POLITECNICO DI TORINO

CORSO DI LAUREA MAGISTRALE IN

INGEGNERIA GESTIONALE E DELLA PRODUZIONE INDUSTRIALE

TESI DI LAUREA

Valutazione della bontà dell'implementazione della metodologia Agile in una società di servizi

RELATORE

PROF. DE MARCO ALBERTO

CANDIDATO

ALDO MANCINI

Anno Accademico 2017/2018

Alla mia magnifica famiglia,
che mi ha sempre seguito e supportato
e a Sara una persona stupenda con cui ho
condiviso questo percorso
Grazie

SOMMARIO

INTRODUZIONE

Il numero di progetti a giorno d'oggi è in continuo aumento, si sta' sempre più passando da un approccio orientato allo sviluppo di un singolo prodotto, slegato rispetto all'intera processo di cui fa parte, a un approccio complesso di gestione di tutte le fasi che portano alla realizzazione dello stesso. Questo cambiamento è frutto di necessità e cambiamenti che stanno interessando tutte le aziende da quella manifatturiera a quelle di servizi.

Il forte aumento dei progetti, ha portato ad un incremento di progetti senza successo o con scarsi risultati o, comunque al disotto delle aspettative attese e l'utilizzo di un sforzo incongruente. Tale circostanza genera impatti diretti e indiretti sulle organizzazioni. Il rapporto, stretto, tra progetto e organizzazione ruota attorno ad aspetti peculiari:

- Il budget, lo sviluppo di progetti richiede a sforzi economici ingenti, e, l'immobilizzazione di capitale su un progetto non permette l'impiego delle stesse risorse per un altro
- La scelta strategica, la scelta del progetto a livello strategico è fondamentale perché questo non risulti un fallimento in termini pratici sebbene formidabile dal punto di vista della sua esecuzione.

Potrebbe sembrare che entrambi questi punti esulino dall'ambito della pianificazione, esecuzione e gestone di un progetto, poiché tali variabili vengono date all'inizio di una pianificazione e vengono prese ad un livello decisionale più alto rispetto a quello operativo. In realtà una corretta esecuzione e pianificazione può ad esempio permettere un risparmio di budget, o almeno il suo rispetto e può anche permettere una migliore identificazione degli obbiettivi progettuale e rendere il loro raggiungimento il più veloce possibile riducendo il rischio di progetti che al termine della loro realizzazione non hanno più quell'attrattività che avevano al loro inizio. Perché alla fine il progetto non è di successo solo se rispetta i tempi e le consegne senza considerare se lo stesso è utile e con benefici per il cliente finale.

Dunque, l'approccio corretto, è quello di stabilire una relazione tra gestione del progetto, come causalità, e i suoi propri obbiettivi. La pre-analisi del contesto deve essere orientata al cercare di capire come la

gestione di un progetto secondo la metodologia migliore possa portare a un risultato di successo in linea con le aspettative del cliente finale. Lo svolgimento che segue vuole analizzare e chiarire le differenze esistenti tra le due metodologie ad oggi utilizzate nella gestione dei progetti: Waterfall e Agile. L'esame è condotto ricorrendo, come modello, a un caso di studio di un grande azienda di servizi IT, dove la gestione del progetto per soddisfare tutti i requisiti richiesti dal cliente è fondamentale per ridurre allo stesso tempo costi e Time to Market.

CAPITOLO PRIMO – IL CONTESTO

Il contesto di una grande realtà bancaria

La diffusione di internet e della connessione digitale ha rivoluzionato le società, il loro approccio al cliente cosi come la loro operatività interna. Spesso si parla di "era digitale", ma quello che le aziende stanno affrontando è una vera e propria rivoluzione digitale, con l'obiettivo per rimanere competitivi, tutelare i propri clienti e mantenere/accrescere le proprie quote di mercato. In un'analisi condotta tra i massimi esponenti di realtà bancarie è emersa la consapevolezza che il mondo cambierà radicalmente nei prossimi 3/4 anni, cambiamento mosso anche a causa di nuovi entranti che spingeranno questa rivoluzione. Google, Amazon, Apple, PayPal iniziano ad offrire nuovi sistemi di pagamento cosi come altri servizi bancari, le banche tradizionali dovranno dunque essere capaci di trasformare i propri modelli per recepire i cambiamenti e le richieste del mercato. La popolazione mondiale sta imparando sempre più velocemente a sfruttare le potenzialità dell'innovazione digitale perché comprende e preferisce in maniera sempre più evidente la possibilità di adottare strumenti più semplici, veloci e snelli. Ci sono fenomeni come quello dell'e-commerce che hanno scardinato in meno di 10 anni le vecchie logiche di acquisto richiedendo un adeguamento veloce e massivo della tecnologia di supporto come quella dei pagamenti digitali. Il problema più rilevante è che alcune tecnologie non sono più viste come innovative bensì come requisiti necessari, che il cliente si aspetta di trovare pena una alta delusione. In altre parole la sfida è quella certamente di offrire e trovare soluzioni innovative ma soprattutto quella di offrire servizi ormai largamente utilizzati che siano stabili, efficienti e funzionali. Fino a pochi anni fa recarsi fisicamente agli sportelli era considerato un qualcosa usuale, ma con l'avvento della digitalizzazione e dello sviluppo di specifiche applicazioni mobile le abitudini sono radicalmente cambiate, tanto che ci si sta movendo sempre più nel ridurre la presenza del cliente fisco in filiale agevolando strumenti remoti per la predisposizione di preventivi, apertura pratiche e richiesta prestiti. Il reale fattore di differenziazione, oltre alla sicurezza e solidità del istituto, è quanto si è veloci a soddisfare il desiderio del cliente. Il cliente ha sempre un desiderio, la disponibilità finanziaria ovvero il denaro è uno strumento. L'obbiettivo è rendere efficienti i processi che rendono lo strumento utilizzabile al momento necessario.

Da studi condotti sulla situazione del mercato italiano sono emersi due importanti punti di attenzione, il primo è che si assiste a una percentuale crescente di clienti multibancarizzati, secondo è che le condizioni economiche sono un importante fattore di scelta. Per il primo punto è evidente che sia necessario offrire al cliente un servizio che come detto possa permettere alla società di mantenere la sua quota di mercato, per il

secondo punto il principio è molto semplice: il cliente (tralasciando per un momento le questioni legate alla sicurezza e solidità dell'istituto bancario) sarà "contento" di pagare solo per un servizio efficiente e a valore aggiunto, la sua percezione quindi è fondamentale.

In questo contesto di sviluppo digitale è fondamentale il supporto proveniente dal mondo IT attraverso cui si è chiamati ad attivare una trasformazione digitale che comprenda tutta la filiera produttiva. I fornitori di servizi IT siano essi interni alla società o esterni dovranno considerare non solo gli elementi tecnologici all'interno dei componenti IT, ma anche considerare un adeguamento più ampio di carattere complessivo di come fare IT. Questo, in altre parole, significa reinterpretare il ruolo dell'IT all'interno della banca creando uno stretto legame tra business e IT: l'IT deve supportare strategicamente la vision del business.

Un gap che tuttavia persiste è rappresentato dal contenimento dei costi da un lato e dall'altro la necessità di investimenti per nuovi servizi verso il cliente. A tal proposito è facile definire uno schema riassuntivo che metta in relazione la propensione all'innovazione con la dimensione della banca.

		Propensione all'innovazione	
		Alta	Bassa
Dimensioni della Banca	Grande	Banca Leader	Lenta Disgregazione
	Piccola	Niche Player	Fallimento Istantaneo

Quello che in realtà viene richiesto è che ci sia una vista strabica; da un lato bisogna garantire il lavoro e la funzionalità giornaliere e dall'altro bisogna avere una vista di lungo periodo per comprendere come funzionano le nuove tecnologie e come esse possono essere implementate, in quanto il tempo che intercorre da quando si comprende che una nuova tecnologia serve e quando questa può essere effettivamente utilizzata, è ampio.

La trasformazione passa per i sistemi informativi, i quali introducono nuove tecnologie avanzate per la relazione con il cliente, cosi come prevedono una maggiore integrazione tra canali digitali e fisici, integrazione che offra all'utente un'esperienza coerente tra i diversi canali con cui si interfaccia. Questa integrazione non è solitamente tutta visibile all'utente finale in quanto nella parte del back end è richiesto che si raggiunga una flessibilità tale che permetta di adattarsi a prodotti e servizi in continuo cambiamento, invece, nella parte del front end si richiede una flessibilità tale

che abiliti i sistemi interni ad essere integrati con ecosistemi esterni in modo da trarre vantaggio da questa integrazione.

