come per lo sviluppo software in generale, il lavoro in Team Agile è emergente, in divenire e oggetto di trasformazione continua (Truex et al., 1999). È pertanto importante capire come le persone riescano ad apportare modifiche alle loro pratiche di lavoro e, più in generale, l'impatto che l'utilizzo di metodologie agili ha sulle organizzazioni.

L'uso di tecniche e pratiche Agile sfidano le tradizionali responsabilità di monitoraggio e controllo del progetto da parte dei Project Managers in quanto agli stessi viene sempre più richiesto di essere in grado di effettuare una gestione ibrida del progetto stesso. La tradizionale funzione di PM sembra essere quindi compromessa dal cambiamento di responsabilità e dalla diffusione della conoscenza tra i membri del Team. Tuttavia, la realtà è che i PMs sono chiamati ad essere più dei "gate-keeper" che dei controller – l'utilizzo difatti della gestione visiva dei progetti (i.e., Sprint Backlog) permette un controllo efficiente delle attività di livello inferiore, consentendo al contempo di dedicare più tempo con il cliente per la definizione della strategia e delle priorità. Per i Team Agile l'attenzione è cambiata, in quanto è passata dai problemi tecnici a una necessità di vedere il "quadro generale", al fine di fornire valore al business. Tuttavia, le differenze negli ideali e nei principi tra il Team di sviluppatori e il business portano a diverse prospettive di "valore", rendendo complessa la costruzione di relazioni con i clienti e la comunicazione – è in questo scenario che va considerata l'importanza fondamentale che ricopre e dovrà ricoprire in futuro il PM. Il ruolo del PM nel mondo Agile sta difatti cambiando ed evolvendo verso un diverso tipo di controllo, più ad alto livello (Taylor, 2016).

A interpretare in termini operativi i principi chiave dell'APM esistono numerosi approcci metodologici e tecnici. Nel glossario delle Agile Methodologies ritroviamo (Caracciolo, 2014):

- Extreme Programming (XP)
- SCRUM
- Feature Driven Development (FDD)
- Dynamic Systems Development Method (DSDM)

Tali metodologie verranno descritte in dettaglio di seguito.

Extreme Programming (XP)

L'Extreme Programming (XP) è una metodologia di sviluppo del software che enfatizza la scrittura di codice di qualità e la rapidità di risposta ai cambiamenti di requisiti. Appartiene alla famiglia delle metodologie agili, e come tale prescrive lo sviluppo iterativo e incrementale strutturato in brevi cicli di sviluppo. Altri elementi chiave dell'XP sono il pair programming, l'uso sistematico di unit testing e refactoring, il divieto ai programmatori di sviluppare codice non strettamente necessario, l'enfasi sulla chiarezza e la semplicità del codice, la preferenza per strutture gestionali non gerarchiche, e

l'importanza data alla comunicazione diretta e frequente fra sviluppatori e cliente e fra gli sviluppatori stessi (Beck et al., 2000).

L'XP, nel processo di sviluppo del software, promette di:

- Mantenere la controllabilità del processo pur riducendo il lavoro di supporto
- Convogliare lo sforzo sulla mera produzione dell'applicazione, evitando la produzione di semilavorati diversi da quelli necessari alla realizzazione delle applicazioni
- Fornire i meccanismi affinché gli sviluppatori possano acquisire, durante lo sviluppo, la consapevolezza che ciò che stanno costruendo soddisferà pienamente le esigenze del committente

L'Extreme programming è un processo di sviluppo basato su quattro valori fondanti, individuati da James Donovan Wells:

- Comunicazione – tra cliente e Development Team e all'interno dello stesso Team, come risorsa necessaria affinché tutte le informazioni siano correttamente elaborate al fine di ottenere un sistema più aderente possibile alle esigenze del cliente
- Feedback – frequenti e costanti da parte del cliente, durante tutta la vita del progetto per riuscire a governare i possibili ed inevitabili cambiamenti
- Semplicità – per mantenere il design del sistema e il codice più puliti possibile, in modo da favorire le modifiche e la manutenzione
- Coraggio – nel modificare il sistema, per l'uso di pratiche di verifica del corretto funzionamento del sistema anche dopo numerose modifiche

Egli individua inoltre quattro fasi di progetto, ognuna delle quali con le sue regole interne:

- Pianificazione (user stories, release planning, small releases, project velocity, load factor, iterative development, iteration planning, move people around, daily stand up meeting, fix XP);
- Progettazione (simplicity, system metaphor, CRC cards, spike solution, never add early, refactoring);
- Sviluppo (Customer Always Available, Standards, Unit Test First, Pair Programming, Sequential Integration, Integrate Often, Collective Code Ownership, Optimize Last, No Overtime);
- Collaudo (unit test framework, bug's found, functional test o acceptance tests).

Con l'XP lo sviluppo del progetto viene diviso in sottoparti con rilasci successivi. Il lavoro dei programmatori procede organizzando quattro attività fondamentali che si ripetono per tutto il corso del progetto:

- Coding - scrittura del codice dell'applicazione
- Testing - verifica delle funzionalità

- Listening - osservazione dell'ambiente, inteso come bisogni del committente, opportunità tecnologiche,trend di mercato
- Design - progetto dell'applicazione

La costruzione dell'applicazione procede in maniera iterativa, cogliendo le reazioni dei committenti e riprogettando, in maniera adeguata, le sue funzionalità e i meccanismi adottati per ottenerle.

L'Extreme Programming è un processo di sviluppo adattivo che si implementa attraverso una serie di pratiche di codifica, di design e socio-psicologiche che aiutano i programmatori a rendere il loro lavoro il più efficiente possibile.

Di seguito sono descritte le principali pratiche del XP:

- Pratiche di Codifica:
  - Uso di standard per la codifica (Coding standards): Il codice deve rendere esplicito il meccanismo logico con cui è stato creato, in modo che ciascun sviluppatore sia in grado di capire e modificare ogni linea di codice scritta dagli altri. Il codice, inteso come principale strumento di comunicazione, deve, pertanto, essere scritto in maniera uniforme e omogenea
  - Verifica di ogni funzionalità (Testing): Ogni funzionalità va sottoposta a verifica, in modo che si possa acquisire una ragionevole certezza sulla sua correttezza. Ciò sia a livello di sistema (test di sistema) sia a livello del singolo metodo (test di unità). I test di sistema sono costruiti sulla base delle storie concordate con il committente che dice l'ultima parola sulla convalida del sistema. I test di unità devono poter essere rieseguiti automaticamente con l'uso di opportuni strumenti
  - Ristrutturazione del codice (Refactoring): Un'applicazione necessita di continue riprogettazioni per eliminare le parti divenute superflue e per adattare il sistema alle nuove esigenze. Ogni volta che si presenta la possibilità di eliminare parti superflue o di semplificarne l'organizzazione, l'intera struttura del codice va adattata ai nuovi principi progettuali
- Pratiche di design:
  - Progetti semplici (Simple design): La struttura dell'applicazione deve essere semplice. L'architettura del sistema deve essere comprensibile da tutte le persone coinvolte nel progetto. Non devono esserci parti superflue o duplicazioni. Solo quando nuove circostanze lo richiederanno, verranno progettati nuovi componenti, eventualmente riprogettando anche quelli già esistenti
  - Rilasci frequenti (Short releases): La vita e lo sviluppo dell'applicazione sono scanditi dai rilasci di versioni del prodotto funzionanti. Ogni rilascio realizza uno degli scenari d'uso (storie) che il sistema deve soddisfare e rappresenta il punto conclusivo di un'iterazione di sviluppo e l'inizio di una nuova pianificazione. Per poter tener conto di cambi di prospettiva, errori di valutazione, nuovi requisiti, restrizioni di bilancio, ogni iterazione dovrebbe durare non più di qualche settimana (in genere, da due a quattro)

    - Integrazione continua (Continuous integration): Deve essere costantemente possibile ottenere una versione funzionante dell'applicazione sulla quale operare le verifiche. Una piattaforma pronta per l'integrazione deve essere sempre disponibile per i programmatori
- Pratiche sociali, psicologiche e organizzative:
    - Programmazione a coppie (Pair programming): Due programmatori che lavorano al medesimo terminale scrivono il codice. Le coppie non sono fisse, ma si compongono associando le migliori competenze per la risoluzione di uno specifico problema. Il lavoro in coppia permette, scambiandosi periodicamente i ruoli, di mantenere mediamente più alto il livello d'attenzione
    - Pianificazione delle attività (Planning the game): Lo sviluppo dell'applicazione è accompagnato dalla stesura di un piano di lavoro. Il piano è definito e, continuamente aggiornato, a intervalli brevi e regolari dai responsabili del progetto, secondo le priorità aziendali e le stime dei programmatori,che partecipano, in modo attivo, alla pianificazione. Gli utenti finali dell'applicazione presentano gli obiettivi da raggiungere descrivendo una serie di scenari (storie) che il sistema deve soddisfare. Gli sviluppatori stimano il tempo necessario per la realizzazione di ogni storia. Le storie vengono ordinate da utenti e responsabili secondo la loro priorità di realizzazione, dopo che gli sviluppatori ne hanno stimata la rispettiva difficoltà. Dalla sintesi di queste valutazioni i responsabili del progetto generano la pianificazione delle attività, misurando e controllando l'andamento delle attività rispetto alla pianificazione stessa. Questo processo viene ripetuto dopo ogni rilascio per pianificare il successivo
    - Partecipazione del committente (Onsite customer): Il committente deve essere coinvolto nello sviluppo perché è l'unica fondamentale fonte di convalida del sistema. Partecipa perciò alla stesura dei test di sistema e verifica, periodicamente, che il sistema realizzato corrisponda effettivamente alle proprie esigenze
    - Collettivizzazione del codice (Collective ownership): Il codice dell'applicazione può essere liberamente manipolato da qualsiasi sviluppatore, perché è scritto rispettando regole condivise da tutti. Naturalmente ogni modifica non deve pregiudicare la correttezza dei test
    - System metaphor - descrivere il sistema con una metafora, anche per la descrizione formale. Questa può essere considerata come una storia che ognuno - clienti, programmatori, e manager - può raccontare circa il funzionamento del sistema.
    - Sustainable pace - il concetto è che i programmatori o gli sviluppatori software non dovrebbero lavorare più di 40 ore a settimana.

L'Extreme Programming risulta dunque essere la metodologia ideale per chi vuole:

- Avere un sistema di sviluppo estremamente rapido
- Coinvolgere al massimo il cliente nel processo
- Essere scrupoloso nella prevenzione dei malfunzionamenti
- "Abbracciare il cambiamento" senza aver paura di modificare il codice

- Coinvolgere al massimo gli sviluppatori dando a tutti l'accesso al codice
- Mettere al centro il rispetto delle persone e dei loro dei bisogni personali, sociali e psicologici.

SCRUM

Scrum è un framework agile per la gestione del ciclo di sviluppo del software, iterativo ed incrementale, concepito per gestire progetti e prodotti software o applicazioni di sviluppo, creato e sviluppato da Ken Schwaber e Jeff Sutherland.

Scrum enfatizza tutti gli aspetti di gestione di progetto legati a contesti in cui è difficile pianificare in anticipo. Vengono utilizzati meccanismi propri di un "processo di controllo empirico", in cui cicli di feedback che ne costituiscono le tecniche di management fondamentali risultano in opposizione alla gestione basata sul concetto tradizionale di command-and-control. Il suo approccio alla pianificazione e gestione di progetti è quello di portare l'autorità decisionale al livello di proprietà e certezze operative.

Storia

Nel 1986, Hirotaka Takeuchi e Ikujiro Nonaka descrissero un nuovo approccio allo sviluppo di prodotti commerciali che avrebbe aumentato la velocità e la flessibilità, basato su casi di studio presi dall'industria automobilistica e quella relativa alla realizzazione di fotocopiatrici e stampanti. Essi lo chiamarono approccio olistico o rugby, in quanto l'intero processo viene eseguito da un Team interfunzionale su più fasi che si sovrappongono, dove la squadra "cerca di raggiungere l'obiettivo come unità, passando la palla avanti e indietro".

Il termine Scrum è mutuato dal termine del rugby che indica il pacchetto di mischia ed è evidentemente una metafora del Development Team che deve lavorare insieme in modo che tutti gli attori del progetto spingano nella stessa direzione, agendo come un'unica entità coordinata.

Teoria

Scrum si basa sulla teoria dei controlli empirici di analisi strumentale e funzionale di processo o empirismo. L'empirismo afferma che la conoscenza deriva dall'esperienza e che le decisioni si basano su ciò che si conosce. Scrum utilizza un metodo iterativo ed un approccio incrementale per ottimizzare la prevedibilità ed il controllo del rischio.

Sono tre i pilastri che sostengono ogni implementazione del controllo empirico di processo: la trasparenza, l'ispezione e l'adattamento.

Relativamente alla trasparenza, la metodologia SCRUM afferma che gli aspetti significativi del processo devono essere visibili ai responsabili del lavoro. La trasparenza richiede che quegli aspetti siano definiti da uno standard comune in modo tale che gli osservatori condividano una comune comprensione di ciò che viene visto. Ad esempio:

- Un linguaggio comune di riferimento al processo deve essere condiviso da tutti i partecipanti
- Una definizione comune della parola "Done" deve essere condivisa da chi esegue il lavoro e da chi deve accettarlo

Chi utilizza Scrum deve ispezionare frequentemente gli artefatti prodotti ed i progressi realizzati verso il conseguimento degli obiettivi prestabiliti, individuando in tal modo precocemente eventuali difformità rispetto a quanto si intende realizzare. La frequenza delle ispezioni non deve essere tale da determinare un'interruzione del lavoro in corso. Le ispezioni devono essere eseguite diligentemente e da ispettori qualificati.

Se chi ispeziona verifica che uno o più aspetti del processo di produzione sono al di fuori dei limiti accettabili e che il prodotto finale non potrà essere accettato, deve intervenire sul processo stesso o sul materiale prodotto dalla lavorazione. L'intervento deve essere portato a termine il più rapidamente possibile per ridurre al minimo l'ulteriore scarto rispetto agli obiettivi prestabiliti.

Scrum prescrive quattro occasioni formali per l'ispezione e l'adattamento, altresì dette Cerimonie:

- Sprint Planning Meeting
- Daily Scrum
- Sprint Review
- Sprint Retrospective

Caratteristiche
Scrum è un framework di processo che prevede di dividere il progetto in blocchi rapidi di lavoro (Sprint) alla fine di ciascuno dei quali creare un incremento del software. Esso indica come definire i dettagli del lavoro da fare nell'immediato futuro e prevede vari meeting con caratteristiche precise per creare occasioni di ispezione e controllo del lavoro svolto.

Il framework Scrum è costituito dai Team Scrum e dai ruoli, eventi, artefatti e regole a essi associati. Ogni parte del framework serve a uno specifico scopo ed è essenziale per il successo e l'utilizzo di Scrum. Le regole di Scrum legano insieme gli eventi, i ruoli e gli artefatti governando le relazioni e le interazioni tra essi.

Le persone che ricoprono i ruoli principali nel processo Scrum costituiscono il Team Scrum e sono quelle impegnate nel progetto e che realizzano il prodotto (obiettivo del progetto).

Il Team Scrum è formato dal Product Owner, il Development Team e da uno Scrum Master. I Team Scrum sono auto-organizzati e cross-funzionali: scelgono come meglio compiere il lavoro organizzandosi e coordinandosi al proprio interno e hanno tutte le competenze necessarie per realizzare il lavoro senza dover dipendere da nessuno al di fuori del Team. Il modello di Team in Scrum è progettato per ottimizzare la flessibilità, la creatività e la produttività. I Team Scrum rilasciano i prodotti in modo iterativo e incrementale, massimizzando le opportunità di feedback. I rilasci incrementali di prodotto Done garantiscono che una versione potenzialmente utile del prodotto funzionante sia sempre disponibile.

Il Product Owner rappresenta gli stakeholders ed è la voce del cliente. È responsabile per assicurare che il Team fornisca valore al business. Il Product Owner definisce gli item (requisiti di prodotto) centrati sui bisogni dei clienti (tipicamente User Stories), assegna loro la priorità, e li aggiunge

al Product Backlog. I Team Scrum debbono avere un Product Owner e si raccomanda che questo ruolo non sia combinato con quello dello Scrum Master.

Il Development Team è responsabile della consegna del prodotto, con incrementi di caratteristiche, che sia potenzialmente rilasciabile alla fine di ogni Sprint. Un Development Team è composto da 3-9 persone, con competenze cross-funzionali, che realizzano il lavoro effettivo (analisi, progettazione, sviluppo, test, comunicazione tecnica, documentazione, etc.). Il Development Team in Scrum si auto-organizza, sebbene possa esserci un'interfaccia verso il Project Management Office (PMO).

Lo Scrum Master è responsabile della rimozione degli ostacoli che limitano la capacità del Team di raggiungere l'obiettivo dello Sprint e i deliverable previsti. Sebbene sia un ruolo manageriale, lo Scrum Master non è il Team leader, ma piuttosto colui che facilita una corretta esecuzione del processo. Lo Scrum Master detiene l'autorità relativa all'applicazione delle norme, spesso presiede le riunioni importanti e pone sfide alla squadra per migliorarla. Una parte fondamentale del ruolo di Scrum Master è quello di proteggere il Development Team e tenerlo concentrato sui compiti fungendo da cuscinetto verso qualsiasi influenza di distrazione. Il ruolo viene anche denominato servant-leader per rinforzare queste due prospettive.

Lo Scrum Master differisce da un Project Manager in quanto quest'ultimo può avere responsabilità di gestione del personale che invece non sono in carico allo Scrum Master.

Sprint

Lo sprint è un'unità di base dello sviluppo in Scrum ed è di durata fissa, generalmente da una a quattro settimane. In Figura 1.16 è mostrata una rappresentazione grafica di un Processo Scrum.