Il contesto competitivo di fatto mostra la presenza di imprese digitali che operano nel settore bancario con sistemi snelli e con Time-To-Market difficili da raggiungere dalle grandi istituzioni, per questo che l'innovazione architetturale deve essere orientata allo sviluppo di soluzioni digitali che tengano conto di esigenze di performance, flessibilità, compliance e sicurezza. Se da un lato la necessità di innovare è fattore fondamentale di sopravvivenza dall'altro un elemento importante è la capacità di captare e investire in soluzioni che siano di reale valore aggiunto. Nei cicli di investimento si prevede un orizzonte temporale a volte troppo lungo che non tiene conto del mutevole ecosistema, il rischio è quello di investire ingenti quantità di risorse in progetti pluriennali al fine dei quali il ritorno previsto è deludente a causa delle mutate necessità. Per questo motivo i punti chiave per lo sviluppo devono essere:

- Integrazione con servizi terzi, permette di utilizzare funzionalità già sviluppate accedendo a economie di scala e non necessitano di investimenti specifici per lo sviluppo ma per l'integrazione;
- Time-To-Market, come detto essenziale da tenere sotto controllo per evitare di arrivare sul mercato troppo tardi;
- Presidiare l'informazione proveniente dal cliente, ormai sempre più complessa e destrutturata, ma essenziale da analizzare per soddisfare le loro richieste;
- Ridurre i costi operativi, per essere contemporaneamente efficienti e capaci di recepire i cambiamenti;
- Ottimizzazione delle risorse, per focalizzarle non solo su attività di adeguamento normativo ma anche su attività a valore aggiunto.

Presente	**Futuro**
Filiali fisiche	Filiali Digitali
Documentazione Cartacea	Approccio Paperless
Focus su filiali e relazione	Focus su Software ed esperienza
Realtà complessa e lenta	Realtà snella e veloce
Forte intensità di capitale	Bassa intensità di capitale
Tecnologia proprietaria	Tecnologia di mercato
Regolamentazione come ostacolo (es. 3D secure)	Regolamentazione come abilitatore

Come visto le banche si collocano in uno scenario di profonda trasformazione che va gestita comprendendo le dinamiche del ambiente circostante focalizzandosi su temi come il digitale che non è più un'opzione ma un elemento indispensabile per la sostenibilità delle imprese. Con il digitale il cliente è al centro e bisogna focalizzarsi sul suo coinvolgimento; in altre parole bisogna focalizzarsi sul core business che va ridisegnato nella prospettiva di "cliente al centro" non sottovalutando il rischio di disintermediazione derivante dall'innovazione. I processi e servizi sono di diverso genere, quelli che coinvolgono il mondo non visibile al cliente è necessario che siano focalizzati sulla stabilità e efficienza, mentre quelli a stretto contatto del cliente è necessario che soddisfino le sue richieste. Gestire la trasformazione digitale con un modello integrato vuol dire diffondere e consolidare una chiara consapevolezza del cambiamento presso tutto il personale puntando sulla specializzazione, ma significa anche porre attenzione all'organizzazione e alla reattività all'innovazione.

Nel contesto bancario è inoltre necessario definire la differenza che esiste tra prodotto e servizio. In realtà la differenza presente è molto sottile: il prodotto è la funzionalità per cui il cliente paga a fronte del quale desidera ricevere qualcosa di concreto in cambio come il caso di prestiti, mutui, assicurazioni, conti corrente o depositi. Il servizio può essere inteso invece come tutta quella serie di attività il più delle volte nascoste al cliente grazie a cui il prodotto può funzionare.

Un esempio semplificativo può essere il caso delle carte di pagamento. In questo caso il prodotto è identificato dalla carta consegnata al cliente in linea con le sue esigenze mentre il servizio riguarda tutta la parte di processamento per cui una volta eseguita una transazione questa va a buon fine con l'addebito sul conto del cliente e l'accredito sul conto del venditore. Sebbene esiste una differenza tra prodotto e servizio, nella realtà bancaria è praticamente difficile avere un prodotto a cui non sia associato un servizio. È opportuno sottolineare questa configurazione di modo che sia facile comprendere che il servizio erogato deve essere sempre garantito; qualsiasi mal funzionalità ha potenziali impatti sul cliente finale e considerando la materia trattata le conseguenze reputazioni sono realmente imprevedibili.

Garantire la stabilità e la continuità del servizio è fondamentale anche in caso di grandi cambiamenti e nuove implementazioni come nel caso del cambiamento di un core banking system dove per limitare gli impatti si possono prevedere campagne pubblicitarie adeguate o interruzioni di servizio programmate in periodi temporali strategici in cui le attività dei clienti saranno prevedibilmente inferiori (ad esempio durante la notte o nei weekend). Il problema reale è che l'utente difficilmente accetta disservizi nell'ambito del proprio "patrimonio", oltre al fatto che alcune attività sono passate da *nice to have* a *crucial importance*. Fino a qualche anno fa il contante era il metodo di pagamento più utilizzato, oggi con l'utilizzo dell'e-commerce, la sicurezza del pagamento digitale e la sua stessa comodità

hanno stravolto le abitudini del utente rendendo indispensabile questa metodologia più innovativa. Questo è un ulteriore esempio di come il mercato abbia stravolto questo settore, e come tuttora stia richiedendo adattamenti continui e repentini.

Approcci di project management e metodologie di project management

Approccio di Project Manager (PM) può essere definito come una serie di principi guida che definisco come il progetto deve essere gestito. In connessione con questa definizione la metodologia di PM incorpora le serie di strumenti, regole, templates, metodi e best practices da usare all'interno del progetto. In questa ottica si può affermare che ad oggi esistono due principali approcci di PM uno tradizionale e l'altro Agile. La combinazione di questi approcci è spesso richiesto per adattarsi alle necessità progettuali, una volta trovata la combinazione prevista è possibile poi identificare le metodologie che assicurino la più alta customizzazione rispetto al progetto e all'ambiente.

A questo punto è tuttavia opportuno definire con maggiore precisione la differenza che esiste tra due termini spesso utilizzati come sinonimi: Project management approach e Project management methodology.

Project management methodology

Come già introdotto, ed anche definito dal Project Management Istitute, per metodologie si intende quel set di tecniche, procedure, strumenti usati nella conduzione del progetto. L'obbiettivo è quello di assicurare una maggiore probabilità di successo, attraverso una maggiore qualità e controllo, con una minore complessità.

Per il raggiungimento degli obbiettivi di successo è fondamentale passare attraverso la definizione rigorosa di una serie ti punti cardini:

- Templates e report da utilizzare;
- Piano standardizzato;
- Gestione del tempo e dei costi progettuali;
- Flessibilità negli sviluppi;
- Processi standardizzati durante il ciclo di vita del software.

Project management approaches

Esistono due tipi di approcci a cui si aggiungono tutte le varie declinazione degli stessi: approccio Tradizionale e approccio Agile.

Approccio Tradizionale

Un concetto alla base di un approccio tradizionale è che il progetto su cui verrà applicato sia relativamente semplice, prevedibile e distribuito linearmente nel tempo. L'obbiettivo è quello di ottimizzare ed efficientare seguendo un piano di progetto dettagliato, un budget e scope definito.

Uno delle peculiarità dell'approccio tradizionale è la sua robustezza derivante da una serie di regole e basi definite sin dall'inizio del progetto, ma allo stesso tempo questa rigidità che assicura la sua robustezza fa

emergere un difficile utilizzo in ambienti dinamici e molto complessi. Un ulteriore criticità dell'approccio tradizionale è sicuramente la poca connessione con l'ambiente circostante fino alle fasi finali del progetto. Come ultimo punto da evidenziare è la spesso difficile definizione di un piano di progetto completo e sufficientemente dettagliato a partire dal lancio dell'iniziativa.

Approccio Agile

Sebbene l'approccio Agile viene spesso chiamato in differenti modi le peculiarità rimangono le stesse e per lo più applicate a situazioni di sviluppo software.

Una delle caratteristiche universalmente attribuite a questo approccio è l'adattabilità ai cambiamenti che occorrono durante il ciclo di vita del prodotto. La possibilità di adattarsi in ambienti mutevoli, dove i cambiamenti sono inevitabili, è la caratteristica più importante anche rispetto alla prevedibilità.

Nel "Manifesto for Agile Software Development" sono stati definiti quattro valori fondamentali di questo approccio:

- Interazioni tra i processi e gli strumenti;
- Sviluppi basati su una documentazione completa;
- Collaborazione con il cliente finale;
- Adattamento ai cambiamenti.

Un termine che è stato scelto per riassumere questo approccio è l'agilità, definita come la possibilità di rispondere ai cambiamenti, creando valore aggiunto in un ambiente instabile e in fase di cambiamento. L'adattamento ai cambiamenti significa anche una continua innovazione, adeguamento del prodotto, veloci rilasci, aggiustamenti nei processi e risultati tangibili.

È possibile illustrare l'approccio Agile in 5 fasi:

Metodologia Agile	
Concepimento	Definizione della vision, dello scope e l'organizzazione del progetto
Ideazione	Sviluppo del modello progettuale definito dalle caratteristiche del prodotto
Esplorazione	Rilasciare parti sviluppate e testate nel breve periodo, ridurre il rischio e l'incertezza progettuale
Adattamento	Controllare i rilasci e l'ambiente attuale, attuare eventuali azioni adattive
Chiusura	Chiusura del progetto e creazione delle L&L (Lessons and Learned)

La possibilità di recepire i cambiamenti durante l'esecuzione del progetto coinvolge ciascuna iterazione che deve essere la più corta possibile in modo da costruire lo scope del progetto in modo dinamico da ciascuna iterazione, questo con l'obbiettivo da ridurre il rischio derivante da sezioni incerte che possono impattare sulla qualità finale.

È difficile affermare quale approccio sia preferibile se non si effettua un'analisi dettagliata di quale possono essere i vantaggi e svantaggi di ciascuna implementazione.

Per riassumere quanto già illustrato una metodologia tradizionale porta buoni risultati nel caso progetti con bassa incertezza e requisiti chiari e definiti quasi a priori, cosi come nel caso della presenza di routine ben conosciute e indirizzate al raggiungimento degli obbiettivi progettuali. In questi progetti ci si aspetta di ricevere pochi cambiamenti in corso d'opera e non è prevista/necessaria un forte coinvolgimento del cliente finale durante la fase progettuale. In questi casi viene data rilevanza al piano definito e alla possibilità di seguirlo in maniera rigorosa.

In parallelo un approccio Agile è indicato per progetti innovativi di sviluppo e ricerca dove ci si aspetta di osservare una maggiore incertezza durante la fase di sviluppo cosi come è impossibile in tali situazioni poter "blindare" lo scope e gli obbiettivi sin dalle fasi iniziali del progetto. A causa di questo continuo cambiamento il progetto è organizzato in via iterativa, non lineare con frequenti modifiche e impatti sul piano che richiedono un'intensa collaborazione e coinvolgimento dei team e dei clienti finali.

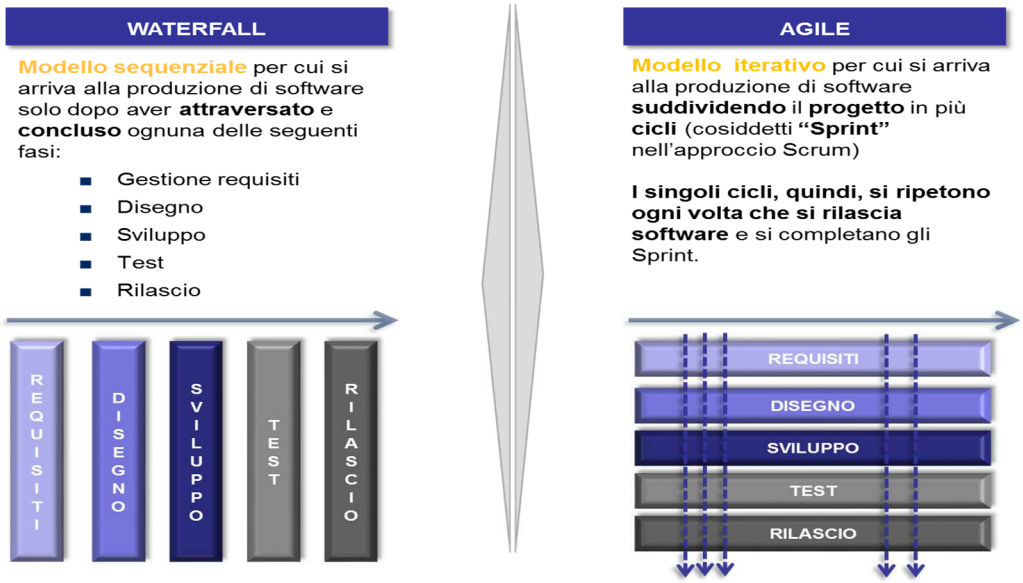
Come visto esiste, dunque, una connessione tra gli approcci e le metodologie. Esistono vantaggi e svantaggi per ciascun approccio. Nella selezione dell'approccio da adottare è necessario tenere in considerazione le caratteristiche del progetto e dell'organizzazione.

CAPITOLO SECONDO – LE METODOLOGIE

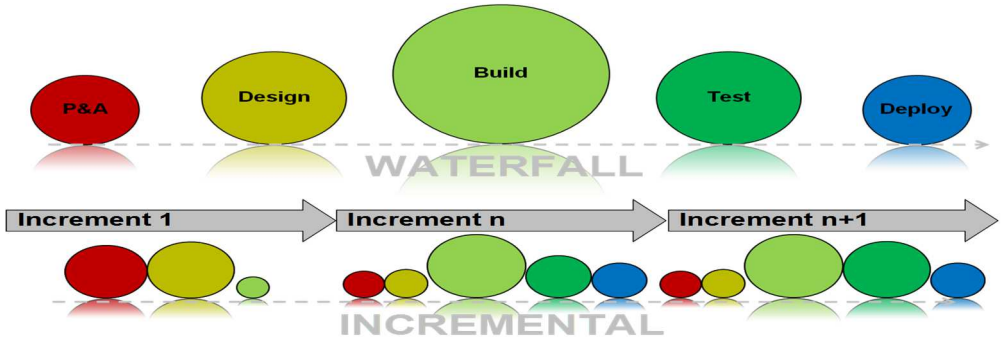
Metodologia Tradizionale – Agile

Come più volte affermato la necessità di strutturare un progetto secondo una pianificazione e una metodologia che supporti la sua gestione è necessaria per controllare tutte quelle funzioni e variabili che potrebbero impattare sul successo del progetto cosi come sui tempi, sulla soddisfazione e sui costi. Negli anni si sono affermate principalmente due differenti metodologie, quella tradizionale anche conosciuta come Waterfall e quella orientata allo sviluppo di sistemi e applicativi IT Agile. La prima segue quella che si può definire un approccio naturale ovvero la sequenzialità delle attività. Questa sequenzialità deriva da un concetto molto semplice, ovvero il prerequisito. In altre parole ogni attività è identificata come il prerequisito della successiva cosi che si arriva a definire la prima azione come quella fondamentale o ancora come quella che è da prerequisito al numero più alto di attività successive. In maniera similare ma con un diverso approccio la metodologia Agile può essere riconosciuta come un qualcosa che deriva da un'esperienza naturale e oggettiva per cui quando si affronta qualcosa di non pienamente conosciuto è meglio suddividerlo in parti piccole, svilupparle e analizzarle a fasi incrementali. La metodologia Agile pone attenzione agli individui più che ai processi e agli strumenti, al software funzionante più che alla documentazione esaustiva, alla collaborazione con il cliente più che alla negoziazione dei contratti, a rispondere al cambiamento più che seguire un piano. La metodologia Agile richiede disciplina, compromesso che passa attraverso la collaborazione tra i team e la trasparenza tra gli stessi. La disciplina è definita con l'attribuzione di responsabilità ben definite e attraverso la partecipazione a processi di comunicazione e strumenti che abilitano alla gestione Agile. In via di principio generale tutti i progetti IT possono essere gestiti sia in modalità Waterfall che Agile, la scelta della metodologia più adatta deve essere effettuata in base alle caratteristiche ed alle esigenze strategiche del progetto.

Il grafico sottostante mostra un primo confronto metodologico tra i due approcci.

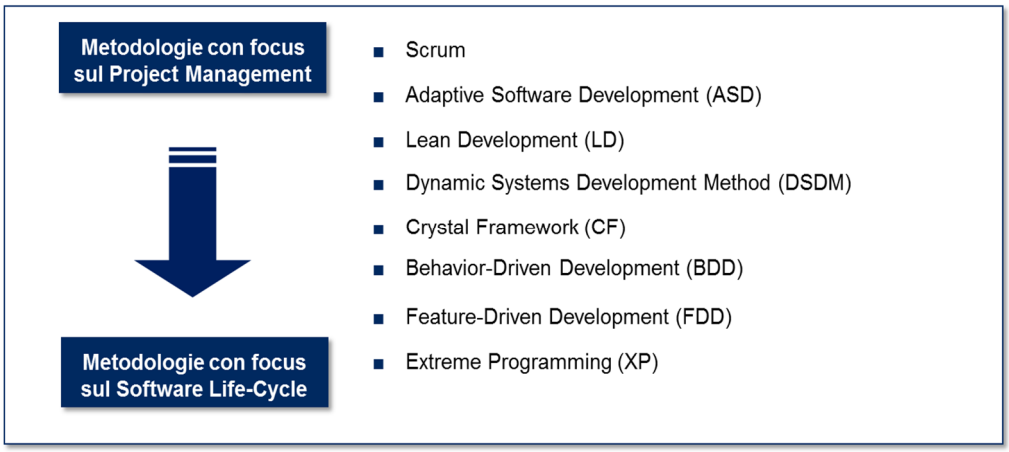


Agile è una sorta di ombrello che racchiude diverse metodologie, le quali modellano l'organizzazione iterativa del lavoro, in parallelo allo sviluppo progressivo dei requisiti.