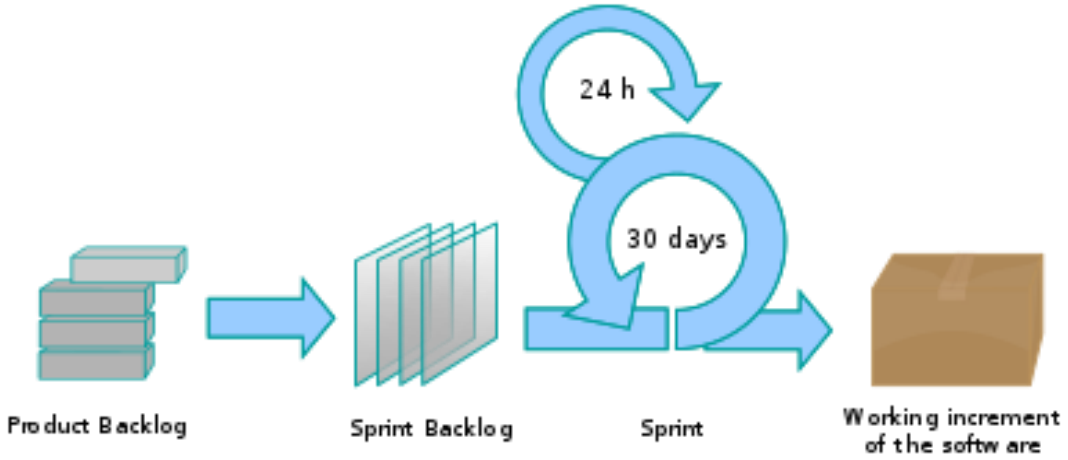

Figura 1.16 - Processo Scrum [19]

Ogni Sprint è preceduto da una riunione di pianificazione in cui vengono identificati gli obiettivi e vengono stimati i tempi. Durante uno sprint non è permesso cambiare gli obiettivi, quindi le modifiche sono sospese fino alla successiva riunione di pianificazione, e potranno essere prese in considerazione nel successivo Sprint.

Al termine di ogni sprint il Development Team consegna una versione potenzialmente completa e funzionante del prodotto, contenente gli avanzamenti decisi nella riunione di pianificazione dello sprint.

Nel corso di ogni sprint, il Team crea porzioni complete di un prodotto. L'insieme delle funzionalità che vengono inserite in un determinato sprint provengono dal Product Backlog, che è una lista ordinata di requisiti. La selezione di quali item del backlog verranno effettivamente inseriti nello sprint (a costituire l'obiettivo dello sprint o "Sprint Goal") viene effettuata durante lo sprint planning meeting. Durante questo meeting, il Product Owner comunica al Team quali item del product backlog vorrebbe che fossero completati (quelli con più alta priorità). Il Team determina quindi quanti di questi pensa di poter completare durante il prossimo sprint e registra questo dato nello Sprint Backlog.

Lo Sprint Backlog è di esclusiva proprietà del Development Team, quindi durante uno sprint la modifica dello sprint backlog non è consentita a nessun altro ad eccezione del Development Team. Lo sprint goal non dovrebbe essere cambiato durante lo sprint. Lo sviluppo è di durata fissa, in modo tale che lo sprint termini alla data prefissata; se i requisiti non sono stati completati per una qualsiasi ragione vengono esclusi dalla review e reinseriti nel product backlog, a discrezione del Product Owner.

Eventi

Gli eventi previsti sono utilizzati in Scrum per creare regolarità e ridurre al minimo la necessità di riunioni non definite da Scrum stesso. Scrum utilizza eventi time-box, in modo che ogni evento abbia una durata massima. Questo assicura che una quantità appropriata di tempo è trascorsa pianificando senza permettere l'introduzione di sprechi nel processo di pianificazione. Oltre allo stesso Sprint, che è un contenitore di tutti gli altri eventi, ogni evento in Scrum è una occasione formale per ispezionare e adattare qualcosa. Questi eventi sono specificamente progettati per consentire trasparenza critica e ispezione. La mancata inclusione di uno qualsiasi dei risultati di questi eventi riduce la trasparenza ed è un'occasione persa per ispezionare e adattarsi.

Lo Sprint contiene e consiste dello Sprint Planning meeting, del Daily Scrum, del lavoro di sviluppo, dello Sprint Review e della Sprint Retrospective. Oltre questi eventi principali, possono aggiungersi altri due meeting: il Backlog Grooming e lo Scrum of Scrums.

All'inizio di ogni ciclo di sprint (ogni 7–30 giorni), viene tenuto uno “Sprint planning meeting”.

Il lavoro da svolgere nello Sprint è pianificato durante lo Sprint Planning Meeting. Questo piano è creato dal lavoro collaborativo dell'intero Team Scrum.

Il meeting Sprint Planning è un incontro della durata di otto ore per uno Sprint di un mese. Per Sprint più brevi, l'evento è proporzionalmente più rapido. Ad esempio, Sprint di due settimane ha un meeting Sprint Planning di quattro ore.

Lo Sprint Planning include i seguenti elementi:

- Selezionare il lavoro da fare
- Preparare lo Sprint Backlog che dettagli il tempo necessario per fare quel lavoro, con tutta la squadra
- Identificare e comunicare la maggior parte del lavoro che è probabile sarà effettuato durante l'attuale sprint

L'intero Scrum Team definisce insieme al Product Owner l'obiettivo dello Sprint e l'insieme di storie su cui impegnarsi.

Il Development Team definisce un piano per lo Sprint, risultante nello Sprint Backlog.

Questo incontro è frequentato dal Product Owner, dallo Scrum Master e dall'intero Development Team. Possono partecipare anche tutti i manager del caso interessati o i rappresentanti della clientela.

Ogni giorno durante lo Sprint, viene tenuta una riunione di comunicazione del Team di progetto. Questo meeting viene chiamato "Daily Scrum", o "Daily Standup" ed ha un insieme di regole specifiche:

- Tutti i membri del Development Team vengono preparati con gli aggiornamenti per la riunione
- L'incontro inizia puntualmente anche se qualche membro del Team è assente
- Il meeting dovrebbe avvenire ogni giorno nello stesso luogo e allo stesso tempo per ridurre la complessità
- La durata del meeting è fissata (timeboxed) al tempo massimo di 15 minuti
- Si partecipa rimanendo in piedi, per non dare modo ai partecipanti di distrarsi ed isolarsi come accade nelle riunioni "tradizionali"
- Durante l'incontro quotidiano, ogni membro del Team risponde a tre domande:
    - Che cosa è stato fatto dopo l'ultima riunione?
    - Che cosa si farà prima della prossima riunione?
    - Quali sono gli impedimenti / ostacoli incontrati?

Gli eventuali impedimenti od ostacoli identificati durante questo meeting vengono documentati dallo ScrumMaster per essere poi lavorati allo scopo di essere risolti al di fuori di questo incontro. Durante il Daily Scrum non dovranno essere affrontate discussioni approfondite.

Lo Scrum Master impone la regola che soltanto i membri del Development Team possono partecipare al Daily Scrum. Questo incontro non è uno status meeting ed è rivolto alle persone che trasformano le voci del Product Backlog in un Incremento.

Il Daily Meeting migliora le comunicazioni, elimina altri incontri, identifica e rimuove gli ostacoli allo sviluppo, evidenzia e promuove il rapido processo decisionale e migliora il livello di conoscenza del progetto da parte del Development Team. Rappresenta un incontro chiave d'ispezione e adattamento.

Per attività di sviluppo maggiori che sono suddivise tra vari Team Scrum, durante l'esecuzione dello Sprint vengono generalmente tenuti altri due incontri di coordinamento: il "Backlog Grooming" e lo "Scrum of Scrums"

Il Team dovrebbe impiegare del tempo durante uno sprint per effettuare il product backlog grooming. Questo è il processo di stima del backlog esistente utilizzando sforzo/punti, raffinando i criteri di accettazione per le storie, e dividendo storie più grandi in storie di minore grandezza e complessità.

Il backlog grooming dovrebbe seguire le seguenti regole:

- Gli incontri dovrebbero essere di durata inferiore al 10% del tempo totale
- Le sessioni di grooming non includono la suddivisione di storie in attività (task)
- Il Team può decidere quanti incontri sono necessari ogni settimana

Il metodo più comune di stima utilizzato è quello del planning poker: un "gioco di carte" per discutere, giustificare e valutare diverse stime realizzative effettuate da tutti i membri del Team per arrivare ad una stima condivisa, ma questa pratica può essere sostituita da altre che fossero ritenute più opportune.

Durante lo Sprint, con cadenza quotidiana o leggermente inferiore, viene tenuto normalmente dopo il Daily Scrum un incontro chiamato Scrum of Scrums.

Questi incontri consentono a gruppi di Team di discutere assieme il loro lavoro, con particolare attenzione sulle aree di sovrapposizione e integrazione.

Anche lo Scrum of Scrums segue regole definite:

- Partecipa una persona designata per ciascun Team
- L'agenda è la stessa dei Daily Scrum, più le seguenti quattro domande:
    - Che cosa ha fatto la tua squadra dal nostro ultimo incontro?
    - Cosa conterà di realizzare il tuo Team prima che ci incontriamo nuovamente?
    - C'è qualcosa che vi rallenta o vi impedisce di ottenere l'obiettivo?
    - Siete in procinto di fare qualcosa che possa essere utilizzato da un altro Team?

Al termine di un ciclo di Sprint, vengono tenute due riunioni: la “Sprint Review” e la “Sprint Retrospective”. L’incontro di Sprint Review è tenuto al termine di uno Sprint per ispezionare l'incremento e adattare, se necessario, il Product Backlog. Durante la riunione di Sprint Review il Development Team e gli stakeholder collaborano su ciò che è stato fatto durante lo Sprint. In conformità a questo e dei cambiamenti al Product Backlog fatti durante lo Sprint, i partecipanti collaborano sulle prossime cose che potrebbero esser fatte. Si tratta di un incontro informale e la presentazione dell'Incremento ha lo scopo di suscitare commenti e promuovere la collaborazione.

Si tratta di un incontro della durata di quattro ore per uno Sprint di un mese. La durata è proporzionalmente inferiore per Sprint più brevi. Ad esempio, se una Sprint dura due settimane, l'incontro di Sprint Review dura due ore.

La Sprint Review include i seguenti elementi:

- Il Product Owner identifica ciò che è Done e ciò che non è Done
- Il Development Team discute su cosa è andato bene durante lo Sprint, quali problemi si sono incontrati e come questi problemi sono stati risolti
- Il Development Team mostra il lavoro che ha fatto e risponde alle domande sull'incremento
- Il Product Owner discute il Product Backlog così com'è. Questi progetta la possibile data di completamento in base alla misura del progresso fino ad oggi
- L'intero gruppo collabora su cosa fare dopo, così la Sprint Review fornisce un prezioso contributo alle successive riunioni di Sprint Planning.

La Sprint Retrospective è l'occasione per il Team Scrum per ispezionare sé stesso e creare un piano di miglioramento da attuare durante il prossimo Sprint. Dopo la Sprint Review e prima del prossimo incontro Sprint Planning, il Team Scrum si riunisce per lo Sprint Retrospective. Si tratta di una riunione di tre ore, per Sprint della durata mensile; in modo proporzionale è allocato meno tempo per Sprint più brevi.

Lo scopo della Sprint Retrospective è di:

- Esaminare come l'ultimo Sprint è andato per quanto riguarda le persone, le relazioni, I processi e gli strumenti
- Identificare e ordinare i maggiori elementi che son andati bene e il potenziale di miglioramento
- Creare un piano per attuare i miglioramenti al modo di lavorare dello Scrum Team.

Lo Scrum Master incoraggia il Team Scrum a migliorare, all'interno del framework di processo Scrum, il proprio processo di sviluppo e le pratiche per rendere più efficace e divertente il prossimo Sprint. Durante ogni Sprint Retrospective, il Team Scrum pianifica i modi per aumentare la qualità del prodotto adattando la definizione di Done secondo i casi.

Entro la fine della Sprint Retrospective, il Team Scrum dovrebbe aver individuato i miglioramenti che saranno implementati nel prossimo Sprint. Attuare tali miglioramenti durante il prossimo Sprint è l'adattamento all'ispezione del Team Scrum stesso. Anche se i miglioramenti possono essere implementati in ogni momento, la Sprint Retrospective fornisce una opportunità formale per focalizzarsi sull'ispezione e l'adattamento.

Artefatti

Gli artefatti di Scrum rappresentano il lavoro o il valore in diversi modi tale da essere utili a fornire trasparenza e opportunità di ispezione e adattamento. Gli artefatti definiti da Scrum sono specificatamente progettati per massimizzare la trasparenza delle informazioni chiavi necessarie ad assicurare ai Team Scrum il successo nella realizzazione di un incremento Done.

Product Backlog

Il product backlog è una lista ordinata dei "requisiti" relativi ad un prodotto. Contiene i Product Backlog Item (PBI) a cui viene assegnata dal Product Owner una priorità in base a considerazioni quali il rischio, il valore di business, le date in cui devono essere realizzati. Le funzionalità aggiunte al backlog sono comunemente scritte utilizzando il formato delle "storie".

Il product backlog rappresenta “cosa” deve essere fatto, organizzato in base all'ordine relativo in cui dovrà essere realizzato. È aperto e modificabile da tutti, ma il Product Owner è il responsabile ultimo della sua gestione e delle priorità da dare alle storie nel backlog per il Development Team.

Il product backlog contiene delle stime approssimative sia del valore di business che dello sforzo necessario a svilupparle; questi ultimi valori sono spesso espressi mediante story point utilizzando una successione Fibonacci arrotondata. Queste stime aiutano il Product Owner a calcolare la timeline e possono influenzare l'ordine dei backlog item. Ad esempio se le funzionalità "aggiungi il controllo ortografico" e "aggiungi un supporto alle tabelle" avessero lo stesso valore di business, quella che richiede il minore sforzo di sviluppo avrà probabilmente una priorità più alta, in quanto il ROI (Return on Investment) sarebbe maggiore.

Il Product Backlog, e il valore di business associato a ciascun item è responsabilità del Product Owner. Invece, lo sforzo stimato per completare ciascun backlog item è determinato dal Development Team. Il Team contribuisce nello stimare gli item e le User-Story, mediante Story-point o mediante diverse misurazioni, quali quelle temporali (per es. ore o giorni).

Sprint Backlog

Lo Sprint backlog è la lista del lavoro che il Development Team deve effettuare nel corso dello sprint successivo (Figura 1.17).

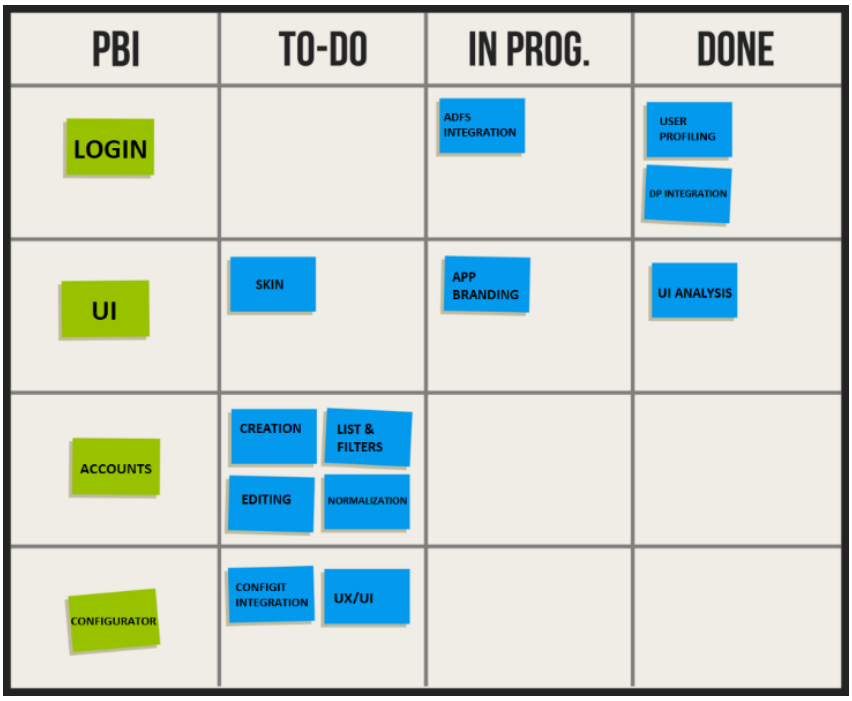

Figura 1.17 - Scrum task board [20]

Questa lista viene generata selezionando una quantità di storie/funzionalità a partire dalla cima del product backlog determinata da quanto il Development Team ritiene possa realizzare durante lo sprint: ovvero avere una quantità di lavoro tale da riempire lo sprint.

Questo viene fatto dal Development Team chiedendosi "Possiamo fare anche questa?" e aggiungendo storie/funzionalità allo sprint backlog. Il Development Team dovrebbe tener conto della Velocity media ottenuta durante gli sprint precedenti (il totale degli story point accumulati per ciascuna delle storie completate durante gli sprint precedenti) nel momento in cui seleziona le storie/funzionalità per il nuovo sprint, e utilizzare tale numero come guida di quanto lavoro potranno realizzare. Un altro parametro che si può anche tenere in considerazione è la capacità, per molti aspetti correlata alla Velocity.

Le storie/funzionalità sono suddivise dal Development Team in attività (task) che taluni considerano buona pratica di durata tra le quattro e le sedici ore di lavoro. Grazie a questo livello di dettaglio il Development Team comprende meglio cosa fare, e potenzialmente, ognuno può prendere in carico un'attività dalla lista. I task non vengono mai assegnati, piuttosto, le attività vengono prese in carico dai membri del Team durante il daily scrum, in base alle priorità predefinite e alle competenze dei membri del Team. Questo promuove l'auto-organizzazione e la responsabilizzazione (buy-in) degli sviluppatori.

Lo sprint backlog è di proprietà del Development Team, e tutte le stime incluse sono effettuate dal Team stesso. Spesso viene utilizzata una task board per visualizzare i cambiamenti di stato dei task nello sprint corrente, come ad esempio ""“to do”, “in progress” e “done”.

Incremento
L'incremento è la somma di tutti gli elementi del Product Backlog completati durante uno Sprint e tutti gli Sprint precedenti.

Al termine dello sprint, l'incremento dovrà essere realizzato in base a quanto concordato dal Development Team nella "Definition of Done". L'incremento deve fornire un prodotto utilizzabile indipendentemente dal fatto che il Product Owner decida effettivamente di rilasciarlo.

Burndown
Un burn down chart è una rappresentazione grafica del lavoro da fare su un progetto nel tempo. Di solito il lavoro rimanente (o backlog) è indicato sull'asse verticale e il tempo sull'asse orizzontale. Il diagramma rappresenta una serie storica del lavoro da fare. Esso è utile per prevedere quando avverrà il completamento del lavoro.

In Figura 1.18 è mostrato un esempio di burn down chart per un'iterazione completata.