La metodologia Agile più diffusa è quella Scrum perché più orientata al Project Management. Lo Scrum prevede un forte coinvolgimento dei referenti Business durante il corso del progetto, una collaborazione constante di tutto il team ed una forte automazione dei test.

Per effettuare un'analisi omnicomprensiva delle metodologie Agile che sarà utile a definire quale possa essere l'approccio che più si adatta alle caratteristiche di ogni singolo progetto, di seguito sono riportate alcune ulteriori metodologie largamente utilizzate che sono però tutte racchiuse nell'ombrello che si citava precedentemente, chiamato Agile. A partire dalla metodologia Scrum, come detto, la più orientata al Project Management si arriva alla Extreme Programming la più orientata al Software Life-Cycle.



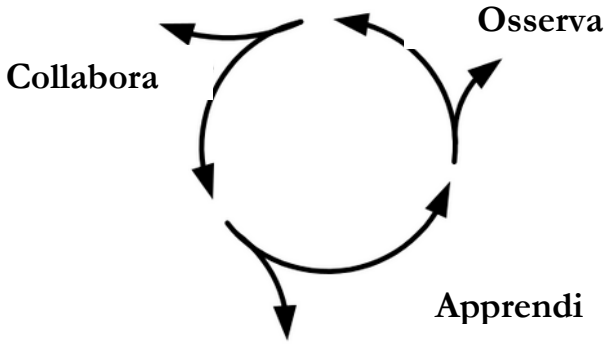
Scrum

Lo Scrum prevede la produzione di software suddividendo il progetto in più iterazioni chiamate Release che possono poi essere a volte suddivise in iterazioni più piccole chiamate Sprint.

In questa metodologia il Product Owner definisce la vision. Esso viene identificato all'inizio del progetto e recepisce con continuità le esigenze del business e collabora nella definizione delle priorità. Il progetto nella sua interezza viene suddiviso in una lista delle priorità chiamata Prioritized Product Backlog. Una volta che viene selezionato lo Sprint Backlog (scelto tra le attività presenti nel Product Backlog) questo entra nella fase di processamento dal team. Durante lo Sprint, che abbiamo ipotizzato durasse 20 giorni, il team lavora con una forte collaborazione allo sviluppo delle attività previste attraverso il coordinamento dello Scrum Master che con una serie di processi coordinativi come ad esempio, meeting giornalieri di allineamento, porta alla fine del ciclo un output definito che va ad aggiungersi incrementalmente all'avanzamento del progetto.

Adaptive Software Development (ASD)

Le pratiche di ASD sono guidate da un continuo adattamento, orientate ad accettare il cambiamento continuo. In ASD a un piano statico viene sostituito uno dinamico pensato per un apprendimento continuo, alla rivalutazione, all'analisi di un futuro incerto e all'intensa collaborazione tra sviluppatori, manager e clienti.



"Osservazione" dà la possibilità di esplorare, in una situazione di incertezza, la possibilità di deviare dai piani con consapevolezza. In altre parole questo significa avere iterazioni brevi. La possibilità di osservare, o come anche definito, di "*Speculate*", riconosce la natura incerta dei problemi complessi e incoraggia all'esplorazione e alla sperimentazione.

"Collaborazione" significa che applicazioni complesse non sono costruite ma evolvono. Applicazioni complesse richiedono che un grande volume di informazioni sia raccolto, analizzato e applicato al problema. La complessità risiede anche nella quantità di informazioni necessarie che evidentemente non possono derivare solo da un'unica persona, ecco che la capacità di collaborare (intesa come la capacità di lavorare in team per produrre risultati, condividere conoscenze e prendere decisioni) è di importanza primaria.

"Apprendimento" partendo dall'assunto che ciascuno non è infallibile e necessita di apprendere, il processo di *Learn* diventa un requisito essenziale per il successo. Testare le conoscenze costantemente, usare analisi retrospettive e confrontarsi con il cliente sono i principi cardini dell'apprendimento.

Lean Development (LD)

LD fonda le sue basi nei principi utilizzati nelle industrie manifatturiere e successivamente applicati in contesti di sviluppo software. I principi sono in generale i medesimi:

- Eliminare gli sprechi: significa concentrarsi su ciò che crea valore aggiunto per il cliente. Questo principio si articola in due step, il primo passo è imparare a vedere gli *sprechi*, il secondo è capire la fonte degli sprechi ed eliminarle. Un esempio di sprechi possono essere: sviluppi parzialmente eseguiti, eccesso di documentazione, sviluppo di componenti errati, bug, caratteristiche addizionali richieste.

- Apprendimento: il processo di apprendimento passa attraverso una buona sincronizzazione. Sincronizzazione e integrazione quotidiana all'interno del team, sincronizzazione settimanale tra i diversi team.
- Decidere il più tardi possibile: non significa procrastinare la decisione, ma tenerla aperta il più lungo possibile considerando però anche i termini di efficienza. Di fatto spesso si resiste nel prendere decisioni irrevocabili in situazioni di incertezza. Più opzioni consentono decisioni basate sulla realtà e non sulla speculazione.
- Rilasciare il più presto possibile: bisogna porre attenzione nel non rilasciare al cliente un prodotto di scarsa qualità bensì, questo principio vuole sottolineare la necessità di consegnare ai clienti ciò che richiedono, quando lo fanno. I clienti vogliono avere consegne rapide, che spesso si traducono in maggiore flessibilità aziendale. In altre parole questa velocità viene misurata come la rapidità con cui si converte in modo affidabile e ripetuto le richieste del cliente in un software.
- Potere alla squadra: non significa abbandonare la leadership ma permettere alle persone che aggiungono valore di usare tutto il loro potenziale. La motivazione del team è un fattore essenziale e non passa attraverso la sua misurazione bensì lasciando "potere" al team. Questo concetto si concretizza spostando le decisioni al livello più basso possibile in un'organizzazione mentre si sviluppa la capacità del team di prendere decisioni con consapevolezza.
- Integrità: investire sull'integrità progettuale vuol dire da un lato raggiungere un equilibrio di funzionalità, usabilità, affidabilità e economicità in linea con le attese del cliente e dall'altro costruire un qualcosa che funzioni in maniera coesa nella sua totalità. L'unico modo di creare questa coesione è utilizzando flussi di informazione eccellenti sia tra cliente e team sia tra i vari processi.
- Vedere la totalità: non significa ignorare i dettagli ma porre attenzione alla tentazione di ottimizzare le singole parti a scapito della totalità. Il pensiero *Lean* suggerisce che l'ottimizzazione delle singole parti porta a un sistema generale sub ottimizzato. Più il sistema è complesso e più alta è la tentazione di suddividerlo in parti gestendole singolarmente. Questa suddivisione spesso porta a misure e analisi relative alle singole parti che creano a livello di sistema effetti che riducono le prestazioni complessive.

La produzione in ottica Lean è buona per lo sviluppo software se vengono applicati i principi che sono alla sua base: eliminazione degli sprechi, responsabilizzazione delle persone, risposta immediata alle richieste del cliente e ottimizzazione attraverso la catena del valore.

Dynamic Systems Development (DSDM)

Come suggerito dal nome, questo è un approccio dinamico allo sviluppo di sistemi e sfrutta l'uso della prototipazione incrementale con il coinvolgimento continuo dei clienti. L'obbiettivo è quello di rilasciare lo sviluppo in linea con tempi e budget, mentre cambiamenti dei requisiti sono gestiti in parallelo. Questa metodologia è particolarmente utilizzata nei casi di sistemi da sviluppare in poco tempo e dove i requisiti non possono essere definiti sin dall'inizio dello sviluppo software. La metodologia DSDM ha cinque fasi nel ciclo di vita dell'implementazione:

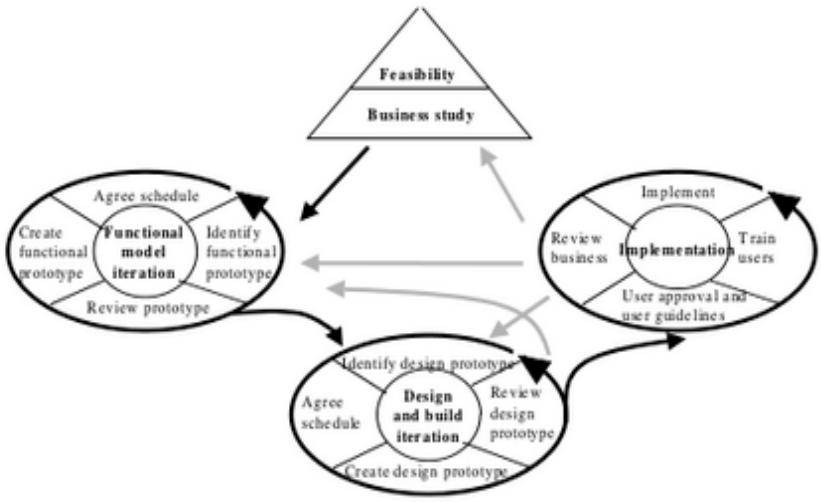
- Feasibility e Business Study: in questa fase viene definito il contesto e verificata la fattibilità tecnica. In questa fase vengono quindi definiti i requisiti ad un alto livello e l'architettura di base del sistema.
- Functional Model Iteration: l'obbiettivo in questa fase è lo sviluppo iterativo del prototipo e l'ottenimento della revisione da parte degli utenti che permette di fare emergere i requisiti desiderati. Il prototipo è migliorato attraverso le dimostrazioni all'utente, prendendo i loro feedback e incorporando le modifiche. Questo processo viene generalmente ripetuto più volte fino al raggiungimento di un modello funzionante accettato dall'utente. Dunque il prodotto finale di questa fase è un modello con componenti software sviluppati che contengono le principali funzionalità.
- Design e Build Iteration: in questa fase ci si assicura che i prototipi siano soddisfacenti e opportunamente progettati per adattarsi al ambiente operativo. I modelli progettati nella fase precedente vengono ulteriormente raffinati fino a raggiungere uno standard soddisfacente. Il prodotto di questa fase è un sistema testato pronto per l'implementazione, tuttavia non essendoci un confine netto tra questa fase e la precedente, spesso le due fasi continuano parallelamente.
- Implementation: questa è l'ultima fase di sviluppo prevista da questa metodologia. In questa fase gli utenti vengono istruiti e il sistema viene effettivamente rilasciato nell'ambiente di produzione. Alla fine di questa fase ci sono quattro possibilità:
    - Tutto è stato rilasciato secondo le richieste dell'utente quindi non sono rischisti ulteriore sviluppi;
    - È stata scoperta una nuova area funzionale, quindi è necessario tornare alla fase di analisi/fattibilità e ripetere l'intero processo;
    - Una parte non essenziale del progetto non è stata sviluppata a causa di limiti di tempo, quindi lo sviluppo torna alla fase 2 (Functional Model Iteration);

- Alcuni requisiti non funzionali non sono stati soddisfatti, lo sviluppo, quindi, torna alla fase 3 di progettazione (Design e Build Iteration).

DSDM è una metodologia che dà priorità alla pianificazione e alla qualità rispetto alle funzionalità. Questa metodologia utilizza il metodo di prioritizzazione MoSCoW, che suddivide un progetto in quattro diversi tipi di requisiti:

- Must have (Mo);
- Should have (S);
- Could have (Co);
- Won't have (W);

Questa suddivisione permette in definitiva di concentrare gli sforzi in base ai requisiti e ai limiti di tempo imposti, ed eventualmente investire ulteriore tempo di sviluppo per parti *"nice to have"*.



Crystal Framework (CF)

I metodi *Crystal* sono focalizzati su principi come l'interazione, la comunicazione, le persone e i talenti. Nella formalizzazione iniziale l'autore di questo metodo considerava il processo di secondaria importanza rispetto ai principi prima elencati. Il concetto di base è che i team coinvolti nello sviluppo software hanno abilità e talenti diversi, più importanti rispetto al processo. Il modello *Crystal* ha sviluppato una famiglia di metodologie e il loro peso è identificato da un colore. Per un progetto piccolo si potrebbe

dare un colore *Crystal Clear*, mentre per progetti dal peso molto più rilevante si potrebbe assegnare una metodologia *Crystal Sapphire:*

- Crystal Clear
- Crystal Yellow
- Crystal Orange
- Crystal Orange Web
- Crystal Red
- Crystal Maroon
- Crystal Diamond
- Crystal Sapphire

Per le metodologie sopra elencate ci sono sette proprietà comuni:

- Consegne frequenti: è il rilascio regolare delle iterazioni del programma software. Gli sviluppatori decidono quali funzionalità includere in ogni versione. Con i metodi Crystal, le iterazioni devono essere rilasciate settimanalmente fino a un trimestre - i tempi di rilascio dipendono dalla lunghezza del progetto.
- Miglioramento riflessivo: Il miglioramento riflessivo coinvolge gli sviluppatori che dallo sviluppo regolare, cercano di trovare modi per migliorare i loro processi. Le iterazioni aiutano a fornire un feedback se il processo attuale funzioni o no.
- Comunicazione ravvicinata: la comunicazione ravvicinata implica che la squadra sia localizzata nel medesimo spazio e che le informazioni scorrano attorno ad esso. Il team deve essere nella stessa stanza perché funzioni. Questo perché se lo sviluppatore deve spostarsi da un'altra parte per fare una domanda, ci potrebbe essere una perdita di informazioni.
- Sicurezza personale: la sicurezza in questo contesto ha a che fare con il problema della libertà di parola all'interno di un gruppo di persone. Ogni persona viene valorizzata ogni volta che fa una domanda, suggerisce un'idea, o solleva un problema. Le persone del team devono potersi fidare l'una dell'altra e sentirsi libere di parlare dei problemi.
- Messa a fuoco: in primo luogo significa concentrarsi sul singolo compito per un tempo sufficiente per avere risultati tangibili e in secondo luogo, si riferisce a un focus per cui si analizza la direzione del progetto.
- Accesso a utenti esperti: questo coinvolge gli sviluppatori che lavorano con una persona esperta del progetto in modo che l'esperto risponda a qualsiasi domanda o suggerisca soluzioni ai problemi. L'utente esperto deve essere un utente reale e non solo un tester del team di sviluppo.

- Test automatici, gestione della configurazione e integrazione frequente: l'idea alla base è che dovrebbero esserci continue integrazioni e test in modo che, se vengono apportate modifiche, si possano individuare eventuali errori.

Behaviour-Driven Development (BDD)

Il BDD è una metodologia che combina lo sviluppo del software basato sui test (TDD) e sul design basato sul dominio (DDD - il domino è inteso come una serie di requisiti). Lo sviluppo basato sui test focalizza lo sviluppo su brevi iterazioni cicliche in cui inizialmente vengono creati test che definiscono la funzionalità desiderata. Se eseguita in modo coerente e rapido, questa logica orientata ai test può ridurre drasticamente i tempi di sviluppo complessivo e ridurre gli errori segnalati, poiché il codice sviluppato tende a essere più conforme e stabile durante tutto il ciclo di vita dello sviluppo del software. Una progettazione orientata al dominio si incentra sul concetto di dominio e logica di dominio, che comprende la "sfera di conoscenza o attività attorno alla quale ruota la logica dell'applicazione". Le pratiche DDD tentano di semplificare la terminologia nell'ambito del progetto focalizzando e definendo tutto come oggetti o concetti del mondo reale a cui la maggior parte delle persone ha familiarità. Ciò semplifica in gran parte la comunicazione e incoraggia il team a sviluppare un'applicazione che soddisfi esattamente le esigenze del particolare dominio in questione. Combinando parti di TDD e DDD, lo sviluppo orientato al *Behaviour* mira a semplificare lo sviluppo attraverso l'uso di un linguaggio specifico di dominio comune, che viene utilizzato per adattare frasi in un linguaggio naturale a test eseguibili. Questa pratica di usare frasi in linguaggio naturale per descrivere e definire i test viene utilizzata in numerosi linguaggi di programmazione e suite di test. Mentre il TDD "libero" consente ai test di concentrarsi su tutti i livelli di requisiti all'interno dell'applicazione, BDD afferma che i test dovrebbero essere definiti in termini di "comportamento desiderato dell'unità".

In genere, la definizione dei comportamenti all'interno di BDD viene realizzata attraverso le *User Stories*. Questi sono scenari scritti che includono una sorta di testo che riassume l'intento, oltra a una sezione narrativa che descrive chi e cosa dovrebbe essere coinvolto nel raggiungimento degli obiettivi. Il BDD non applica alcuna sintassi o formato particolare per queste *User Stories*, ma suggerisce che ciascun team standardizzi un formato a cui attenersi. Ciò garantirà che il team possa discutere e modificare facilmente le *User Stories*, e più membri del team possono creare *User Stories* senza la necessità di lavorare a stretto contatto. Poiché lo sviluppo basato sul comportamento è fortemente influenzato dallo sviluppo basato sui test,

molti degli stessi vantaggi che si applicano al TDD si applicano anche al BDD:

- Riduce i problemi di regressione: con una serie completa di test eseguiti continuamente e con l'aggiunta di nuovi test, il BDD riduce drasticamente la probabilità che i bug di regressione spuntino, poiché il codice si troverà in uno stato costante di monitoraggio e test.
- Migliora la comunicazione tra team: la dipendenza da una lingua ben definita implica che il BDD spesso migliora la comunicazione attraverso l'intero team o anche tra le organizzazioni, poiché esiste una base comune e reale per la comunicazione e la terminologia quando si discute del progetto.