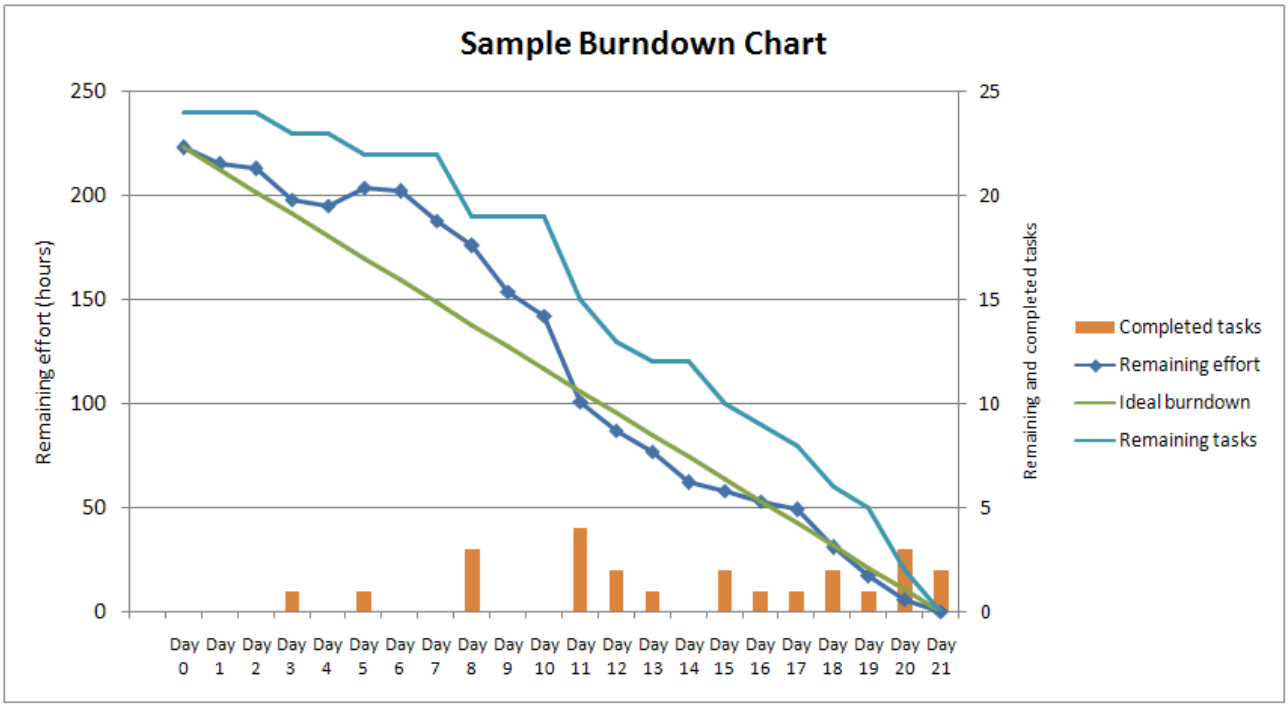

Figura 1.18 - Esempio di Burndown Chart [21]

Feature Driven Development (FDD)

Il feature driven development è una metodologia agile, ideata da Jeff De Luca e Peter Coad, che propone una robusta fase di analisi e progettazione integrata con un modello di sviluppo agile.

Il progetto viene diviso in cinque fasi, dette processi:

1. Sviluppare un modello generale
2. Costruire una lista di funzionalità
3. Pianificare per funzionalità
4. Progettare per funzionalità
5. Sviluppare per funzionalità

Nella fase di sviluppo del modello generale il project manager deve formare il Team di modellazione. Esso deve offrire una panoramica del dominio del problema, preparare i documenti funzionali e, diviso in piccoli gruppi, sviluppare il modello. Il capo-architetto deve rifinire il modello generale ad oggetti insieme al Team di modellazione e scrivere le note al modello insieme ai capi-programmatori. Il risultato di questo processo è la definizione del modello ad oggetti, avendo quindi a disposizione i diagrammi delle classi, i metodi e gli attributi delle classi, la sequenza delle classi (se esiste), le note al modello.

Nella fase di costruzione della lista di funzionalità il project manager deve formare il Team della lista delle funzionalità che deve comprendere i capi-programmatori del Team di modellazione. Il Team della lista delle funzionalità deve definire la lista delle funzionalità in termini di azione-risultato-oggetto. Il risultato di questo processo è la definizione della lista delle funzionalità avendo quindi a disposizione la lista delle aree oggetto, la lista delle attività di business per ogni area oggetto e la lista delle funzionalità che soddisfino tutti i punti di ogni lista delle attività.

Nella fase di pianificazione per funzionalità il project manager deve formare il Team di progettazione che comprende capi-programmatori e manager dello sviluppo; il Team di progettazione deve definire la sequenza di sviluppo, assegnare le attività di business ai capi-programmatori ed assegnare le classi agli sviluppatori. Il risultato di questo processo è la definizione del piano di sviluppo comprendente le attività di business con le date di completamento ed i capi-programmatori responsabili assegnati, la date di completamento delle aree oggetto (derivate da quelle delle attività di business) e la lista delle classi con relativi sviluppatori.

Nella fase di progettazione per funzionalità ogni capo-programmatore forma il Team delle funzionalità ed ogni esperto del problema definisce la strada per affrontare il problema e risolverlo. Il Team delle funzionalità studia i documenti dei requisiti delle proprie funzionalità, mentre il Team di progettazione sviluppa i diagrammi di sequenza. Il capo-programmatore raffina il modello ad oggetti per definire se scrivere o modificare classi, metodi, attributi e il Team delle funzionalità scrive le classi e gli header dei metodi. Il risultato di questo processo è la definizione di un pacchetto di progettazione completo che comprenda un documento esplicativo dell'intero progetto con le specifiche referenziate (se esistono riferimenti), i diagrammi di sequenza, le alternative di progetto (se esistono), il modello ad oggetti completo di classi, metodi e attributi, gli header di classi e metodi, una todo list con un calendario delle scadenze per ogni attività ed ogni membro del Team.

Nella fase finale del progetto il Team delle funzionalità implementa classi e metodi, ispeziona il codice ed effettua i test unitari. Il capo-programmatore decide (dopo i test unitari) insieme al Team delle funzionalità quali classi siano promuovibili come utili alla costruzione del progetto in riguardo alle funzionalità richieste. Il risultato di questo processo finale l'ottenimento di classi e metodi che siano stati ispezionati e testati con successo, infine promossi all'integrazione nel progetto (ovviamente a copertura di tutte le funzionalità previste).

L'iterazione si può ridurre al semplice ciclo composto dai seguenti passi: definire una lista di funzionalità e sviluppare per funzionalità. Il processo critico è il primo, ossia sviluppare un modello generale, ed è questo che deve dare in uscita la documentazione UML. Questi diagrammi devono essere impostati in maniera tale da ottenere una frammentazione (una granularità) dall'alto verso il basso, in modo da avere funzionalità da sviluppare che possano essere prodotte in brevi iterazioni. Queste funzionalità devono poi essere riunite in un singolo package alla fine di ogni iterazione.

Ogni iterazione è composta da sei fasi (Palmer et al., 2002):

1. Kickoff Meeting del Package (viene chiarito ogni dettaglio delle funzionalità incluse)
2. Design (vengono definiti classi/metodi/documentazioni richiesti)
3. Design review (tutti gli attori dello sviluppo revisionano il progetto proposto)
4. Development (si implementa il codice previsto con i relativi unit test)
5. Code review meeting (la revisione del codice viene effettuata da tutti i programmatori)
6. Release meeting (le funzionalità implementate vengono "promosse" al processo di integrazione)

Il processo di sviluppo

La collezione dei requisiti dell'utente e delle specifiche avviene coinvolgendo il cliente anche durante lo sviluppo del progetto e non solo in una fase iniziale. Inizialmente si ottengono una lista di funzionalità richieste, che diventerà la base della timeline del progetto, ed uno o più diagrammi UML che definiscano il dominio del problema. Partendo da questa base, cliente e sviluppatori lavoreranno insieme ai requisiti in brevi iterazioni che coprano frammenti di business.

Feature Driven Development non richiede esplicitamente la stesura di documentazione, ma obbliga all'utilizzo dei diagrammi UML. Questo per avere una base decisionale che sia stabile durante tutto il processo di sviluppo, solo in seconda battuta sarà utile per scrivere una documentazione formale, se richiesta.

Nel corso del progetto ci sono molti documenti che devono essere disponibili per i diversi attori, quindi la miglior soluzione è organizzare un sito web interno che contenga tutte le informazioni sul progetto: la lista di sviluppatori ed esperti del problema, il modello UML con i commenti, i forum di discussione, le convenzioni di scrittura del codice, la lista degli strumenti e delle librerie usate, i report dei test unitari, lo status del progetto, la timeline comprensiva della pianificazione futura, etc.

Durante lo sviluppo, organizzato in iterazioni brevi, si forma una struttura gerarchica con figure a metà strada fra il project manager e gli sviluppatori: i capi-programmatori. Questi sono a capo di ogni singola iterazione, che quindi possono essere numerose e procedere parallelamente, scegliendo anche il Team (composto da 3-5 persone) che se ne occuperà. L'esperienza dei capi-programmatori e la frammentazione del lavoro in iterazioni, sono i meccanismi di controllo e regolazione di Feature Driven Development. All'inizio di ogni iterazione si organizzano delle riunioni di progettazione che servono a far confrontare i membri del Team e ad ottenere la documentazione del codice.

La stesura del codice prevede l'utilizzo rigoroso di uno standard comune di scrittura e il ricorso ad i test unitari, che possono essere organizzati a discrezione dei capi-programmatori.

Date le numerose riunioni effettuate prima di cominciare a scrivere il codice, questa attività diventa qualcosa di meccanico, infatti Feature Driven Development scoraggia l'uso di pratiche tipo il Refactoring mentre incoraggia la condivisione del codice prodotto (in maniera particolare della documentazione relativa) fra i diversi programmatori.

Per la revisione del codice si va oltre il Pair programming visto che la condivisione all'interno del Team dell'iterazione permette una verifica molto più ampia. In ogni caso è proprio per permettere la miglior revisione possibile del codice che i Development Team devono essere poco numerosi e che le iterazioni devono essere brevi, fra 1 e 3 settimane.

Il rilascio delle versioni è previsto per la fine di ogni iterazione, raramente di più iterazioni, ma ciò permette di coinvolgere molto di meno il cliente rispetto a quanto facciano le altre metodologie

leggere. E permette anche di non consegnare alcune versioni intermedie quando le condizioni al contorno non lo rendano possibile, ad esempio in caso di software medici embedded.

Tenere una traccia dello status del progetto è un compito reso semplice da Feature Driven Development in quanto si ha a disposizione sin dall'inizio la lista delle funzionalità da implementare ed ogni iterazione ha dei pesi ben definiti per ogni passo:

- Iteration planning meeting - domain walkthrough (1%),
- design phase (40%),
- design review meeting - inspection (3%),
- coding phase (45%),
- code review meeting - inspection (10%),
- promote to build moment (1%).

Partendo da questi dati, Jeff De Luca afferma che cambiamenti fino al 10% dei requisiti durante il corso del progetto non incidono in maniera tale da inficiare la deadline prevista (Palmer et al., 2002).

Dynamic Systems Development Method

Il Dynamic Systems Development Method è una metodologia agile distribuita gratuitamente dal consorzio DSDM ai propri membri ed è fornito in pacchetto insieme ad un framework.

DSDM è una metodologia che si integra facilmente con le altre metodologie agili (esistono risorse già pronte per farlo con Extreme Programming, Prince2, RUP, etc.).

Principi

Il metodo DSDM si basa su quattro principi:

- Lo sviluppo è un lavoro di gruppo
- L'alta qualità si ottiene con velocità e robustezza
- Lo sviluppo può essere incrementale
- Bisogna spendere il tempo dedicato allo sviluppo concentrandosi prima sulle funzionalità che rendono di più in termini di business

Nell'ambito disegnato da questi principi si inseriscono le regole di base:

- Coinvolgimento attivo degli utenti
- Potere decisionale al Team
- Distribuzioni frequenti del prodotto
- Sviluppo iterativo ed incrementale
- Tutti i cambiamenti effettuati durante lo sviluppo devono essere reversibili
- I requisiti e le specifiche sono definite solo ad alto livello
- La fase di test è integrata nel ciclo di vita del prodotto
- Collaborazione e cooperazione fra gli attori coinvolti nel progetto sono un obbligo

L'assunto da cui si parte per ottenere un buon prodotto in tempi migliori è che, anche se le prime versioni non saranno perfette, in generale è meglio distribuirle subito perché l'80% del prodotto richiesto può essere sviluppato nel 20% del tempo necessario a sviluppare il prodotto intero. Poi, nel resto del tempo, si potrà terminare il lavoro e correggere gli errori che gli utenti segnaleranno.

Ruoli

Differenza sostanziale rispetto alle altre metodologie agile è la definizione dei ruoli degli attori del progetto, ognuno dei quali può essere coperto da più persone (o viceversa, una persona può ricoprire più ruoli):

- Executive Sponsor (cioè il Project Champion)
- Visionary (responsabile dell'attivazione del progetto)
- Ambassador User (proveniente dall'area di business coperta)
- Advisor User (automatizza la conoscenza del lavoro giorno per giorno)
- Project Manager (può provenire dalla comunità degli utenti o dell'IT)
- Technical Co-ordinator (presiede ogni Development Team)
- Team Leader (responsabile del lavoro di un singolo Team)
- Developer (modella ed interpreta le richieste degli utenti, sviluppando prototipi e versioni da distribuire)
- Tester (effettua solo i test che gli utenti non possono svolgere)
- Scribe (presiede tutte le riunioni per mettere per iscritto i punti importanti)
- Facilitator (gestisce le riunioni di lavoro, è indipendente dal Team di progetto)
- Specialist Roles (tutte le figure specialistiche che possano servire, Business Architect, Quality Manager, ecc.)

Il Facilitator gestisce perciò le riunioni di lavoro, che sono il punto importante del metodo e sono dette Facilitaded Workshops (Stapleton, 1997).

2. Coolshop srl

Di seguito sarà introdotta l'azienda titolare del progetto analizzato nell'elaborato di tesi.

2.1 Origine e breve storia dell'azienda

Coolshop srl è una società specializzata nel fornire soluzioni ICT, fondata nel 2005 a Torino dall'Ingegnere Mattia Pontacolone. Appassionata di tecnologia e novità, con attenzione a creatività e design, da anni si occupa dello sviluppo di soluzioni informatiche per aziende, con particolare attenzione all'ottimizzazione dell'esperienza utente, oltre che al codice stesso. La filosofia aziendale di Coolshop è che i business di successo siano basati su una grande User Experience (UX): per tale motivo l'azienda collabora quotidianamente con i clienti fornendo una solida conoscenza della tecnologia unitamente ad visione consumer-oriented.

Coolshop srl garantisce ai propri clienti la massima riservatezza dei dati e delle informazioni condivise. Assicura inoltre un' integrità professionale assoluta agli stessi, offrendo sempre un processo di supporto ICT che consenta il trasferimento delle competenze necessarie per sfruttare i nuovi strumenti sviluppati e forniti. Con una visione globale, nel 2015 Coolshop fonda una nuova filiale in Olanda al fine di seguire i clienti chiave in Europa e in America. Negli anni successivi vengono inoltre fondate due nuove sedi a Dubai e a Chicago.

2.2 Organizzazione Aziendale

In Figura 2.1 viene riportato l'organigramma aziendale:

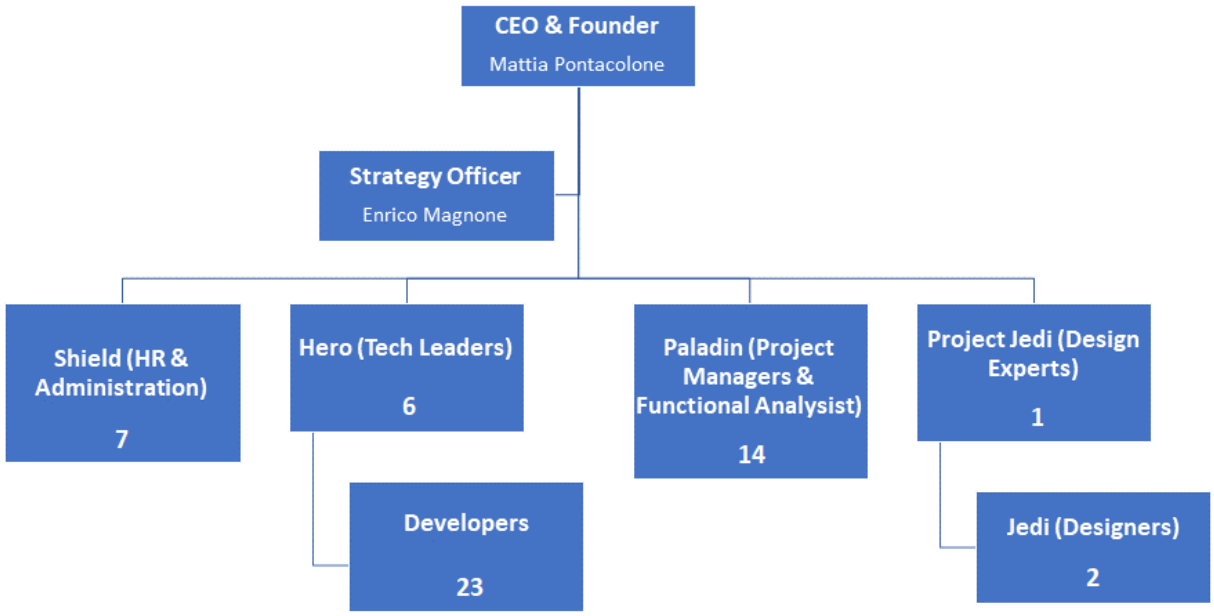

Figura 2.1– Organigramma aziendale

2.3 Prodotti e/o servizi offerti dall'azienda

Coolshop fornisce ai propri clienti soluzioni ICT customizzate e svariati servizi legati al mondo del digitale, per i quali vanta grande esperienza, conoscenza e il know how acquisito negli anni.

2.3.1 Web & Mobile Development

Coolshop presenta come proprio core-business lo sviluppo di applicazioni Web e Mobile per differenti clienti.

Lavora principalmente su progetti di Sales ed E-commerce, sfruttando i migliori strumenti open source nel mercato. Volge verso le applicazioni Mobile nello specifico particolare passione e predilezione, essendo convinta che quest'ultime permettano di avere il proprio business a portata di mano e, perciò, siano il futuro nel mondo dell'ICT. Le applicazioni fornite ai propri clienti sono sviluppate con una forte attenzione al front-end design, sfruttando la grande tradizione presente a Torino: viva è la collaborazione con istituti di design e professionisti del settore.

Coolshop inoltre fornisce analisi di scenari di utilizzo, sperimentazione con A/B testing e studi di usabilità, con l'obiettivo di individuare il miglior design possibile, unitamente all'efficacia funzionale delle proprie applicazioni. Coolshop infine offre ai propri clienti applicazioni che consentano di gestire i dati aziendali in una maniera sicura e flessibile, grazie all'esperienza acquisita nel campo delle integrazioni back-end con tutti i principali sistemi quali ERP (Enterprise Resource Planning), CRM (Customer Relationship Management), CPQ (Configure, Price, Quote), AI (Artificial Intelligence) e BI (Business Intelligence).

2.3.2 Marketing Experience

Coolshop fornisce ai propri clienti supporto nell'ambito del Marketing digitale, promuovendo un approccio guidato dai dati di comportamento del cliente finale raccolti grazie alle soluzioni sviluppate. Nel fornire tale servizio, l'azienda integra l'esperienza acquisita nell'ambito dei servizi ICT online ed offline, con l'obiettivo di massimizzare gli investimenti grazie a messaggi di marketing che siano su misura.

Fornisce inoltre un design interattivo studiato per acquisire maggiori informazioni sui clienti e sui prospect, con l'obiettivo di seguire e guidare le preferenze del pubblico delle aziende per le quali lavora.

2.3.3 Applicazioni Enterprise

Consapevole che un approccio diretto con le moderne tecnologie sia in grado di risolvere molti dei problemi di sviluppo IT aziendali che devono sostenere la possibilità di integrazioni SAP, Coolshop fornisce lo sviluppo di applicazioni custom per i clienti enterprise. Integrando competenze di marketing e di e-commerce nella progettazione delle applicazioni, garantisce allo stesso tempo i più attenti standard enterprise: unit test, miglioramento continuo, implementazioni pianificate, monitoraggio delle anomalie e integrazione SSO (Single Sign On).

Valore aggiunto a tutto ciò è il vantaggio di poter fornire la portabilità su mobile, l'accesso offline e la disponibilità via extranet dei dati con effort limitato.

2.3.4 Integrazione SAP

Coolshop fornisce ai propri clienti una solida esperienza nello sviluppo di applicazioni mobile che fungono da interfacce SAP per garantire un'interazione user-friendly con lo standard dei sistemi di gestione aziendale. L'integrazione diretta di SAP con le tecnologie web moderne fornita da Coolshoop offre ai propri clienti diversi vantaggi tra i quali la validazione dei dati in tempo reale, la semplificazione delle integrazioni esterne e una interfaccia utente più chiara ed intuitiva. Dal punto di vista dello sviluppo l'utilizzo di SAP come engine, libera dalla necessità di sviluppare costose viste personalizzate e permette facilmente l'implementazione di uno strumento esterno che viene poi collegato al sistema aziendale interno.

2.3.5 Consulenza

Vivendo tutti in un ecosistema digitale, il mercato sta diventando sempre più consapevole, intelligente ed esigente. Alla luce di ciò, Coolshop supporta i propri clienti attraverso attività di Social Media Management, grazie allo sviluppo di nuovi concetti di comunicazione, ottenibili con l'adozione di nuove tecnologie per far fronte ad uno scenario in rapida evoluzione.

Svolge inoltre attività di Project Management per i propri clienti, attraverso l'inserimento di risorse dedicate, in azienda o da remoto, con Team qualificati composti a seconda dello specifico scenario di business.

2.4 Principali Clienti

Coolshop annovera tra i propri clienti importanti aziende, italiane e non, operanti in diversi settori commerciali (i.e., Automotive, Petroleum Industry, Manufacturing, etc.). Tra questi, i principali per reputazione e durata del rapporto di collaborazione, sono i seguenti:

- FCA
- Jeep
- CNHi
- Iveco
- Cameron
- FG Wilson
- Abarth
- Enoc Link
- CAT
- Mopar
- CMCO
- Beema

3. CNHi Sales App

Nel presente capitolo verrà descritto il progetto oggetto del lavoro di tesi, definendone obiettivi, caratteristiche e specifiche, ed analizzandone ogni singola fase. Verrà inoltre fornita una descrizione dettagliata dell'approccio Agile utilizzato per ogni ciascuna fase in una sezione di dettaglio.

3.1 Il progetto e i suoi obiettivi

CNH Industrial Group ha pianificato di gestire l'evoluzione dell'attuale soluzione utilizzata per il Customer Relationship Management (CRM) all'ultima disponibile, Microsoft CRM 365, unitamente all'integrazione di una soluzione di Configure, Price and Quotation (CPQ) che sia condivisa dai tutti i brand appartenenti al gruppo, con l'obiettivo di raggiungere un modello di convergenza tra essi.

Il progetto ha dunque l'obiettivo di fornire una soluzione unica che garantisca l'accesso alle funzionalità CPQ e CRM, che funga dunque per CNHi come supporto al processo di vendita dei veicoli nuovi, appartenenti allo stock, ed usati, e fornisca un'interfaccia per la gestione delle relative entità e funzionalità di tipo CRM ereditate dal nuovo CRM Dynamics 365. Tale soluzione è per l'appunto l'applicativo denominato CNHi Sales App, pensato come un'interfaccia user-friendly e disponibile su molteplici piattaforme per la gestione delle funzionalità CRM con l'integrazione delle funzionalità CPQ.

La soluzione verrà fornita sia agli utenti interni, i dipendenti CNHi, che agli utenti esterni, i Dealers.

Il sistema fornito deve garantire:

- Ottima End User Experience, che si traduce in interazioni semplificate per l'accesso alle informazioni di valore, unitamente a un alto livello di prestazioni per tutti gli attori coinvolti (i.e., Dealer Manager, Salespersons, Central Users, Market Users, etc.)
- Miglioramento dei processi grazie ad una solida integrazione con sistemi esterni all'applicazione
- Capacità di adattamento a Business Changes futuri
- Migliore capacità di monitoraggio delle attività di tipo CRM

Al fine di garantire a CNH Industrial Group il raggiungimento dei benefici precedentemente elencati, l'applicazione, Sales App, è disegnata al fine di garantire i seguenti obiettivi chiave:

- Aumentare il livello di utilizzo dell'applicazione da parte dei Dealers
- Ridurre l'effort richiesto per il supporto back office
- Fornire una piattaforma software che sia sicura e compliant con le normative di privacy e sicurezza
- Ridurrei costi di mantenimento del software
- Garantire una standardizzazione dei processi e delle funzionalità
- Ridurre l'overlap delle funzionalità
- Garantire un'interfaccia user-friendly al fine di ridurre la necessità di supporto agli utilizzatori finali

3.1.1 Attività di progetto

In fase di definizione commerciale del progetto è stato concordato col committente il perimetro di azione di Coolshop, definendo le attività progettuali delle quali l'azienda stessa si sarebbe occupata in relativamente al progetto oggetto dell'elaborato. Le attività concordate e fornite da Coolshop sono quelle standard di un progetto IT e sono le seguenti:

- Discovery, ovvero scoperta e definizione dei requisiti:
    - Raccolta e consolidamento dei requisiti provenienti dal Team Business di CNHi (il committente)
    - Definizione della User Interface (UI) e della User Experience (UX)
    - Produzione del documento di dettaglio relativo al design tecnico della soluzione (i.e., Business Blueprint, BBP)
    - Esecuzione delle analisi funzionali
- Implementation, ovvero sviluppo dell'applicativo:
    - Sales App User Interface (UI) e relative logiche
    - Sales App Business Logic (back-end), ovvero tutte quelle logiche applicative che risiedono nel backend dell'applicazione stessa e non sono visibili direttamente all'utente. Esse hanno a che fare con la gestione e il mantenimento dei flussi di dati e dell'infrastruttura dell'applicativo (i.e., server e ambienti)
    - Setup del prodotto, ovvero la definizione delle logiche necessarie affinché lo stesso sia reso accessibile agli utenti e funzionante (es. profilazioni utente, definizione settings applicativi, setup degli ambienti)
- Test, ovvero attività relative al testing dell'applicativo e supporto nelle fasi di validazione del prodotto da parte del committente (par. 1.2.1 – IT Project Life Cycle):
    - Unit and Integration test
    - Stress and performance test
    - Preparazionetraining material (manuale utente)
    - User Acceptance test (UAT)
- Production release
    - Attività di Cutover
    - Supporto Post Go-Live

Sulla base delle attività di progetto descritte precedentemente si è definita la WBS di progetto come segue:

- Workshops (WS) and Business Blue Print (BBP): fase in cui vengono individuati gli obiettivi di business dell'azienda e si determinano i requisiti necessari per il loro raggiungimento. Tutto ciò avviene attraverso attività di WS, ovvero incontri della durata di un'intera giornata in cui avviene un dibattito tra il fornitore ed il committente riguardante la nuova tecnologia da installare. Tale livello è stato scomposto nel seguente sottolivello:
    - Completamento dei WS;
    - Produzione della BBP da parte di Coolshop;
    - Approvazione della BBP da parte del committente;
- Design & Approval: fase nella quale il fornitore fornisce al cliente i Functional Designs (FDD) delle funzionalità che verranno deliverate, ovvero le User Stories descritte in Tabella 3.3. Il cliente deve approvare li stessi al fine di garantire l'avvio delle attività di sviluppo vere e proprie. Tale livello è stato scomposto nel seguente sottolivello:
    - Definizione delle User Stories e analisi tecnica;
    - Creazione dei FDD;
    - Approvazione dei FDD;
- Build & Unit Test: fase vera e propria di sviluppo delle funzionalità (User Stories descritte in Tabella 3.3). Ad essa si affianca lo Unit Testing, ovvero l'attività di testing di singole unità software. Per unità si intende normalmente il minimo componente di un programma dotato di funzionamento autonomo. Tale livello è stato scomposto nel seguente sottolivello:
    - Completamento sviluppi delle User Stories secondo priorità;
    - Unit Testing degli sviluppi completati (i.e., Definizione di Fatto o Done)
- Integration Test: fase nella quale si procede al testing dell'applicazione, cercando di ottenere un risultato in risposta a una certa azione. Questo genere di test verifica non solo il corretto comportamento di ogni singolo oggetto, ma anche le relazioni con gli altri componenti dell'applicazione. Tale livello è stato scomposto nel seguente sottolivello:
    - Superamento Integration Test.
- UAT Support: fase nella quale il fornitore supporta il cliente durante gli UAT. Tale livello è stato scomposto nel seguente sottolivello:
    - Rilascio funzionalità per UAT;
    - Approvazione UAT.
- Training Documentation: fase nella quale il fornitore provvede alla stesura dei manuali necessari agli utenti per gli UAT. Tale livello è stato scomposto nel seguente sottolivello:
    - Creazione della documentazione di Training;
    - Approvazione della documentazione di Training.
- NRT: fase nella quale si verifica che verificare che le vecchie funzionalità disponibili non siano state danneggiate dal nuovo codice introdotto. Tale livello è stato scomposto nel seguente sottolivello:
    - Superamento NRT.
- Cut Over & Go Live: fase nella quale si portano a termine una serie di step necessari per il Go Live dell'applicazione, ovvero il rilascio al cliente;
- PGL Support: fase di supporto al cliente dopo il Go Live dell'applicazione.

Il primo livello della WBS è stato ottenuto applicando una logica di scomposizione per processi di lavoro.

Il secondo livello invece ha visto l'applicazione della logica per obiettivi.

3.1.2 Piattaforme

L'applicazione Sales App è disegnata ed implementata per essere utilizzata su piattaforme differenti (Figura 3.1):

- Web application
- Desktop application
- Tablet & SmartphSales Application
- B2C website

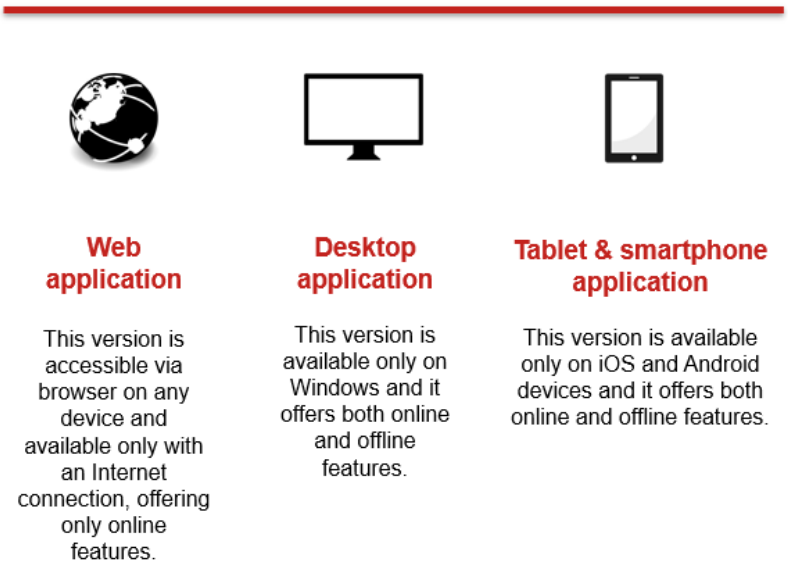

Figura 3.1 - Piattaforme Sales App(Coolshop, 2019, "CNHi Sales App Business Blue Print")

Le versioni di Sales App per utilizzo su Desktop e dispositivi Mobile è stata disegnata per essere utilizzabile:

- **Online:** quando una connessione ad internet è disponibile per essere utilizzata dall'applicazione. In questo scenario i dati degli utenti sono sincronizzati in maniera automatica e dunque mantenuti sempre aggiornati
- **Offline:** quando una connessione ad Internet non è disponibile per essere utilizzata dall'applicazione. Alcune funzionalità saranno non raggiungibili, l'utente sarà però in grado di utilizzare i dati sincronizzati precedentemente in presenza di connessione.

Inoltre, una **Web application** è disponibile ed accessibile tramite browser: in questo scenario una connessione ad Internet è obbligatoria.

3.1.3 Data Management

In Figura 3.2 è mostrata la relazione che intercorre tra Sales App e il nuovo CRM che verrà utilizzato da CNHi con riferimento alla gestione delle principali entità coinvolte nei processi CPQ e CRM che sono garantiti dall'applicazione.

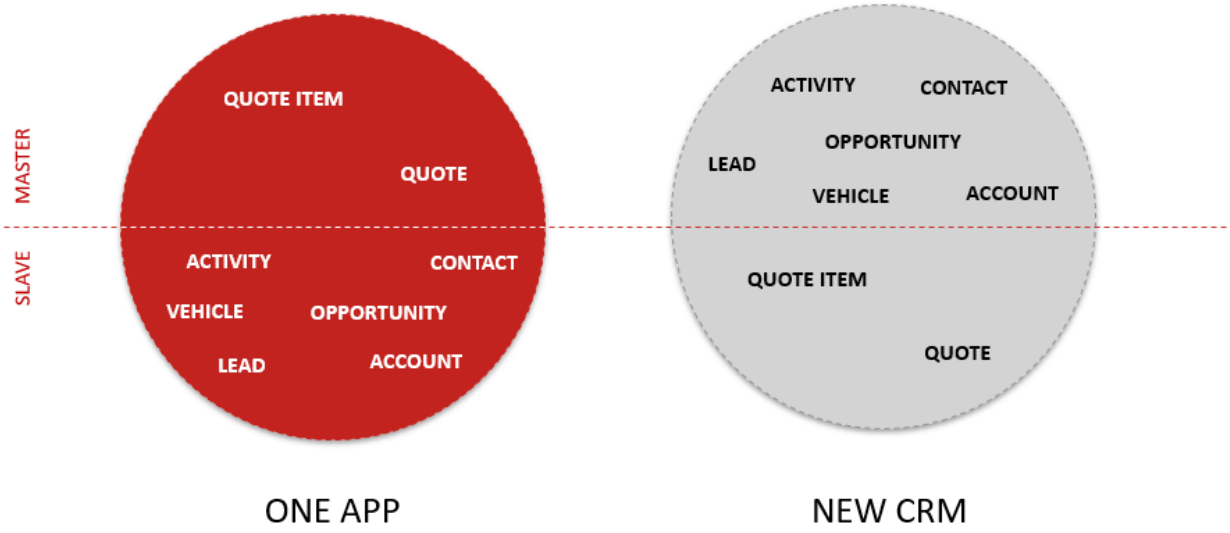

Figura 3.2 - Data Management (Coolshop, 2019, "CNHi Sales App Business Blue Print")

In dettaglio, i sistemi sono stati identificati come:

- Master, se il sistema in questione detiene le informazioni più aggiornate e veritiere su di un'entità essendo la stessa un'entità "naturale" del sistema (l'entità ha generalmente origine in quel sistema e quel sistema è considerato come fonte di informazioni più autorevole per l'entità stessa)

Slave, se il sistema in questione non detiene le informazioni più aggiornate su di un'entità ma importa l'entità stessa da un sistema terzo, che è il Master di quella entità. E' possibile che un'entità sia creabile anche direttamente su di un sistema che è Slave di un altro sistema per quella specifica entità, ma l'autorità relativamente all'entità stessa non appartiene al sistema Slave nonIl Sistema CRM Microsoft Dynamics è data owner, dunque Master , delle informazioni relative alle seguenti entità:

- Lead
- Account
- Contact

- Opportunity
- Activity
- Vehicles

Sales App, invece, è Master di:

- Offer
- Offer Item

Come sottolineato al paragrafo 3.1.3…., i database dei due sistemi risultano essere Slave dell'altro rispettivamente per tutte le entità per le quali non risultano master.

3.1.4 Landscape

Nel presente paragrafo sarà descritto il landscape dell'applicazione fornita e le principali funzionalità che essa garantisce. Con Landscape si intende l'insieme di tutti i sistemi coinvolti nel progetto in analisi. In Figura 3.3 è mostrato il Landscape relativo al progetto CNHi Sales App.