Alcuni svantaggi possono essere:

- Richiede le specifiche prima dello sviluppo: lo sviluppo richiede che il team scriva la documentazione sulle specifiche derivanti dalle *User Stories* per ogni particolare scenario o caratteristica, prima della scrittura del codice.
- Si affida a feedback esterni costanti: pur rimanendo in contatto con utenti, clienti o esperti potrebbe non essere un problema per alcuni team, ma per molte organizzazioni questo requisito di contatto costante con persone esterne può avere un impatto negativo sui tempi di sviluppo. Nei casi in cui è necessario un feedback per arricchire una nuova *User Stories* o scenario prima che i test vengano scritti se l'esperto non è disponibile in quel momento, lo sviluppo può fermarsi.

Feature-Driven Development (FDD)

Lo sviluppo guidato dalle funzionalità è incentrato sul cliente. Le funzionalità con questa metodologia sono aggiunte in modo incrementale al prodotto finale e il prototipo funzionante è presentato al cliente per una sua prima valutazione.

Ci sono cinque processi:

1. Sviluppare un modello generale;
2. Creare un elenco di funzionalità;
3. Pianificare le funzionalità;
4. Design delle funzionalità;
5. Creazione delle funzionalità.

L'idea principale di FDD è la creazione di un prototipo funzionante da presentare al cliente, scomponendo funzionalità complesse in parti più semplici in modo da presentare in maniera frequente risultati al cliente. Il team di progetto lavora collaborando per trovare le migliori opzioni per lo sviluppo delle funzionalità, in modo da ridurre i rischi derivanti dall'azione solitaria. In definitiva è necessario che ci siano frequenti ispezioni in maniera tale che i problemi o le discrepanze rispetto al desiderio del cliente vengano identificate tempestivamente.

Extreme Programming (XP)

XP si basa su tecniche di programmazione, comunicazione e lavoro in team. Ha in comune con altre metodologia il principio di sviluppare solo parti che aggiungono valore al cliente ma ha come vantaggio di poter adattarsi in condizioni in cui i requisiti possono ricevere rapide evoluzioni, oltre che a funzionare bene con diversi dimensioni di team. XP affronta i rischi a tutti i livelli di sviluppo, dagli slittamenti ai bug, dai cambiamenti di requisiti dal Business alla rotazione del personale. I cinque valori chiave della metodologia XP sono.

- Comunicazione: la comunicazione è fondamentale e alla base della riduzione dei problemi. Vengono implementate una serie di pratiche che non possono essere condotte senza la comunicazione e viene inoltre impiegato un referente il cui compito è quello di agevolare e abilitare la comunicazione.
- Semplicità: questo principio è fondamentale per non entrare in problemi derivanti da artefatti complessi, la comunicazione può favorire il confronto per trovare la soluzione migliore.
- Feedback: i programmatori ricevono un feedback grazie agli unit tests, i clienti scrivono nuovi requisiti e i programmatori rispondo immediatamente con il loro feedback circa la qualità dei dettagli, il cliente rivede le parti rilasciate e fornisce un immediato feedback allo sviluppatore.
- Coraggio: il team è incoraggiato a fare cambiamenti anche nell'architettura se ci sono difetti rilevanti. Il team tuttavia è chiamato sempre ad analizzare le potenziali conseguenze.
- Rispetto: rispetto delle regole, del progetto e del team di lavoro.

Differenza tra le metodologie nella gestione dei team

La sfida che tutte le società devono affrontare è il veloce cambiamento e adattamento dell'ambiente esterno in modo da rimanere competitivi e continuare la loro crescita.

Il metodo tradizionale è molto solido ma recepisce con difficoltà cambiamenti a differenza dell'approccio agile il quale funziona molto bene in relazione alle trasformazioni, il punto è capire se tale metodo ha gli stessi benefici su progetti con differenti caratteristiche e dimensioni di team.

Background

Per poter mantenere il controllo su progetti grandi spesso ci si avvale di una struttura organizzativa più complessa, con più livelli decisionali e con il coinvolgimento di maggiori responsabili progettuali.

In quest'ottica un approccio Agile prevede il coinvolgimento iterativo di tutti i team partecipanti a differenza dell'approccio tradizionale che prevede specifici momenti di interazione in fasi prestabilite del progetto. L'approccio Agile, deve essere orientato in un approccio il più *"flat"* possibile, con livelli manageriali essenziali e con un forte coinvolgimento dei team. L'obbiettivo del coinvolgimento dei team è avere una larga interconnessione tra i gruppi di lavoro, in cui le persone chiave sono sempre coinvolte negli incontri di allineamento in modo da intensificare la collaborazione evitando complicazioni grazie al coordinamento.

Distribuzione del team e processo di implementazione

Nella conformazione attuale delle organizzazioni è ormai diventata esperienza usuale l'interazione di diversi team distanti fisicamente; la gestione della distribuzione del lavoro sui diversi team è un qualcosa che deve essere correttamente guidato dal processo di implementazione progettuale scelto.

Distribuzione del team usando una metodologia tradizionale

Un chiaro vantaggio di una metodologia Waterfall è una facile gestione di team distribuiti perché si basa su una struttura chiara per il controllo e l'organizzazione delle attività. Questa facilità deriva anche dalle caratteristiche organizzative del lavoro intrinseche nell'approccio, in altre parole la definizione rigida di ogni fase che parte dall'identificazione dei requisiti e continua fino agli sviluppi e ai rilasci al cliente. In questo contesto la distribuzione in diversi luoghi del team non impatta sui rilasci finali perché a ogni team viene assegnato uno specifico set di informazioni e documenti da cui ci si aspetta uno specifico output, che verrà integrato solo successivamente. La possibilità di dividere il lavoro sui diversi team riduce

il rischio dall'assenza di una stretta correlazione e comunicazione resa più difficoltosa dalla distanza fisica.

Distribuzione del team usando la metodologia Agile

Sebbene i primi utilizzi di questa metodologia sono stati applicati a progetti piccoli con il coinvolgimento di team concentrati nella stessa area di lavoro, ora con l'espansione della metodologia Agile, il suo utilizzo in progetti molto più grandi ha richiesto inevitabilmente il coinvolgimento di diversi team e dislocati in diverse sedi di lavoro.

L'obbiettivo rimane tuttavia sempre il medesimo, ovvero incontrare il soddisfacimento del cliente, fornendogli tutti i requisiti richiesti con la qualità desiderata. La sfida è la gestione e la suddivisione dei compiti tra tutti i team coinvolti con un'ottica Agile.

Nella gestione dell'organizzazione del progetto in ottica Agile, ci sono 4 fattori da poter prendere in considerazione:

- Struttura organizzativa: in team molto ampi si potrebbe ricondurre alla segmentazione degli stessi per competenze (architetti, PM, responsabili della qualità) oltre a prevedere un team guida che costituito da diverse figure al suo interno (esperti, PMs, Product Manager) ha il compito di indirizzare il progetto nella sua gestione cosi come nel coordinamento della comunicazione, in modo da facilitare il processo decisionale.
- Processo Decisionale: questo processo deve coinvolgere la rappresentanza di tutti i team che a diverso titolo verranno coinvolti per la loro area di interesse.
- Collaborazione e coordinamento: prevedere una struttura organizzata di comunicazione con canali prestabiliti per la condivisione delle informazioni.
- Cultura Agile: tra gli aspetti più critici che a volte si tende a sottovalutare è che non basta decidere di adottare un approccio Agile. Il punto critico da smarcare è la condivisione dell'approccio Agile, come strumento accettato e conosciuto da tutti i membri dei team del progetto. Questo è l'unico modo per poter applicare in maniera efficace tutti quegli strumenti previsti dall'Agile.

Nell'adozione della metodologia Agile, con il coinvolgimento di più team è necessario apportare delle modifiche agli strumenti solitamente utilizzati:

- Backlog: si può optare nell'individuazione di un unico backlog condiviso tra tutti i team oppure più backlog per ciascun team. La prima opzione è utile alla trasparenza e a rendere consapevoli tutti i team delle lavorazioni in corso ma se a un livello troppo alto

potrebbe distogliere l'attenzione di ciascun team dal proprio pacchetto di competenza.

- Incontri: un rischio che spesso deve essere mitigato è la tendenza ad avere incontri di allineamento multipli dei team coinvolti, questo oltre a generare poco valore aggiunto spreca tempo in iterazioni non necessarie. In quest'ottica ebbene condurre incontri operativi a livello di team (stand-up meeting, *retrospective meetings*) ma sarà poi compito dello Scrum Master indirizzare e coinvolgere il proprio team nel coordinamento con altri al momento del bisogno.
- Infrastruttura: l'infrastruttura di sviluppo a cui tutti i team progettuali accederanno dovrà essere pronta a sostenere questa interazione con strumenti di sharing, test e controllo (ad esempio come i controlli e la tracciatura delle modifiche apportate).