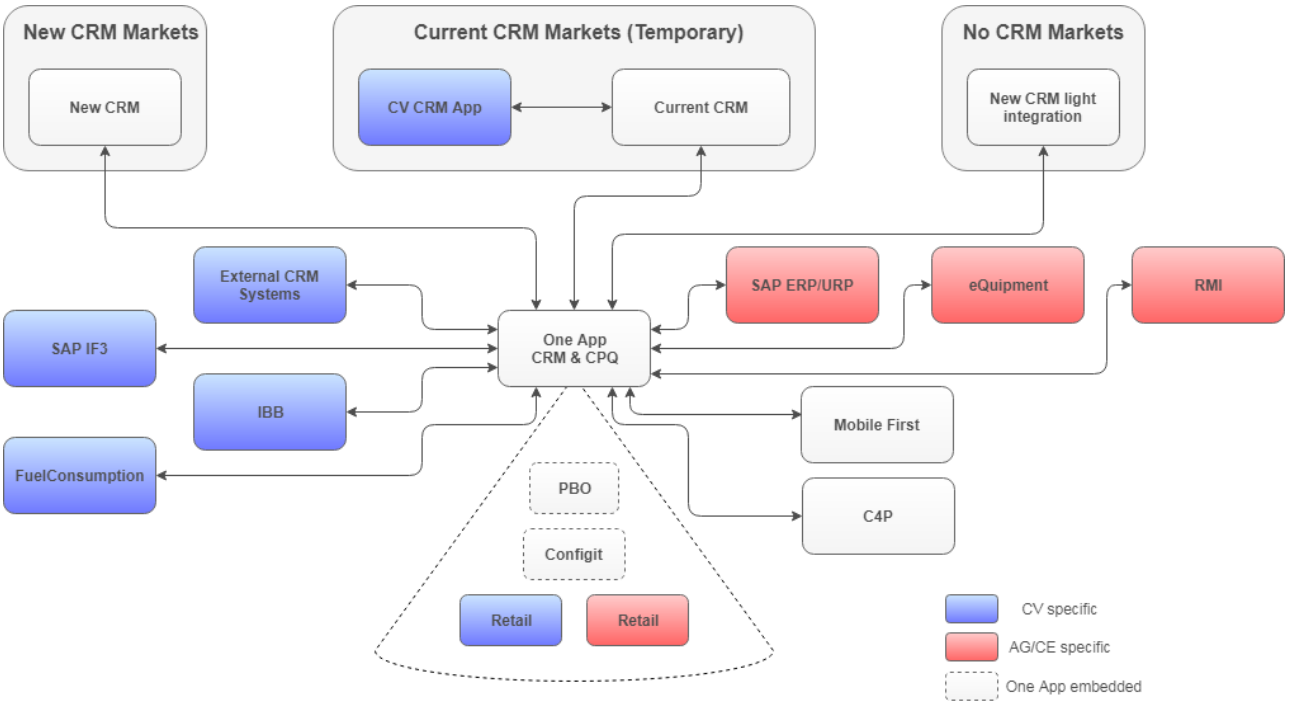

Figura 3.3 - Sales App landscape (Coolshop, 2019, "CNHi Sales App Business Blue Print")

.

Il sistema Sales App è composto dai seguenti subsystems:

Il sistema Sales App è composto dai seguenti subsystems:

- Pricebook Back office (PBO), Content Management System per il caricamento di contenuti a livello di prodotto quali immagini, descrizioni, dati tecnici e informazioni addizionali
- Configit (AG/CE and IVECO), engine di configurazione integrato nell'applicativo
- Mobile First, provide di servizi per l'autenticazione
- Retail Application, configuratore per i clienti finali embeddato sul sito dei brands

Esso è integrato con i seguenti sistemi esterni:

- SAP ERP/URP, ambienti SAP dei brands AG&CE
- SAP IF3, ambiente SAP del brand CV
- eQuipment, tool utilizzato dai Dealer per la conversione dei preventivi creati su One App in ordini
- Current CRM
- New CRM and CRM light integration
- External CRM systems (Romana Diesel, Schouten), sistemi CRM indipendenti di Dealer CV
- C4P, provider di piani di finanziamento per i clienti gestito da BNP
- RMI (AG/CE only), portale di rivendita dei veicoli usati posseduti dai Dealers
- IBB (Iveco Body Builder), tool per la creazione di allestimenti
- FuelConsumption, provider di simulazioni di consumo carburante per prodotti CV
- Dealer Locator, servizio per la localizzazione dei dealer nelle vicinanze e provider di informazioni relative ai Dealer
- BA (Cognos), tool di Business Analytics
- BA (QlikSense), tool di Business Analytics
- Data quality provider, validatore di Accounts
- Wareplace / Ubiest, normalizzatore di indirizzi

In Figura 3.4 sono mostrate le informazioni che sono scambiate con i sistemi esterni descritti in precedenza:

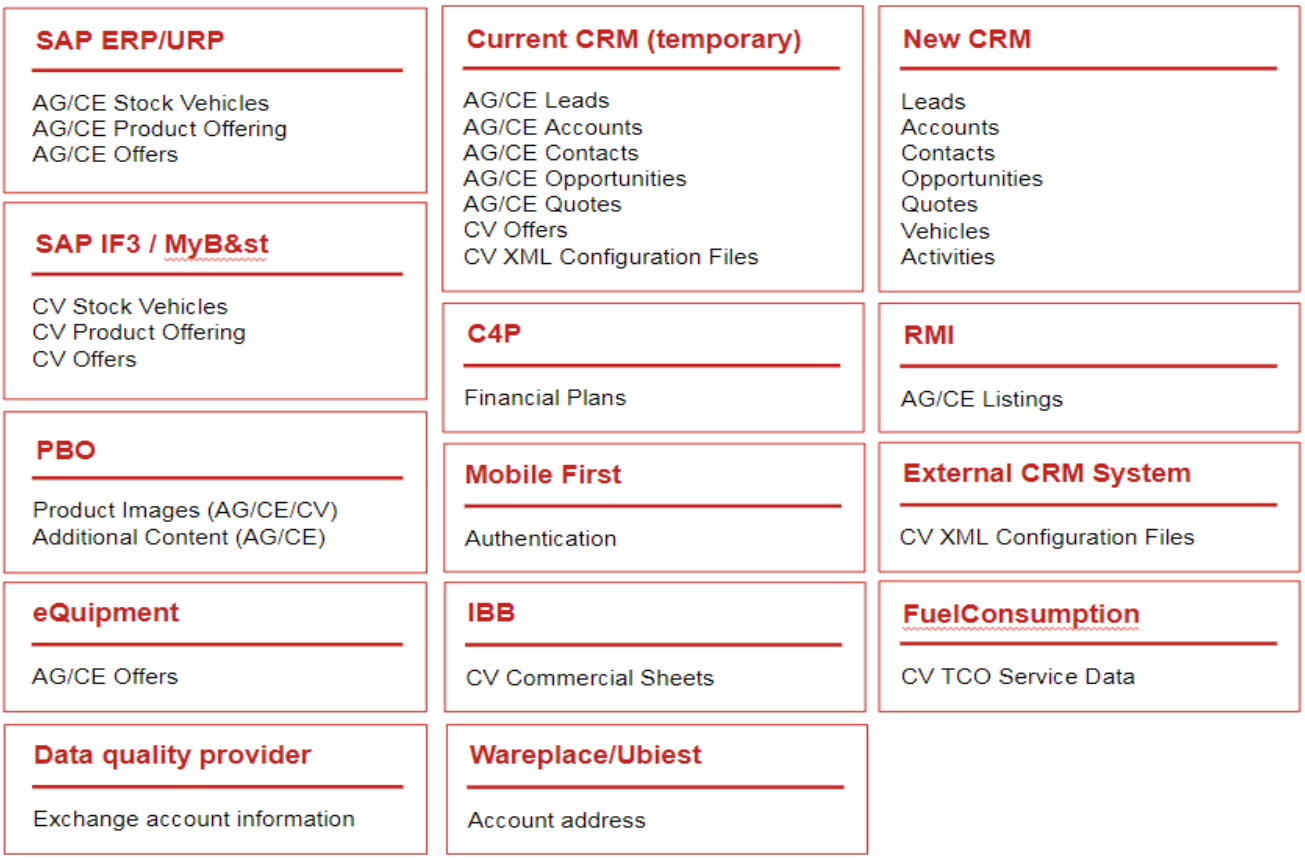

Figura 3.4 - Dati scambiati tra Sales App e i sistemi esterni (Coolshop, 2019, "CNHi Sales App Business Blue Print")

3.1.5 Login & Profiling & Visibility Rules

In Sales App l'autenticazione degli utenti è gestita tramite il servizio esterno Mobile First: tutti gli utenti devono essere profilati all'interno di esso. Il login è garantito attraverso l'utilizzo di un account (i.e., username e password) CNHi. L'applicazione è in grado di gestire differenti tipologie di utenti, quali:

- Dealer manager / Salesman
- Internal Users
- Key Account

Nelle versioni mobile dell'applicazione all'utente è richiesto di inserire un PIN per l'accesso nei seguenti scenari:

- Primo login
- All'avvio dell'applicazione se Offline (Internet connection not available)
- In caso di timeout per inattività

L'applicazione è in grado di supportare il multi-user (e.g., salesmen diversi possono condividere lo stesso tablet, e dunque la stessa applicazione). Gli utenti sono profilati basandosi sulle seguenti informazioni:

- Market
- Brand
- Ruolo

3.1.6 Layout

Sales App è disegnata al fine di garantire lo stesso layout su tutti i sistemi operativi, sulle diverse piattaforme e su device differenti.

Per tutti i brand appartenenti a CNHi Group il layout dell'applicazione è comunque, ad eccezione di customizzazioni minime derivanti dalle guidelines dei brand quali i colori del layout, le immagini mostrate e il logo del brand.

Il Main Menu è collocato nella sezione sinistra dello schermo, come shoulder bar: attraverso esso l'utente è in grado di navigare attraverso i differenti moduli dell'applicazione.

L'applicazione, all'avvio, mostra la sezione della Dashboard come Home Page di default: all'interno di essa gli utenti sono in grado di consultare un'overview dei dati più rilevanti.

3.1.7 Data Synchronization

All'interno dell'applicazione sono previsti due diversi tipi di sincronizzazione selezionabili dall'utente:

- Sincronizzazione live, gestita in background nell'applicazione
- Sincronizzazione manuale, attivata da un'azione dell'utente (ad esempio, selezionando il pulsante "Sincronizza ora")

Inoltre, è fornito all'utente una Dashboard di gestione degli errori per raccogliere tutti gli errori che si verificano durante la sincronizzazione.

Quando si lavora offline, per mantenere la coerenza dei dati sincronizzati, l'utente sarà costretto a sincronizzare dopo un determinato numero di giorni. Questa opzione è definibile per Brand e Mercato.

Al fine di aumentare l'usabilità dell'applicazione e gestire i vincoli tecnici connessi allo spazio di archiviazione dei dispositivi, la quantità di dati da sincronizzare è gestita tramite parametri definiti dall'utente nella pagina dei Settings. Questa funzionalità è disponibile anche per gli utenti interni e può essere applicata per i dati CRM e CPQ.

L'applicazione gestisce due diversi tipi di sincronizzazione dei dati:

- Sincronizza nell'applicazione (tra frontend e backend)
- Sincronizzazione con CRM 365 (tra il back-end dell'applicazione e il CRM)

La funzionalità di sincronizzazione è disponibile solo online.

Sync in app
La sincronizzazione nell'applicazione è necessaria per sincronizzare i dati tra il frontend ed il backend dell'applicazione.

Per utilizzare le versioni Mobile (iOS / Android) o Desktop (Windows) dell'applicazione è necessario sincronizzare (scaricare) tutte le informazioni sul dispositivo: questo è un processo di sincronizzazione "completo".

Dopo la prima sincronizzazione "completa", tutti gli scambi di dati tra l'applicazione ed il back-end, per mantenere le informazioni aggiornate, vengono eseguiti in "delta": solo le informazioni modificate dall'ultima sincronizzazione vengono aggiornate; questo permette di mantenere il processo di sincronizzazione il più veloce ed efficiente possibile.

Per eseguire la sincronizzazione è richiesta una connessione Internet.

Quando l'app viene utilizzata dal suo frontend mobile o desktop, i dati inseriti o modificati dall'utente vengono prima memorizzati localmente: è necessario un processo di sincronizzazione per allineare i dati con il backend; una volta nel backend dell'app One, le informazioni verranno sincronizzate con il database del CRM 365.

È possibile mettere l'applicazione in uno stato "offline" anche se il dispositivo ha una connessione Internet attiva disponibile: in questo caso è visualizzata un'icona specifica per evidenziare all'utente se l'applicazione è in modalità Online/Offline in qualsiasi momento.

Inoltre, all'interno della sezione Settings, l'utente può decidere se desidera eseguire le sincronizzazioni solo in caso di connessione Wi-Fi o anche mentre utilizza i dati mobili.

L'immagine sotto mostra il processo precedentemente descritto:

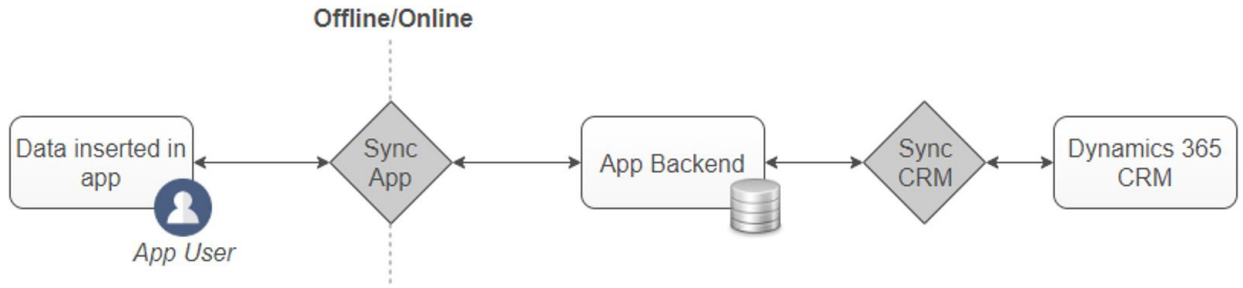

Figura 3.5 – Processo di Synch device app (Coolshop, 2019, "CNHi Sales App Business Blue Print")

L'app Web (accessibile con un browser) è accessibile solo con una connessione Internet attiva disponibile. Quando i dati vengono inseriti o aggiornati sull'app, le informazioni vengono immediatamente memorizzate sul backend. Non è possibile scrivere localmente alcun dato.

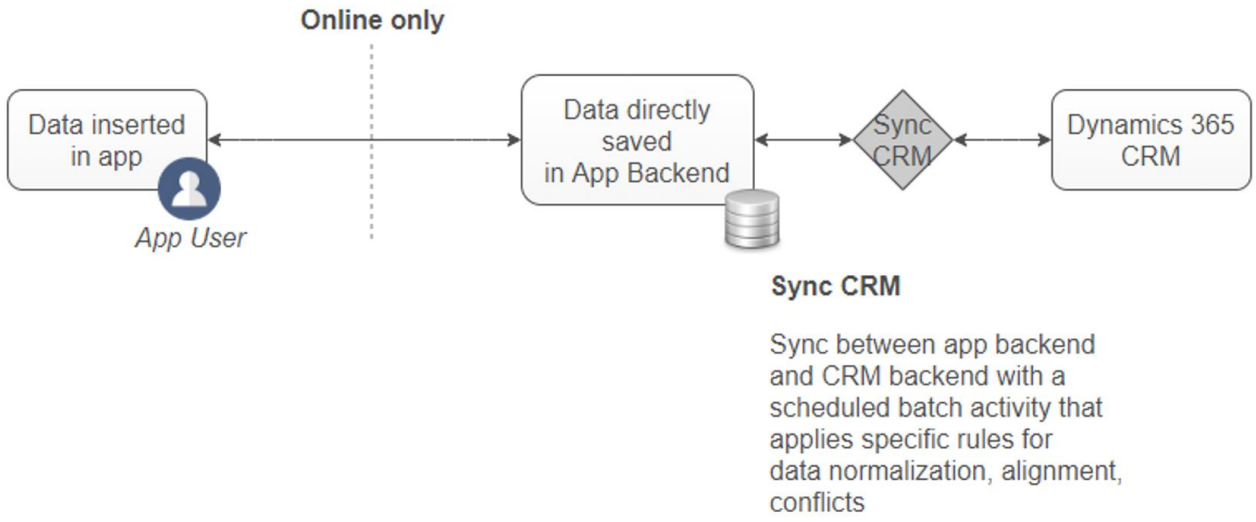

Figura 3.6 - Processo di Synch Web app (Coolshop, 2019, "CNHi Sales App Business Blue Print")

Sync col CRM

La sincronizzazione con il CRM 365 consiste in un processo di sincronizzazione tra il backend di Sales App e il backend del CRM. Questa sincronizzazione viene eseguita tramite attività batch in processi pianificati predefiniti.

Durante questo processo di sincronizzazione, le seguenti attività vengono eseguite da un layer intermedio del CRM che applica regole specifiche:

- Normalizzazione dei dati
- Allineamento dati
- Processo di risoluzione dei conflitti dei dati

Un'icona e una data di ultima sincronizzazione evidenziano nell'applicazione se un'entità è già stata sincronizzata con il sistema CRM: inoltre, un simbolo specifico evidenzia ogni entità che è stata normalizzata dal CRM.

3.1.8 Data Model

Nel seguente capitolo viene presentato il data Model di Sales App, che descrive le entità principali, le loro informazioni principali e le relazioni in cui sono coinvolte.

Lead

Un Lead è un potenziale cliente (proveniente da sito web, eventi di Google, pubblicità, ecc.) che è interessato ad essere contattato da un Dealer. Il Lead è anche il passo iniziale di una potenziale vendita. I dettagli relativi ai Leads sono visualizzati nell'applicazione.

Account

Un Account è l'entità utilizzata per identificare un cliente (normalmente una società) che può avere una relazione con il Gruppo CNH Industrial ed essere coinvolto nella sua attività.

Gli account sono essenziali per identificare e gestire i clienti, vendere prodotti e servizi e fornire servizi di qualità superiore ai clienti. I clienti/potenziali clienti sono gestiti in CRM per tracciare le informazioni utili ai venditori e ai responsabili dei processi post-vendita. L'account nel sistema è univoco per tutti i Brand e Mercati.

Contact

Un Contact è una qualsiasi persona fisica che risulta essere un referente per un Account e può essere contattato in qualche modo, ad es. tramite chiamata telefonica, SMS o e-mail o incontrato direttamente.

Vehicle

Un Vehicle è un prodotto posseduto da un Account e contiene tutte le informazioni ad esso relative.

Opportunity

Un Opportunity è l'opportunità di business relativa direttamente ad un Account; può essere considerato come l'interesse di acquisto espresso dal cliente (Account o Contact).

L'Opportunity è l'entità principale nel processo di vendita: essa difatti rappresenta una potenziale vendita per un cliente specifico.

Le Opportunity consentono all'organizzazione di monitorare il successo degli sforzi di marketing, rintracciando le vendite alla sorgente di origine e alla campagna di origine principale.

Allo stesso modo, quando si perde un Opportunity, le ragioni vengono tracciate e i dati raccolti possono essere utilizzati a scopo di analisi.

Offer

Un Offer è un'offerta commerciale di prodotti e servizi per soddisfare le esigenze dei clienti. Una nuova offerta può essere creata solo se collegata da un Opportunity o copiando un'offerta esistente già collegata ad un Opportunity.

Offer Item

L'Offer Item è l'entità utilizzata per memorizzare le informazioni del prodotto offerto, l'elemento appartenente a un'offerta, e può essere di diversi tipi (ad esempio Nuovo, Stock, Usato, etc.). Offer Item multipli possono essere aggiunti alla stessa Offer.

Activities and Agenda

Le attività sono utilizzate per organizzare e gestire i rapporti con un Account (cliente o rivenditore in caso di attività di servizio), un Contact o un Lead (e-mail, chiamata, appuntamento, test drive, etc.).

L'utilizzo delle attività aiuta a comprendere tutte le comunicazioni effettuate con ciascun cliente o potenziale cliente.

Le attività sono mostrate sia come lista che nella vista del calendario (giornaliera, settimanale e mensile) nell'applicazione.

Entities Lifecycle

Figura 3.7 mostra il ciclo di vita delle entità descritto nella sezione seguente del documento.

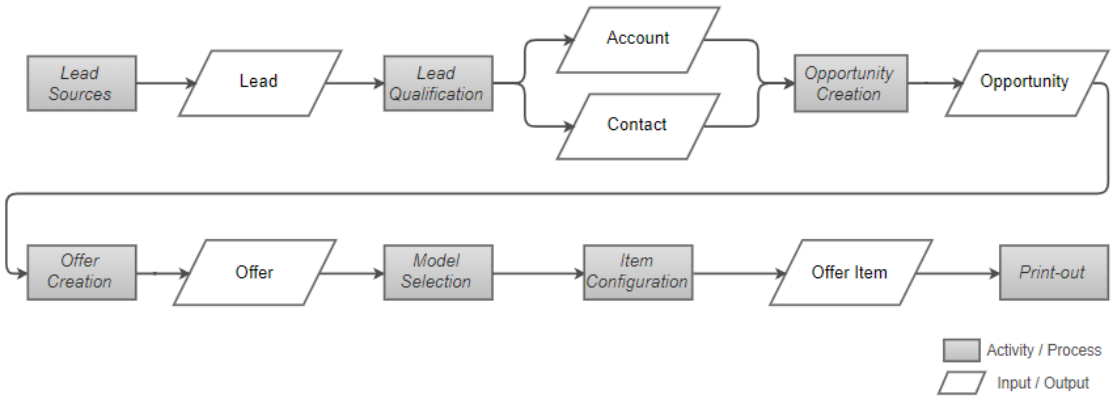

Figura 3.7 - Entities Lifecycle (Coolshop, 2019, "CNHi Sales App Business Blue Print")

La prima entità coinvolta nel flusso è il Lead, avente diversi sistemi come fonti.

Il Lead viene convertito in un Account o in un Contact attraverso la procedura di qualificazione del Lead.

Inoltre, Account e Contact possono essere creati stand-alone direttamente all'interno di Sales App o in Microsoft CRM, non essendo necessariamente il risultato del processo di qualificazione del Lead.

Attraverso il processo di creazione dell'opportunità, viene creata un'entità Opportunity.

Una o più Opportunity possono essere collegate a un singolo Account o Contact. Ogni Opportunity deve essere obbligatoriamente collegata ad un Account o Contact.

Attraverso il processo di creazione dell'offerta, viene creata un'entità Offer.

Una o più Offer possono essere collegate a una singola Opportunity. Ogni Offer deve essere obbligatoriamente collegata ad una sola Opportunity.

Un'entità Offer Item creata tramite i processi Model Selection e Configurazione del veicolo.

Uno o più Offer Item possono essere collegati a una singola Offer. Ogni Offer Item deve essere obbligatoriamente collegato add una sola Offer.

Il ciclo di vita delle entità termina con il processo di Printout dell'offerta che consente agli utenti di generare un documento che contiene informazioni su Offer ed Offer Item proposto al cliente.

E-R Diagram

Figura 3.8 mostra il diagramma E-R relativo alle entità principali di Sales App descritte nella sezione precedente del documento.

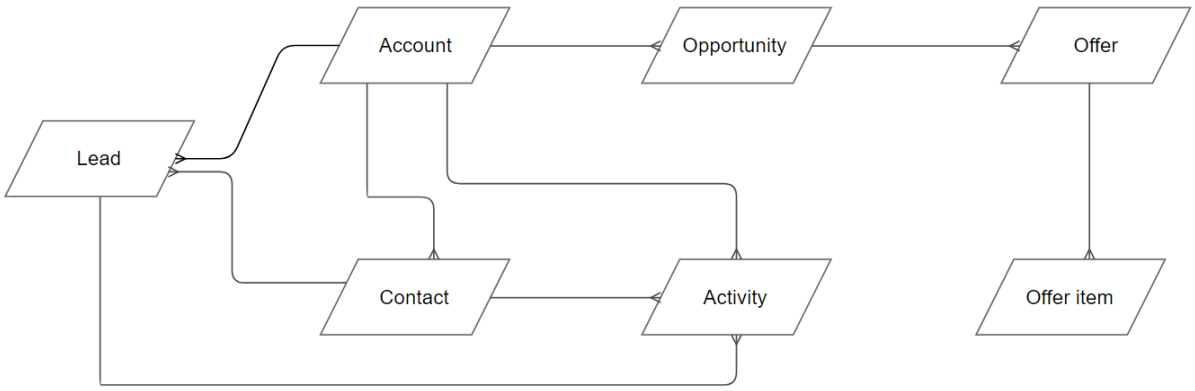

Figura 3.8 - Diagramma E-R (Coolshop, 2019, "CNHi Sales App Business Blue Print")

L'entità Lead è il punto di ingresso per l'intero processo gestito dall'applicazione. Un Account può essere associato a più Lead, mentre un Lead è sempre collegato a un singolo Account. Un Contact può essere associato a più Lead, mentre un Lead è sempre collegato a un singolo Contact. Un Account può avere più Contact, mentre un Contact appartiene sempre a un singolo Account. Un Account può essere associato a più attività e un'attività a più Account. Un Contact può essere associato a più attività e un'attività a più Contact. Un Lead può essere associato a più attività e un'attività a più Lead. Un Account può essere associato a più Opportunity, mentre un Opportunity è sempre collegata a un singolo Account. Un'Opportunity può essere associata a più Offer, mentre un Offer è sempre collegata a una singola Opportunity. Un'Offer può essere associata a più Offer Item, mentre un Offer Item è sempre collegato a una singola Offer.

3.1.9 Overview delle funzionalità principali

La soluzione proposta garantirà l'accesso alle seguenti funzionalità:

- Login & User Profiling
    - Autenticazione e profilazione
    - Visibilità dei dati basata sul ruolo degli utenti e sulla profilazione
- Data synchronization
- Leads Management
- Accounts e Contacts Management
- Opportunities e Offers Management
- Vehicles Management
- Dashboard
- Agenda e Activities Management

- Check-in/Check-out
- Around Me
- Service Management (today called “Field Support Tool”)
- Multilanguage compatibility
- Content Management (PBO) integration
- Pricebook (AG/CE Only)
- Configuration Experience
- Model Selection disponibile nelle seguenti tipologie:
    - Classic
    - Total Cost of Ownership (TCO)
    - Mission
    - Retail Experience
- Product Locator (visibilità dei veicoli a Stock dei Dealer e di CNHi)
- Used Equipment management (RMI)
- Integrazione con Mobile First per garantire le seguenti funzionalità:
    - Autenticazione
    - Certificate pinning
    - Version Management
    - Direct update
    - Live update
    - Statistiche di utilizzo & Monitoring
- Notifiche Push

3.2 Definition e Scheduling

Le funzionalità e le specifiche tecniche descritte al Paragrafi 3.1 sono il risultato di un continuo e duraturo confronto con il cliente, avvenuto durante la fase di definizione del progetto e, in seguito, durante la fase di sviluppo stessa.In particolare, il punto di partenza per la definizione dei requisiti e, dunque, delle attività in scopo è da considerarsi la fase di Business Blueprint (BBP). In questa fase vengono individuati gli obiettivi di business dell'azienda e si determinano i requisiti necessari per il loro raggiungimento.

Tutto ciò avviene attraverso attività di Workshop (incontri della durata di un'intera giornata) in cui avviene un dibattito diretto tra il fornitore ed il committente riguardante la nuova tecnologia da installare. Il documento chiave, che viene redatto dall'azienda fornitrice, è appunto la Business Blueprint. Nella Business Blueprint è presente una dettagliata rappresentazione organizzativa ed una prima stesura dei processi di business, in forma scritta e grafica, oltre che una descrizione high-level delle funzionalità che la nuova applicazione andrà a garantire. Una volta approvata, la Business Blueprint diventa il documento di riferimento dell'intero progetto ed è utilizzato come base per tutte le successive attività. L'autore dell'elaborato ha preso parte ai Workshop tenuti presso l'azienda committente per tutta la durata del periodo di definizione dei requisiti, partecipando attivamente alla creazione dei materiali (es. presentazioni, demo, etc.) necessarie per sostenere le attività di Workshop stesse. Ha altresì partecipato alla produzione della BBP dopo aver analizzato i feedback raccolti durante il workshop assieme al Team di analisti, tecnici e designer presente in Coolshop e assegnato al progetto.

Durante la fase di definizione del perimetro di Sales App, Coolshop ha provveduto dunque, in seguito a numerosi Workshop svoltisi presso il cliente, a consegnare una prima versione di BBP, che è stata revisionata dai responsabili Business dei brand componenti il Gruppo CNH Industrial. Il risultato di questa analisi è stata la collezione di nuove funzionalità, non coperte dalla versione iniziale della BBP. La fase di Business Blueprint si è dunque conclusa con l'approvazione della stessa ed il consolidamento delle funzionalità.

Al fine di avere una visione completa delle funzionalità in scope, si è dapprima suddiviso il progetto nelle seguenti macro-categorie, che identificano la tipologia di funzionalità da garantire:

- CRM: funzionalità relative al processo di Customer Relationship Management
- CPQ: funzionalità relative al processo di Configure, Quote and Price
- General: funzionalità generiche dell'applicazione
- System Integration: funzionalità che prevedono l'integrazione con sistemi esterni

Successivamente, all'interno di tali macro-categorie, si sono definite quelle funzionalità indipendenti, complete e collaudabili.

L'autore dell'elaborato, in quanto analista funzionale, ha preso parte alla definizione dei requisiti dettagliati in seguito. La lista è stata identificata attraverso attività di analisi e brainstorming interno all'azienda assieme al Team tecnico e funzionale dedicato. E' stata presa in considerazione la lista di

macro funzionalità definita nella fasi di stesura della BBP e successivamente si è andati a suddividere le stesse in funzionalità più dettagliate e puntuali.

Di seguito la lista di funzionalità, o user stories, definite (Tabella 3.1):

Tabella 3.1 – User Stories

Macro Categoria	User Story
	Configuration Experience - Classic Model Selection
	Configuration Experience - Preview and Printout
	Configurator - Features and Enhancements
	Opportunities, Offers and Offer Items Management
	Offer Printout
	Stock Management
	Product Locator
	Used Vehicles
	Offer Items Management - C4P Financing integration
	Offer and Offer Items Management - Financing Manual Calculator
	Offer and Offer Items Management - Pricing and Marginality
	NAFTA - Financing
	NAFTA - Marketing Automation
	NAFTA - Leads, Prospects Management
	CPO Process
	Configuration Experience - TCO Model Selection
	Configuration Experience - Rear & Tyres Model Selection by Mission
	Configuration Experience - Model Selection by Volume and Engine Power
	Configuration Experience - Retail
	Organizational Area (CV)
	SiQuota Integration (CV)
	SAP Integration - Offer Send to SAP (CV)
CPQ	SAP Integration - Buyback (CV)

	IBB Integration - Vehicle Technical Datasheet (CV)
CRM	Accounts and Contacts Management
	Leads Management
	Agenda and Activities Management
	Demo Units Management
	Marketing List Management
	Vehicles Management
	SuperSearch
	Dashboard
	Service Management
	Around Me
	Check-in/Check-out Management (CV)
GENERAL	App Layout, Settings, Features, Multilanguage Contents
	Data Synchronization
	Visibility Rules and User Roles
	Login Management
	GDPR Management
	Data Migration Approach
SYSTEM INTEGRATIONS	Data Integration with CRM
	WS for Offer Creation, View, Printout in CRM
	WS for Offer Creation, Printout in eQ
	Mobile First Integration
	SAP Integration - Discount & Initiatives
	Integration with Address Normalization service
	Integration with Data Quality Provider service
	Integration with External CRM systems (CV)
	QlikView and QlikSense connectors

SETUP	MySQL Database Setup
	Couch Database Setup
	Infrastructure Setup support

Le funzionalità definite sono state considerate come **User Stories**, componendo così il **Product Backlog** iniziale del progetto.

Parallelamente alla definizione delle User Stories, si è definita internamente la composizione del Team di Progetto.

In particolare sono stati individuati:

- Il Product Owner (i.e., PM Senior)
- Lo Scrum Master
- I component del Development Team così suddivisi:
    - 1 Analista funzionale senior
    - 3 Analisti funzionali junior, tra i quali l'autore dell'elaborato
    - 2 Sviluppatori senior
    - 4 Sviluppatori junior
    - 1 Designer/UI-UX Expert

Definito il Team di Progetto, il Product Owner, in accordo col cliente, ha definito per ogni User Story presente in Product Backlog la Priority relativa.

Il Development Team ha invece definito gli Story Points associati ad ogni User Story.

In dettaglio, sono stati definiti dei driver temporali di progetto sulla base dell'esperienza dei singoli e di funzionalità simili già deliverate in progetti precedentemente realizzati.
Le funzionalità sono state scomposte nelle seguenti macro-categorie:

- Nuova funzionalità
- Funzionalità esistente che richiede una review

Tale scomposizione distingue quali funzionalità sono già fornite da Coolshop a CNHi Industrial Group attraverso le applicazioni esistenti che verranno sostituite dalla nuova Sales App (funzionalità CPQ), da quelle invece mai gestite e che dovranno essere create ex novo (funzionalità CRM).

Per ognuna di queste categorie si sono definiti cinque distinti livelli di difficoltà nella realizzazione, e ad ognuno dei record risultanti da tale classificazione, è stato assegnato un effort funzionale e di design (stesura dei Solution Design) ed uno in termini di sviluppo.

L'autore della tesi, assieme al Team di progetto, ha effettuato tale operazione utilizzando la metodologia della stima per analogia – sulla base delle esperienze passate dei membri del Team relativamente a sviluppi simili o assimilabili a quelli in scopo, il Team in una riunione dedicata ha

definito ad alto livello il massimo effort necessario per ogni grado di difficoltà nella realizzazione di un'attività definito.

Lo schema risultante è stato il seguente (Tabella 3.2):

Tabella 3.2 - Driver per la stima dell'effort

Driver	Design	Sviluppo	Totale
New VH	13	25	**38**
New H	8	15	**23**
New M	5	10	**15**
New L	3	6	**9**
New VL	2	3	**5**
Review VH	5	10	**15**
Review H	4	7	**11**
Review M	2	4	**6**
Review L	1	2	**3**
Review VL	1	1	**2**

L'effort riportato è in termini di man-days (MD) richiesti per la realizzazione dell'attività.

Ogni User Story è stata poi associata ad una delle tipologie mostrate in tabella 3.3, definendone così gli Story Points relativi.

Di seguito (tabella 3.3) la lista di User Stories definite, con la relativa Priority stabilita dal Product Owner e gli Story Points assegnati dal Team di progetto:

Tabella 2.3 - Product Backlog

Macro Categoria	User Story	Story Points	Priority
CPQ	Configuration Experience - Classic Model Selection	14	1
	Configuration Experience - Preview and Printout	18	2
	Configurator - Features and Enhancements	23	1
	Opportunities, Offers and Offer Items Management	30	1
	Offer Printout	18	1

	Stock Management	9	2
	Product Locator	14	3
	Used Vehicles	14	3
	Offer Items Management - C4P Financing integration	45	4
	Offer and Offer Items Management - Financing Manual Calculator	6	5
	Offer and Offer Items Management - Pricing and Marginality	18	1
	NAFTA - Financing	14	6
	NAFTA - Marketing Automation	14	6
	NAFTA - Leads, Prospects Management	9	6
	CPO Process	9	6
	Configuration Experience - TCO Model Selection	18	4
	Configuration Experience - Rear & Tyres Model Selection by Mission	9	4
	Configuration Experience - Model Selection by Volume and Engine Power	9	4
	Configuration Experience - Retail	6	4
	Organizational Area (CV)	9	3
	SiQuota Integration (CV)	9	2
	SAP Integration - Offer Send to SAP (CV)	3	1
	SAP Integration - Buyback (CV)	9	2
	IBB Integration - Vehicle Technical Datasheet (CV)	6	2
	Accounts and Contacts Management	30	1
	Leads Management	30	2
	Agenda and Activities Management	23	2
	Demo Units Management	12	3
	Marketing List Management	23	4
	Vehicles Management	12	3
CRM	SuperSearch	14	2

	Dashboard	45	2
	Service Management	45	5
	Around Me	30	4
	Check-in/Check-out Management (CV)	23	4
GENERAL	Applicaton Layout, Settings, Features, Multilanguage Contents	45	1
	Data Synchronization	45	1
	Visibility Rules and User Roles	23	1
	Login Management	8	1
	GDPR Management	23	1
	Data Migration Approach	30	1
SYSTEM INTEGRATIONS	Data Integration with CRM	45	1
	WS for Offer Creation, View, Printout in CRM	23	2
	WS for Offer Creation, Printout in eQ	3	1
	Mobile First Integration	45	1
	SAP Integration - Discount & Initiatives	23	2
	Integration with Address Normalization service	12	2
	Integration with Data Quality Provider service	12	2
	Integration with External CRM systems (CV)	30	1
	QlikView and QlikSense connectors	45	5
SETUP	MySQL Database Setup	30	1
	Couch Database Setup	45	1
	Infrastructure Setup support	8	1

Infine, al fine di fornire al cliente un'overview della durata plausibile di progetto e dello sviluppo nel tempo dello stesso, si è comunque proceduti ad una pianificazione tramite più comune approccio Waterfall delle attività in scope.

Il piano di progetto high-level identifica, tramite un Diagramma di Gantt, le differenti fasi caratteristiche di un progetto IT - tali fasi sono state schedulate nel tempo, secondo necessità, in

modo da dare evidenza delle precedenze e definire il termine ultimo di inizio delle attività, oltre che di fine delle stesse.

La schedulazione è stata effettuata utilizzando un classico approccio Waterfall. L'autore dell'elaborato, utilizzando un tool per la creazione di diagrammi di Gantt (i.e., GanttPro), ha pianificato le principali attività in scopo (i.e., quelle relative al ciclo di vita di un classico progetto IT) con le relative precedenze, definendo per ognuna di esse una durata in termini di effort da erogare e attribuendogli delle risorse. Tramite la tecnica del crashing ha poi ottenuto il piano definitivo mostrato in Figura 3.9. In particolare si è proceduti alla parallelizzazione delle attività inizialmente definite in maniera sequenziale, mantenendo fisso il numero di risorse assegnate ad ogni fase e la relativa durata.