Altre Caratteristiche e Differenze

Un processo software è un insieme di attività e risultati funzionali alla realizzazione di un prodotto software eseguibile. Le attività possono variare da un processo a un altro ma quattro sono fondamentali e utilizzate da tutte le metodologie:

- Definizione delle specifiche software, ovvero le funzionalità e vincoli operativi che devono essere osservati;
- Sviluppo software, ovvero il processo di implementazione per soddisfare le specifiche definite;
- Convalida del software, ovvero quel processo utile a garantire che il prodotto sviluppato sia in linea con le richieste del cliente;
- Evoluzione del software, ovvero la previsione che il prodotto possa soddisfare future evoluzioni in linea con nuovi bisogni;

Un progetto IT si riferisce a quelle attività associate allo sviluppo software per uno specifico impiego o scopo. I passi e le procedure seguite dalle società IT per lo sviluppo software seguono processi customizzati internamente o procedure più standard. Esiste di fatto un quesito aperto nell'applicazione di una determinata metodologia, questo quesito viene posto ogni qualvolta che il processo di sviluppo deve iniziare. Il dibattito vede come possibili scelte un approccio prescrittivo e quello descrittivo. Le metodologie più tradizionali sostengono un approccio più prescrittivo in cui i passaggi del processo sono definite in anticipo e in dettaglio cosi come gli obiettivi per il progetto sono abbastanza stabili durante l'intero progetto. Dall'altra parte c'è il relativamente più innovativo approccio Agile che sostiene un approccio descrittivo in cui i passaggi e gli obiettivi vengono determinati in modo dinamico. La decisione e quindi la scelta deriva spesso da esperienze derivanti da sviluppi precedenti ed esperienze simili.

Un passo preliminare nel processo software è di determinare una serie ordinata di passi da seguire per lo sviluppo nonché i criteri per passare da uno step al suo successivo fino a raggiungere quello finale che consisterà nella conclusione del progetto e consegna del prodotto al cliente. Il ciclo di vita del prodotto è solo un'estrapolazione dell'intero processo previsto, la sua unica funzione è quella di rappresentare il processo da utilizzare.

Lo sviluppo con un coinvolgimento *globale* è diventato una pratica comune nelle società moderne, superando barriere introdotte dalla distanza e dalle differenze culturali, concentrandosi sulle capacità individuali e sulla soluzione migliore in termini costi/efficienza. Il processo e la metodologia utilizzata deve quindi essere analizzata in questo tipo di contesto diventato ormai quasi più comune di situazioni di sviluppo software completamente locale. Il processo agile richiede una comunicazione con il cliente attraverso tutte le fasi cosi some un rapporto di scambio personale (face-to-face) come approccio preferito per comunicare. Al contrario l'approccio tradizionale

pone enfasi nell'importanza di mantenere una documentazione accurata. Vi sono una serie di confronti che sono utili a comprendere quali possono essere elementi importanti nell'approccio di sviluppo scelto.

Costi sviluppi in loco o off-shore

Solitamente gli sviluppi che prevedono che il team lavori con il cliente, vicino in termini di distanze logistiche, offrono una più semplice collaborazione e coordinamento con il cliente ma allo stesso tempo può comportare costi maggiori (in base al costo del lavoro). Di fatto non esiste una correlazione diretta con questi costi tanto che tale confronto non ha particolare rilevanza nella scelta del team di sviluppo.

Progetto di successo

La maggior parte dei problemi e delle difficoltà nella conduzione di un progetto IT derivano dalle competenze specifiche degli individui, dalla comunicazione e dall'interazione tra i team. Nell'ambito del successo del progetto, società che sono abituate al frequente cambiamento del cliente, o al lavoro in serie per brevi periodi rispetto allo sviluppo incrementale a lungo termine di un singolo prodotto incontrano meno problemi. È utile tuttavia sottolineare che cosi come aziende abituate ad interfacciarsi con clienti diversi migliorano il loro processo e la gestione del progetto, società focalizzate su pochi sviluppi gli permette di accumulare importanti conoscenze. Questo suggerisce che la flessibilità è un'abilità centrale per lo sviluppo software necessario per reagire e minimizzare gli effetti di problemi che possono sorgere.

Qualità del Team

La qualità del team e delle sue *skills* è da ritenersi il fattore principale di successo. Il team è l'attuatore delle richieste derivanti dal cliente, la capacità di recepire in maniera veloce e soprattutto corretta le specifiche permette di sviluppare meglio e potenzialmente richiede minori cambi dovuti a incomprensioni delle specifiche. Il team ha potenzialmente un importante leva che può impattare sulla vita del software. Uno sviluppo fatto a regola d'arte cerca di prevedere e prevenire eventuali problemi che possono sorgere nel corso della vita del software cosi come può prevedere e pianificare come cambiamenti resi necessari da mutate necessità vengano implementati nel modo più efficiente. Elemento che più volte verrà sottolineato e che anche in questo caso ha una rilevanza importante è la comunicazione e la collaborazione che può potenzialmente deteriorare le qualità del prodotto se non viene applicata in maniera corretta.

Modello di comunicazione

Una comunicazione efficace è un fattore di successo del progetto. I mezzi di comunicazione impiegati possono essere catalogati rispetto alla sincronia e all'efficacia. La maggior parte della comunicazione tra gli

sviluppatori avviene utilizzando messagistica istantanea, preferita rispetto a chiamate o incontri specifici, in quanto permette un buon compromesso tra comunicazione in tempo reale e non.

Fluttuazioni del personale

Le fluttuazioni si riferiscono al fenomeno di cambiamento delle risorse all'interno del team. Le fluttuazioni spesso producono una perdita di Know-How, ritardi e aumenti di costi. Queste fluttuazioni spesso sono da ricondursi a una bassa motivazione del personale che per questo decide di lasciare un team.

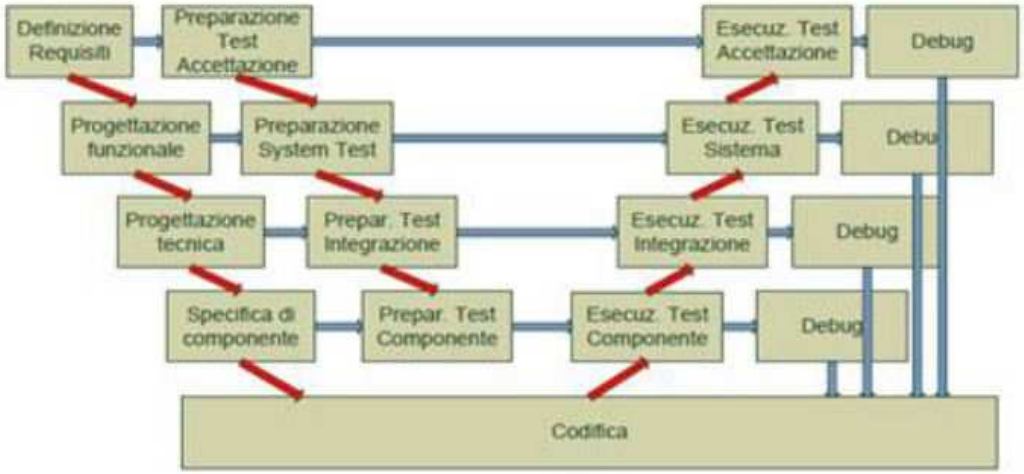
Di seguito alla luce dei feedback proposti finora, sono proposte sette peculiarità dell'approccio Agile analizzate su tre dimensioni: persone, processo e prodotto.

- Rilasci frequenti e continua integrazione: sviluppare in maniera iterativa permette di gestire la crescente complessità in ciascuna iterazione. Affinare frequentemente la metodologia permette un migliore adattamento della stessa rispetto alle necessità. Rivalutare la configurazione del team frequentemente significa adattare la composizione dello stesso in dimensioni e skills in base alle varie fasi di sviluppo.
- Feedback frequenti: significa testare e validare in modo iterativo implementando eventuali miglioramenti. Per quanto riguarda il processo si prevedono milestone per valutare il processo e l'efficienza dello stesso. Dal punto di vista del personale significa coinvolgere in maniera pragmatica le parti interessate e gli sviluppatori in maniera da raccogliere i commenti e eventuali preoccupazioni.
- Gestione proattiva delle modifiche ai requisiti progettuali: la soluzione deve essere sviluppata in modo da essere reattiva ai cambiamenti emergenti del progetto. Il processo deve prevedere una metodologia che permetta che questi bisogni emergenti vengano identificati il più presto possibile. La configurazione del team deve riflettere questa necessità in modo che partecipi all'identificazione degli stessi.
- Ambiente di sviluppo controllato (liberamente): adoperarsi nella realizzazione del software per componenti utilizzando un processo di sviluppo modulare. La responsabilità deve essere delegata il più possibile all'unità di sviluppo locale.
- Pianificazione ridotta al minimo: pianificazione minima del prodotto che deve essere fatta nel momento in cui si rende necessaria, limitata è l'enfasi alla documentazione in quanto ridotta all'essenziale. La soluzione deve emergere "naturalmente" piuttosto che da una pianificazione. Il team deve essere istruito affinché la soluzione emerga.