Il piano definito è il seguente:

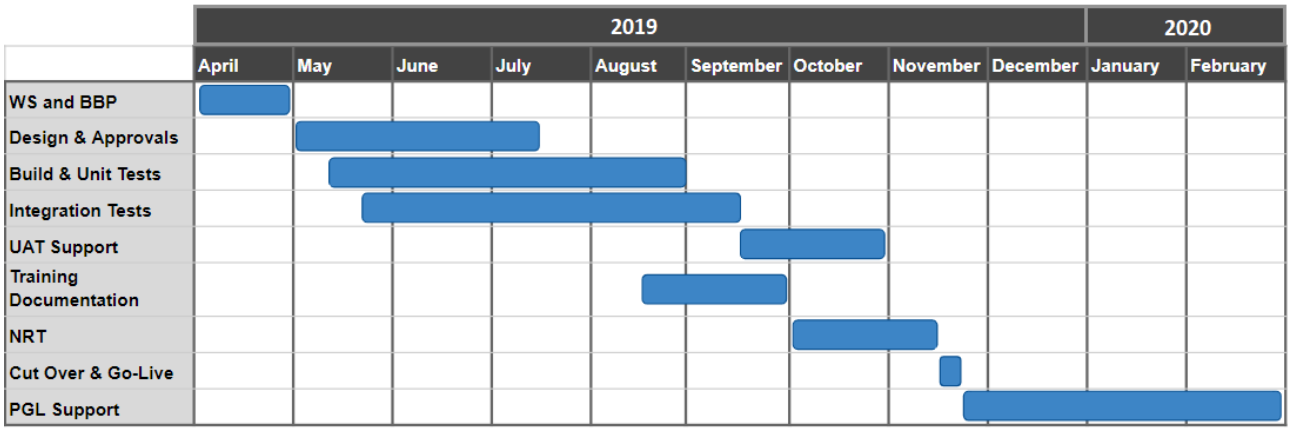

Figura 3.9 - Scheduling di progetto

Si noti che la durata della fase di sviluppo vera e propria (i.e., Design & Approvals, Build & Unit Tests, Integration Tests) è stata calcolata utilizzando le stime prodotte dal Team di Progetto ed utilizzate come Story Points.

Per il calcolo delle durate delle altre fasi invece sono stati utilizzati driver comunemente noti ed utilizzati aziendalmente, definiti come quota parte dell'effort necessario allo sviluppo.

È bene specificare che, come definito precedentemente, tale rappresentazione è stata prodotta al solo scopo di fornire al cliente una vista ad alto livello delle tempistiche necessarie al completamento del progetto e le fasi riportate non hanno avuto alcuna valenza rispetto allo sviluppo reale che è stato eseguito utilizzando un approccio Agile.

3.3 Construction & Monitoring

Come anticipato, la fase di sviluppo del progetto è stata gestita utilizzando la metodologia SCRUM.

Come tool per la gestione del progetto è stato utilizzato Jira, di proprietà di Atlassian.

L'autore dell'elaborato assieme al restante Team di progetto ha definito, in accordo col cliente, la durata dello sprint inizialmente pari a 4 settimane (i.e., 20 MD mediamente), con la fase di test interno definita di durata pari ad una settimana (i.e., 5 MD).

Durante la prima riunione di Sprint Planning, il Team Scrum ha provveduto alla creazione del primo Sprint Backlog, coadiuvati dal Product Owner di progetto. L'autore della tesi ha partecipato attivamente a tale definizione, fornendo le proprie conoscenze al fine di identificare quante e quali User Stories sarebbe stato possibile includere nel primo Sprint.

Si è dato ovviamente priorità a tutte le attività necessarie per il setup del progetto (e.g., setup ambienti, setup database, setup integrazioni). Assieme ad esse si sono inseriti inoltre Task relativi alla stesura dei Functional Designs di alcune delle funzionalità dichiarate come prioritarie dal cliente, unitamente al design delle stesse. L'autore dell'elaborato ha avuto il compito di produrre la maggior parte dei Functional Designs delle funzionalità rilasciate in Sales App nei diversi Sprint, seguendone attivamente le parti di analisi, supporto allo sviluppo e testing.

Il parametro Velocity di partenza è stato definito pari a **220** Story Points dal Team di progetto, in considerazione della durata dello sprint e del numero di risorse disponibili (11 risorse full-time per 20 giorni uomo) – si supponeva dunque di essere in grado di fornire 220 MD di effort per ogni Sprint.

Le User Stories facenti parte dello Sprint Backlog sono state suddivise in Task dal Development Team e lo stesso ha provveduto a stimare l'effort reale (in ore) per il completamento dei singoli Task.

Con l'avvio dello Sprint, lo Scrum Team, incluso l'autore della tesi, ha partecipato giornalmente alla cerimonia del Daily Scrum Meeting.

Come da letteratura, la durata del Daily Scrum è stata definita pari a 15 minuti e i temi toccati a turno dai partecipanti sono stati i seguenti:

- Cosa ho fatto ieri?
- Cosa farò oggi?
- Quali impedimenti rallentano o bloccano i miei progressi?

Verrà fornita in seguito un'overview dei Daily Scrum e alcune considerazioni conclusive sugli stessi.

Già dai primi Daily Scrum ci si è resi conto che la Velocity definita non era inizialmente raggiungibile.

Le principali ragioni alla base di ciò erano le seguenti:

- Mancanza di informazioni
- Tempi di setup più lunghi del previsto

- Dipendenza da terzi per alcuni temi relativi al Setup degli ambienti che causavano delay rispetto ai tempi preventivati
- Efficienza interna ridotta a causa di mancanza di organizzazione e capacità collaborativa iniziale del Team

Il primo Sprint si è dunque concluso con un completamento solo parziale di alcune attività inizialmente parte dello Sprint Backlog.

Il dettaglio delle Stories completate per ogni Sprint verrà fornito in seguito.

Al termine del primo Sprint si è tenuta la prima Cerimonia di Sprint Review.

Alla stessa hanno partecipato tutti gli stakeholders, oltre che lo Scrum Team.

Quest'ultimo ha presentato le User Stories completate, risposto alle domande degli stakeholders e validato con questi ultimi le stesse. Un esempio abbastanza intuitivo è l'approvazione degli FDD prodotti dall'autore dell'elaborato che sono stati revisionati assieme agli stakeholders proprio durante le cerimonie di Sprint Review.

Il cliente ha fornito dunque il benestare per il rilascio, definendo le Stories effettivamente Done. Le User Stories non Done sono state reintrodotte nel Product Backlog e il Product Owner ha avuto il compito di rivedere le Priority.

Il primo Sprint è stato dunque rilasciato in ambiente di produzione.

In seguito si è tenuto la prima cerimonia di Sprint Retrospective.

La durata, come da letteratura, è stata definita pari a 3 ore e si è posto l'attenzione sulle seguenti domande:

- Cosa è andato bene durante lo Sprint precedente?
- Cosa possiamo migliorare?
- Cosa ci impegniamo a migliorare nel prossimo Sprint?

In dettaglio, le seguenti colonne sono state riportate su di una lavagna:

- Cosa è andato bene
- Cosa è andato storto
- Action Points (o Miglioramenti)

Ad ogni partecipante, incluso l'autore della tesi, è stato distribuito un set di post-it sul quale scrivere le proprie osservazioni, o idee.

Il numero di post-it assegnati ad ogni membro dello Scrum Team è stato pari a 5.

Un membro del Team alla volta, in seguito alla compilazione dei post-it, ha raggiunto la lavagna per condividere le proprie osservazioni, spiegarle rapidamente al resto del Team e inserirle infine sulla stessa.

Al termine di questa fase lo Scrum Master, con l'aiuto del Team, ha avuto il compito di raggruppare tutti i post-it che esprimevano pensieri simili e si è passati infine alla fase di votazione. Ogni membro del Team poteva esprimere un numero limitato di voti secondo la seguente formula:

$$\frac{n}{3} + 1$$

dove n è definito come il numero di temi (inteso come gruppi di post-it e post-it rimasti singoli) che il Team ha identificato.

Finita la parte di votazione, lo Scrum Master ha riordinato i temi in base al numero di voti ricevuti e quelli non votati sono stati archiviati.

Si è poi iniziato con la discussione a partire dal tema più votato, cercando di mantenere un tempo di 5 minuti per ciascun argomento, con l'obiettivo di identificare un'azione (Action Point) da riportare nella colonna corrispettiva. Per ognuno di essi è stato inoltre aggiunto un Owner, dove necessario.

Alcuni esempi di Action Points individuati durante il Retrospective relativo al primo Sprint sono i seguenti:

- Ottimizzazione della comunicazione tra le risorse interne
- Aumento della velocità di risposta da parte dei fornitori terzi e del cliente per temi Pending
- Aumento delle informazioni di dettaglio riportate nelle User Stories e nei Task
- Riduzione della durata del Daily Sprint tramite rimozione di discussioni tecniche

Prima di procedere con lo Sprint Planning dello Sprint successivo è stato infine calcolato il parametro Velocity erogato dal Team erogato durante il primo Sprint, che è stato pari a **112** Story Points.

Concluso dunque a tutti gli effetti il primo sprint di progetto si è provveduto ad effettuare lo Sprint Planning del secondo Sprint, reinserendo gli items non Done rimasti dal primo Sprint.

I processi finora descritti sono stati ripetuti in maniera iterativa nel corso degli sprint successivi, dei quali risparmiamo i dettagli.

Il numero di Sprint totale è stato pari a **7**.

La versione finale dell'applicativo rilasciato includeva le seguenti funzionalità (Tabella 3.4):

Tabella 3.4 - CNHi Sales App funzionalità definitive

Macro Categoria	User Story
CPQ	Configuration Experience - Classic Model Selection
	Configuration Experience - Preview and Printout
	Configurator - Features and Enhancements
	Opportunities, Offers and Offer Items Management
	Offer Printout
	Stock Management
	Product Locator
	Used Vehicles
	Offer Items Management - C4P Financing integration
	Offer and Offer Items Management - Pricing and Marginality
CRM	Accounts and Contacts Management
	Leads Management
	Agenda and Activities Management
	Demo Units Management
	Vehicles Management
	SuperSearch
	Dashboard
	Around Me
GENERAL	Applicaton Layout, Settings, Features, Multilanguage Contents
	Data Synchronization
	Visibility Rules and User Roles
	Login Management
	GDPR Management
	Data Migration Approach
SYSTEM INTEGRATIONS	Data Integration with CRM
	WS for Offer Creation, View, Printout in CRM

	WS for Offer Creation, Printout in eQ
	SAP Integration - Discount & Initiatives
	Integration with Address Normalization service
	Integration with Data Quality Provider service
SETUP	MySQL Database Setup
	Couch Database Setup
	Infrastructure Setup support
NEW ITEMS	Subdealer Management
	Phased Out Management
	SAP Configurator
	Demo Process Optimization
	Offer Printout Optimization
	Offer Items Pricing Recalculation
	Compatible Products Management
	STST Integration
	Countly Integration
	Application Lists Optimization
	CRM Integration Rework
	Graphical Otpimization
	Login via DP/ADFS Azure

Di seguito (Tabella 3.5) si riporta la lista di Stories incluse in ogni Sprint:

Tabella 3.5 - User Stories per ogni Sprint

Sprint	User Story
S1	Applicaton Layout, Settings, Features, Multilanguage Contents
	Login Management
	MySQL Database Setup
	Couch Database Setup
	Infrastructure Setup support
S2	Configuration Experience - Classic Model Selection
	Configurator - Features and Enhancements
	Accounts and Contacts Management
	Visibility Rules and User Roles
	Data Integration with CRM
	Login via DP/ADFS Azure
S3	Configuration Experience - Preview and Printout
	Offer Printout
	Offer and Offer Items Management - Pricing and Marginality
	Data Synchronization
	GDPR Management
	Data Migration Approach
	WS for Offer Creation, Printout in eQ
	SAP Integration - Discount & Initiatives
S4	Opportunities, Offers and Offer Items Management
	Stock Management
	Leads Management
	Agenda and Activities Management
	Dashboard
	WS for Offer Creation, View, Printout in CRM
	Integration with Data Quality Provider service

	Countly Integration
S5	Product Locator
	Used Vehicles
	Offer Items Management - C4P Financing integration
	Demo Units Management
	Vehicles Management
	SuperSearch
	Around Me
	Integration with Address Normalization service
	CRM Integration Rework
S6	Subdealer Management
	SAP Configurator
	Demo Process Optimization
	Offer Printout Optimization
	STST Integration
	Application Lists Optimization
	Graphical Otpimization
S7	Phased Out Management
	Offer Items Pricing Recalculation
	Compatible Products Management

Nel corso della vita del progetto sono da segnalare i seguenti highlights:

- Numerose funzionalità sono state rimosse perché ritenute non più necessarie
- Numerose funzionalità sono state aggiunte perché ritenute indispensabili
- Le funzionalità rilasciate sono state spesso oggetto di revisione, modifica, ottimizzazione
- Il brand CV ha deciso di non proseguire con il progetto – tutte le funzionalità specifiche sono state quindi rimosse
- La region NAFTA (North America) ha deciso di non proseguire con il progetto – tutte le funzionalità specifiche sono state quindi rimosse

Il progetto è stato rilasciato in Go-Live agli end users (i dealers e gli internal users CNHi) per i mercati in scope.

Il parametro Velocity è stato incrementato nel corso della vita del progetto, sprint per sprint, fino ad assestarsi ad un valore medio pari a **137** Story Points. Tale valore è stato dunque il massimo quantitativo di effort che il Team Scrum definito all'inizio del progetto è stato in grado di erogare.

Durante le varie iterazioni, ed in particolare in fase di UAT dello sprint da parte del cliente, è stato quasi immediatamente evidente come la pianificazione di massima condivisa ad inizio progetto non sarebbe stata rispettata. Le ragioni alla base di tale avvenimento sono le seguenti:

- Rendimento ed efficienza del Team ridotti rispetto alla capacità lavorativa preventivata
- Tempi di attesa di documentazione o sviluppi da parte di fornitori terzi e/o del cliente non preventivabile
- Aggiunta di funzionalità non incluse nel perimetro iniziale

3.4 Reporting

Di seguito sono riportati i più comuni grafici di monitoraggio progetto gestito con metodologia SCRUM, applicati al progetto CNHi Sales App.

Si noti che le informazioni relative al monitoraggio costi e, più in generale, tutto ciò che ha a che fare con gli Economics di progetto è stato volutamente escluso dall'elaborato essendo dati sensibili.

Hit Rate by Sprint

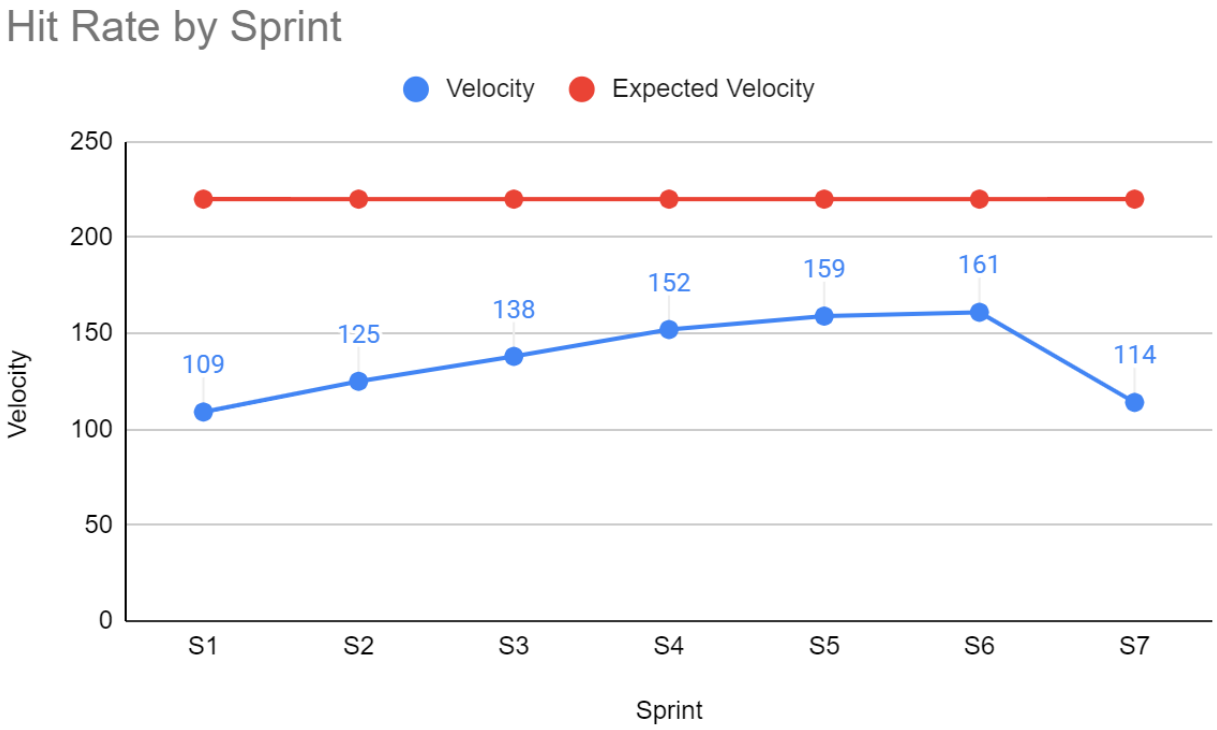

Figura 3.10 - Hit Rate Chart

La Figura 3.10 riporta i valori di Velocity registarti durante la vita del progetto.

Ad ogni Sprint è stato calcolato il valore di Velocity raggiunto. Tale valore (curva "Velocity") è stato messo a paragone con il valore ideale di Velocity preventivato ad inizio progetto (curva "Expected Velocity"), definito nel paragrafo 3.3.

Come si può notare, nonostante il valore effettivo sia risultato nettamente inferiore rispetto a quello stimato, esso è stato comunque migliorato ad ogni Sprint, con andamento linearmente crescente.

Le ragioni di tale crescita sono da ritrovarsi nell'ottimo lavoro svolto dal Team nel risolvere i problemi individuati durante i vari Sprint Retrospective, oltre che dall'ottimizzazione dell'organizzazione interna.

Release (or Product) Burn Down Chart

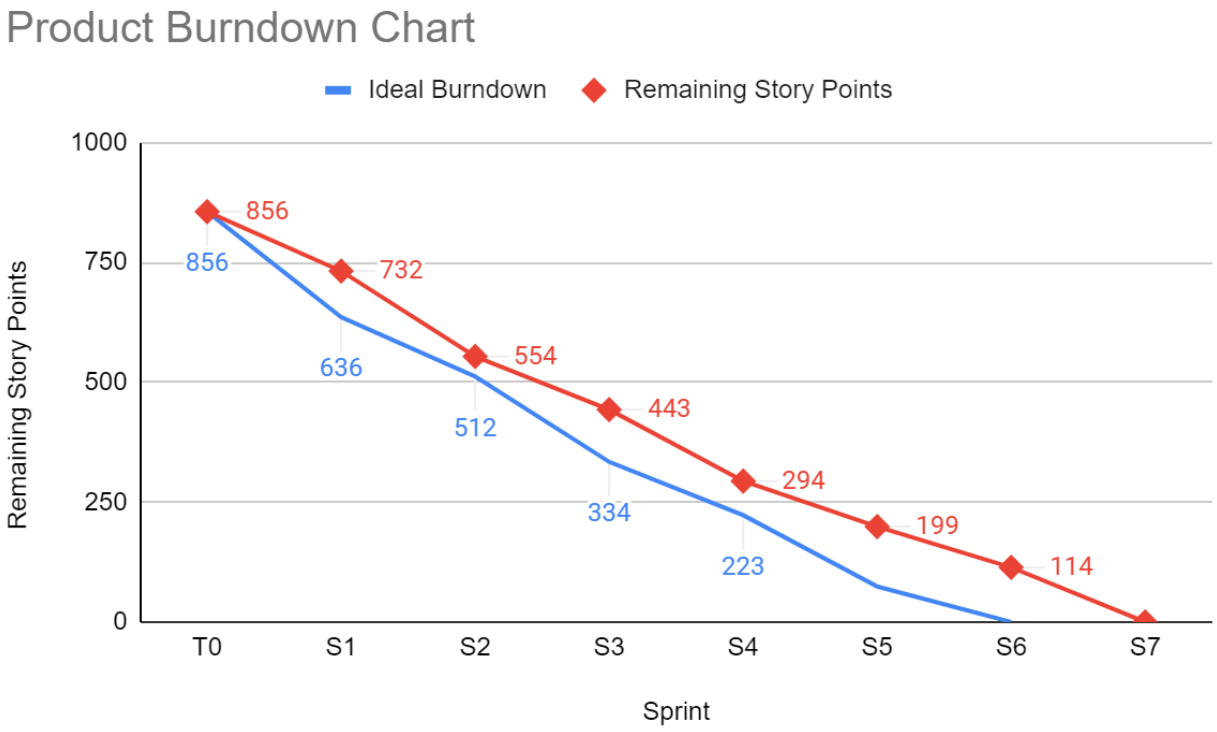

Figura 3.11 Product Burndown Chart

Figura 3.11 riporta la serie storica di Story Points erosi nel corso di vita del progetto in relazione al numero di Sprint effettuati. In dettaglio, si è calcolato il totale degli Story Points da erogare (i.e., Story Points del Produtc Backlog) e si è poi ridotto tale numero ad ogni sprint, sottraendogli gli Story Points totali delle User Stories completate in quello specifico sprint.

Sull'asse delle ascisse è riportato il numero totale di Story Points rimanenti, mentre sull'asse delle X gli Sprint eseguiti.

Si riporta inoltre il confronto tra il burndown ideale, ovverro quello calcolato considerando un valore di Velocity pari a 220, inizialmente preventivata, con il burndown effettivo.

Come si evince dal grafico, la curva di burndown effettivo si è distaccata nettamente da quella che indica il valore ideale, principalmente a causa del parametro Velocity effettiva che si è registrata che, come specificato in precedenza, è stata nettamente inferiore a quanto preventivato.

Si noti inoltre come nonostante il parametro Velocity registrata sia stata lineramente crescente, non è stata registrata una decrescita lineare degli story points rimanenti, in quanto nel corso del progetto sono state aggiunte funzionalità con un valore di story points superiore a quelli delle stories rimosse.

Il numero di Story points finale del progetto è stato pari a **958**, circa 100 in più del valore iniziale.

Tale informazione si può evincere chiaramente nei valori registrati nei nodi relativi ad S3 e S5 della curva Remaining Story Points.

Stories by Sprint

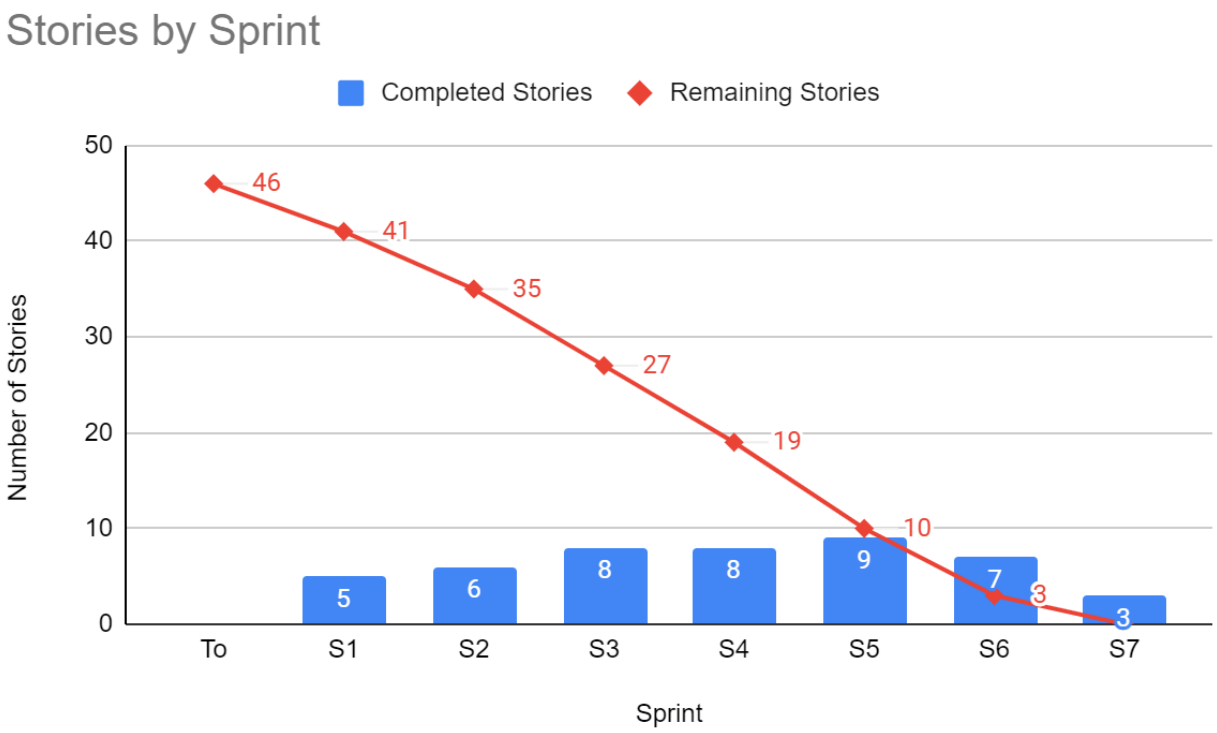

Figura 19 - Stories by Sprint

Figura 3.12 riporta il numero di User Stories rimanenti al termine di ogni sprint, in relazione col quello delle User Stories completate durante quello stesso sprint.

Tale metrica è utile al fine di comprendere la capacità del Team di apprendere, risolvere gli impedimenti riscontrati durante gli Sprint passati e aumentare dunque la produttività.

Come si nota dal grafico, nel corso della vita del progetto il Team è stato in grado di aumentare significativamente la propria produttività, arrivando quasi a raddoppiarla in S5.

Il numero totale di Stories considerato è ovviamente quello basato sulla lista definitiva di funzionalità effettivamente sviluppate (non considera dunque quelle rimosse durante il progetto ed include quelle invece aggiunte).

3.5 Riepilogo finale

Di seguito è presentata una sintesi tabellare dei dati presentati all'interno di questo capitolo.

Hit Rate by Sprint

Tabella 3.6 - Hit Rate by Sprint

	Sprint						
	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**
Expected Velocity	220	220	220	220	220	220	220
Velocity	109	125	138	152	159	161	114
Delta	-111	-95	-82	-68	-61	-59	-106

Come mostrato in Tabella 3.6 la Velocity preventivata, definita in fase di avvio del progetto, non è stata raggiunta dal Team di progetto. La Velocity effettiva è stata difatti inizialmente molto inferiore a quella preventivata (i.e., circa il 50%) ma si è osservato un trend crescente della stessa al susseguirsi degli sprint, sintomo evidente della capacità di crescita e miglioramento del Team stesso.

Release (or Product) Burn Down Chart

Tabella 3.7 – Product Burn Down Chart

	Sprint							
	T0	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**
Ideal Burndown	856	636	512	334	223	112	0	0
Remaining Story Points	856	732	554	443	294	199	114	0

Come mostrato in Tabella 3.7 il Burndown ideale previsto in fase di avvio del progetto non è coerente con quello effettivamente osservato al termine dello stesso. Le principali cause di tale scostamento sono da individuarsi nell'erronea definizione del parametro Velocity preventivato oltre che nell'aggiunta di Story Points legati a nuove richieste del cliente nel corso della vita del progetto.

Stories by Sprint

Tabella 3.8 – Stories by Sprint

	Sprint							
	T0	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**
Remaining Stories	46	41	35	27	27	19	10	3
Completed Stories	0	5	6	8	8	9	7	3

I dati in Tabella 3.8 consentono di ottenere una vista ad alto livello dell'efficienza del Team di progetto durante il ciclo di vita dello stesso. Risulta evidente, sulla base delle Stories effettivamente completate ad ogni sprint, come il Team sia stato in grado di aumentare la propria produttività, coerentemente con quanto indicato dal parametro Velocity effettivo registrato.

4. Conclusioni

Di seguito si riportano le considerazioni finali relative al progetto oggetto di tesi.

4.1 Benefici per l'azienda

Il progetto CNHi Sales App ha segnato per Coolshop un vero e proprio punto di svolta nella vita dell'azienda, per grandezza dello stesso sicuramente, ma anche e soprattutto per l'approccio che è stato scelto per la sua esecuzione. È stato di fatto il primo vero progetto gestito con approccio Agile, in particolare utilizzando la metodologia SCRUM, ed è dunque risultato sicuramente sfidante, logorante per certi versi, ma anche gratificante.

Le principali difficoltà che si è dovuti affrontare sono state collegate a due fattori che si potrebbero definire entrambi "di novità". In dettaglio, come brevemente anticipato all'inizio del paragrafo, il progetto descritto nell'elaborato è stato il primo vero grande progetto in termini di risorse coinvolte, requisiti richiesti dal cliente e anche durata delle attività che l'azienda si è trovata ad affrontare. Basti pensare che la stessa è cresciuta del 50% in termini di dipendenti assunti nel periodo descritto. Parallelamente, la scelta di applicare ad un progetto di tale complessità la metodologia SCRUM, mai realmente utilizzata in azienda e nota solo in maniera teorica, ha amplificato la difficoltà. È stato dunque altamente complesso e sfidante, in particolare nei primi mesi di vita del progetto, riuscire a trovare i giusti processi e la giusta organizzazione del lavoro, ed ha richiesto un sacrificio iniziale per i dipendenti dell'equilibrio vita-lavoro.

Nonostante un avvio poco ottimale e caratterizzato da inefficienza, a causa delle lacune organizzative e cognitive rispetto ai processi, l'azienda e le persone coinvolte in dettaglio sono state in grado di reagire alla sfida uscendone al fine vincitori.

Per tale ragione ed in considerazione dell'elevata complessità del progetto, sia in termini di logiche da sviluppare quanto di numerosità di funzionalità richieste, la metodologia SCRUM si è rivelata quella ottimale per la gestione del progetto stesso, a maggior ragione se si tiene in considerazione l'elevata volatilità delle funzionalità in perimetro.

Numerose sono state le funzionalità rimosse rispetto al perimetro iniziale, altrettanto numerosa è la lista di funzionalità richieste durante gli sviluppi perché definite fondamentali nel momento in cui il progetto prendeva effettivamente forma, e non sottovalutabile è inoltre il numero di rework, ottimizzazioni ed evolutive relativi alle funzionalità che erano già state preventivate nelle fasi iniziali.

L'approccio Agile, e in particolare la metodologia SCRUM, ha permesso di sviluppare un prodotto che si adattava realmente alle necessità del cliente, sia quelle note in fase di definizione del progetto che soprattutto quelle richieste ed evidenziate durante le fasi successive.

Se fosse stato utilizzato il più classico approccio Waterfall il risultato del progetto sarebbe stato un prodotto rilasciato comunque in ritardo rispetto al piano condiviso, probabilmente con un delay maggiore rispetto a quello effettivo, che non era inoltre in grado di attendere realmente alle aspettative del cliente in quanto mancante di funzionalità, logiche ed ottimizzazioni ritenute fondamentali.

Sarebbe stata inoltre di una complessità davvero elevata la gestione del retreating del brand CV in fase di sviluppo – dell'effort era stato già erogato per il setup, la produzione di documentazione e lo sviluppo di funzionaità necessarie per il brand e se non si fosse utilizzato la metodologia SCRUM sicuramente molto altro ne sarebbe stato erogato inutilmente.

Il metodo SCRUM difatti ha permesso di avere una *responsiveness* al cambiamento elevatissima, che ha permesso all'azienda di evitare quasi immediatamente lo spreco di tempo e risorse.

Infine, essendo SCRUM una metodologia fatta di processi iterativi, si segnala che il progetto CNHi Sales App è al momento ancora in progress e mensilmente gli sprint proseguono con l'inclusione continua di nuove funzionalità, fixing di funzionalità già rilasciate in produzione o rework sulle funzionalità esistenti.

4.2 Limiti della tesi

L'elaborato non include al suo interno la trattazione degli aspetti economici della commessa (i.e., rendicontazione progettuale, economics di progetto) e i KPI legati a tale tema, in quanto ritenuti dati sensibili dall'azienda.

È altresì mancante la trattazione delle principali pratiche di Risk Management in quanto non in scopo per il progetto descritto nell'elaborato.

4.3 Sviluppi Futuri

L'esperienza maturata grazie al progetto descritto nell'elaborato consentirà all'azienda Coolshop SRL di proseguire nel suo percorso di crescita, organizzazione e consolidamento.

I risultati ottenuti attraverso l'applicazione della metodologia SCRUM sono stati eccezionali, soprattutto in considerazione del fatto che si trattava per Coolshop del primo tentativo di gestione di un progetto con tale metodologia.

La consapevolezza acquisita, unitamente alla crescita delle competenze delle risorse coinvolte, avrà il naturale effetto di far propendere in futuro l'azienda verso la gestione dei nuovi progetti che saranno acquisiti tramite approccio Agile.

Non si esclude inoltre che si potrebbe tentare di mutare l'approccio Waterfall utilizzato attualmente su progetti già avviati nell'approccio Agile oggetto dell'elaborato, consapevoli dei miglioramenti evidenti che questo porterebbe che si rifletterebbero in benefici per l'azienda stessa ma anche per i clienti.

Bibliografia

Cantamessa, M. and Cobos, E. and Rafele, C. (2007), *Il Project Management – Un approccio sistemico alla gestione dei progetti*, ISEDI

Lappe, M. and Spang, K. (2014), "Investments in project management are profitable: A case study-based analysis of the relationship between the costs and benefits of project management", *International Journal of Project Management*, Vol. 32 No. 4, pp. 603-612.

Miller, D.P. (2008), *Building a project work breakdown structure: Visualizing objectives, deliverables, activities, and schedules*, Crc Press

Velpuri, R. and Das, A. (2011), "CA Clarity PPM Organizational Breakdown Structure"*, Clarity PPM Fundamentals*.

Stewart, R. A. (2008), "A Framework for the Life Cycle Management of Information Technology Projects – Project IT"*, International Journal of Project Management*, Vol. 26, pp. 203-212

Maravas, A. and Pantouvakis, J.P. (2012), "Project cash flow analysis in the presence of uncertainty in activity duration and cost", *International journal of project management*, Vol. 30 No. 3, pp. 374-384

Basu, A. and Muylle, S. (2007), "How to Plan E-business Initiatives in Established Companies", *MIT Sloan Management Review*, Vol. 49 No. 1, pp. 28-36

Hadaya, P. and Cassivi, L. and Chalabi, C. (2012), "IT project management resources and capabilities: a Delphi study", *International Journal of Managing Projects in Business*, Vol. 5 No. 2, pp., 216-229

Goldratt, E. M. (1994), *Theory of Constraints*, North River Pr.

Beck, K. and Fowler, M. (2000), *Planning Extreme Programming*, Addison-Wesley Professional

Stapleton, J. (1997), *Dsdm Dynamic Systems Development Method: The Method in Practice: A framework for business centered development*, Addision-Wesley

Palmer, S.R. and Felsing, J.M (2002), *A Practical Guide to Feature-Driven Development*, The Coad Series

Taylor, K.J. (2016), "Adopting Agile software development: the project manager experience", *Information Technology & People*, Vol. 29 No. 4, pp. 670-687

Eder, S. and Conforto, E. C. and Amaral, D. C. and da Silva, S. L. (2015), "Differentiating traditional and agile project management approaches", *Producao*, Vol. 25 No. 3, pp. 482-497

Sitografia

[1] https://www.pmi.org/about/learn-about-pmi/what-is-project-management

[2] https://www.pinterest.it/pin/538039486703500073/

[3] http://www.free-management-ebooks.com/faqpm/principles-08.htm

[4] https://study.com/academy/lesson/phases-of-the-it-project-life-cycle.html

[5] https://www.pmi.org/about/learn-about-pmi/who-are-project-managers

[6] https://www.projecttimes.com/articles/using-work-breakdown-structure-wbs-for-effective-project-estimation.html

[7] https://vitolavecchia.altervista.org/che-cose-lobs-organization-breakdown-structure-e-la-matrice-delle-responsabilita/

[8] https://www.pmi.it/impresa/business-e-project-management/articolo/557/la-matrice-di-responsabilit.html

[9] https://www.pinterest.it/pin/464011567862304392/

[10] https://www.grapecity.com/componentone/docs/win/online-flexchart/gantt-chart.html

[11] http://uc-mtk.blogspot.com/2012/08/tecniche-di-pm.html

[12] https://www.productplan.com/glossary/pert-chart

[13] https://it.wikipedia.org/wiki/PERT/CPM/

[14] https://www.invensislearning.com/articles/pmp/precedence-diagramming-method

[15] https://www.workpack.in/2018/07/04/s-curves-for-tracking-construction/

[16] https://www.pmtsi.com/wp-content/uploads/2020/01/2020-01-25-Earned-Value-Questo-Sconosciuto.pdf

[17] https://www.pmi.it/impresa/business-e-project-management/articolo/1755/agile-project-management.html

[18] https://twproject.com/it/blog/il-metodo-waterfall-cose-e-cosa-serve/

[19] https://www.linkedin.com/pulse/quando-%C3%A8-possibile-utilizzare-il-metodo-scrum-francesco-castellana

[20] https://clearlyagile.com/agile-blog/the-ultimate-guide-to-the-sprint-backlog

[21] https://www.researchgate.net/figure/Sample-burndown-chart_fig1_321741375