- Agevolare l'apprendimento e il miglioramento continuo: il processo deve prevedere la sperimentazione attraverso cui si apprendono gli errori fatti. Il risultato derivante dall'apprendimento di soluzioni passate ed errori produce un prodotto che oltre ad essere già stato testato, è maggiormente immune a errori nella fase di sviluppo. Il team è abilitato a sperimentare e a commettere errori, il lavoro in team è fondamentale per l'identificazione delle mancanze e per la definizione delle soluzioni.
- Prodotto funzionante: obbiettivo è promuovere il processo che sia rapido nel rilascio di una soluzione software funzionate, per agevolare tale reattività è opportuno lasciare al team la sua organizzazione evitando quando non necessario un non efficiente controllo centralizzato.

Fase di test

Un'altra differenza riguarda l'ambito dei test. Il testing non è un'attività indipendente rispetto alle attività di sviluppo e viene classificato anche esso in base al modello di sviluppo scelto sequenziale o iterativo. Le attività di test nel modello sequenziale o tradizionale coprono ogni fase di sviluppo con una specifica attività (preparazione System Test, Unit Test, esecuzione dei Test di Integrazione). Il processo assume una configurazione a W. La configurazione a W mostra evidentemente come i test vengono si, pianificati sin da subito ma la loro esecuzione avviene in fasi più avanzate, successive alla codifica. Una delle problematiche che si può verificare con l'approccio tradizionale è che se un componente a monte presenta dei problemi (mal funzionamenti, bug, errori nel codice) gli stessi possono ripercuotersi su tutti i componenti a valle con effetti di amplificazione potenzialmente molto onerosi in termini di impatti e tempi di risoluzione.



Differentemente, come più volte ripetuto, nell'approccio incrementale lo sviluppo e il test vengono affrontati in modo iterativo e sono basati su

un insieme crescente di requisiti e sul loro raffinamento, in questo modo si riduce il rischio di produrre il prodotto sbagliato in quanto viene validato dal cliente fin dai primi cicli di sviluppo. I test e le verifiche dell'utente vengono condotte all'interno della release lungo tutti gli sprint e a conclusione della release vengono condotti specifici test per la validazione da parte dell'utente. La verifica dei test avviene nel corso degli sviluppi in quanto si ha lo stretto contatto con l'utente (con autonomia decisionale che permette allo stesso di prendere decisioni), il test è ricorsivo cosi come lo sviluppo ma la differenza con l'approccio tradizionale è che i test vengono definiti durante la stesura dei requisiti.

Adozione della metodologia

L'adozione dell'approccio Agile deve essere condiviso in maniera trasversale in tutte le funzioni aziendali coinvolte, perché la cooperazione è un elemento fondamentale ed è quindi necessario che tutti i team coinvolti siano a conoscenza e condividano le regole in modo da avere un unico strumento a cui tutti si attengono. L'analisi su quale metodologia adottare sarà trattato più avanti, tuttavia le domande preliminari che bisogna porsi sono razionalmente riguardanti quale metodologia è più adatta alla gestione progettuale, se è possibile adottare un approccio Agile e quali sono le raccomandazioni pratiche nella sua adozione. Ci sono diversi punti di attenzione che possono essere presi in considerazione nella visione di alto livello del progetto, aldilà della metodologia adottata, ovvero:

- Importanza del coordinamento con il cliente;
- Importanza della comunicazione attraverso le funzioni aziendali;
- Importanza dei processi e degli strumenti;
- Importanza dei cambiamenti progettuali;
- Importanza della documentazione;

A fronte di questa analisi di alto livello si può quindi definire l'approccio da utilizzare, posizionando obbiettivi realistici attraverso un team adeguato e adeguatamente istruito, utilizzando un approccio iterativo che preveda strumenti di monitoraggio e coinvolgimento del team e dei suggerimenti da loro provenienti. La fase inziale del progetto è una delle parti più complesse a cui bisogna porre la massima attenzione in quanto è necessario definire il problema (o lo Scope), identificare i potenziali gap dell'implementazione suggerita, formalizzare le ipotesi e infine attribuire le attività ai vari team.

All'inizio dell'implementazione è necessario un passaggio fondamentale, in ottica di coordinamento e coinvolgimento globale, è necessario condividere in maniera chiara la vision del progetto.

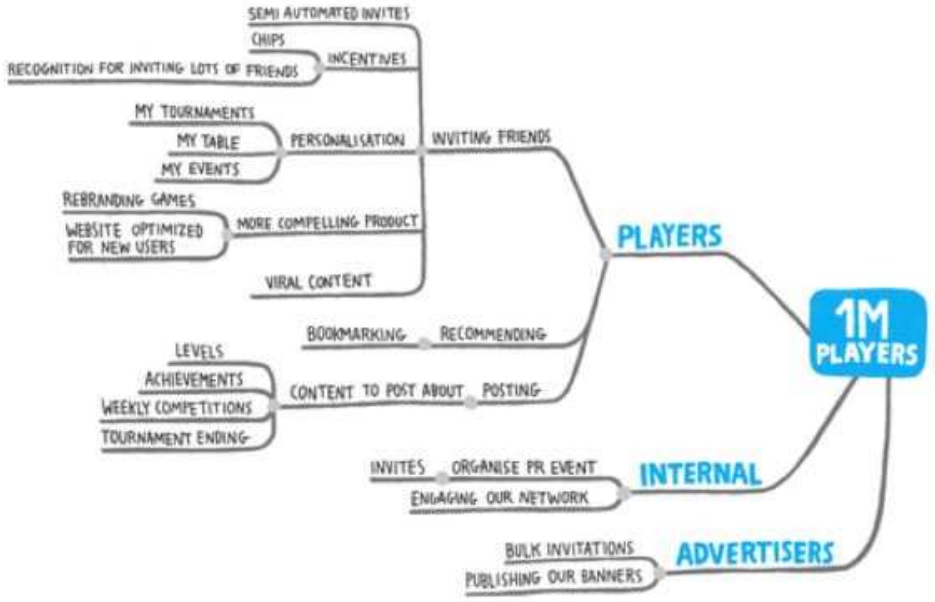
L'Impact Mapping è una tecnica di pianificazione strategica utilizzata per condividere la visione del progetto nella sua globalità, comunicare chiaramente le assunzioni, identificare e condividere gli obiettivi. Tramite questa tecnica viene creata una mappa (Impact map) in cui le *deliverables* di progetto vengono definiti come impatti che si desidera ottenere sulla base dei nuovi comportamenti attesi dagli attori coinvolti. Le funzionalità da realizzare vengono quindi dedotte a partire dalle finalità di business che si vogliono raggiungere e sono sempre riconducibili ad un valore di business.

L'impact Mapping porta anche altri vantaggi:

- Facilita la collaborazione e l'interazione del gruppo di lavoro durante la definizione dei confini del progetto;

- Visualizza chiaramente le assunzioni, permettendo in un secondo momento di prendere decisioni più velocemente basandosi sulle assunzioni fatte;
- Gli incontri di definizione della Impact map sono pochi e veloci;

Finalizzata, la mappa assume un aspetto analogo a quello proposto nell'immagine seguente:



La mappa a partire da un singolo obiettivo di business, lo esplora in base al contesto progettuale ed è possibile quindi disegnare più mapper per più obiettivi.

Considerando l'esempio di mappa riportato sopra è possibile leggerla da sinistra verso destra e ci si sposta dai deliverable, ovvero le singole funzionalità offerta dal sistema, all'obiettivo di business da raggiungere, riportato a destra nella mappa, passando per gli attori coinvolti.

La Impact Map con i suoi rami permette di rispondere a importanti domande progettuali:

- Why?
- Who?
- How?
- What?

Con **why** intendiamo il "perché stiamo affrontando questo progetto", quali obiettivi, goal, vogliamo raggiungere. Nella mappa questi goal sono la componente centrale da cui si dipartono i diversi rami. Alcuni esempi di

obiettivi possono essere incrementare le vendite dalle piattaforme informatiche, aumentare il bacino di utenti o migliorare l'usabilità di un determinato sistema. I goal che si identificano devono essere S.M.A.R.T., ovvero:

- Specifici, la loro descrizione deve indicare con precisione ed efficacia di cosa si tratta
- Misurabili, per avere un criterio oggettivo per valutarne il raggiungimento;
- Raggiungibili (Achievable) e condivisibili;
- Rilevanti, realistici, attinenti alla realtà aziendale;
- Avere una scadenza (Timely) entro la quale valutare se sono stati raggiunti;

È molto importante che gli obiettivi siano misurabili al fine di valutare l'andamento del progetto ed eventualmente correggere in corso d'opera con cicli di Inspect e Adapt.

I primi rami che incontriamo partendo dall'obiettivo (o goal) riportano gli attori, rispondendo di fatto alla domanda sul **Who**:

- Chi verrà impattato da quanto stiamo realizzando?
- Chi sono i clienti?
- Chi può influenzare la riuscita del progetto?

Il livello successivo ci dice come (**How**) gli attori sono colpiti dal progetto e con quali impatti, cercando di analizzare come potrebbe cambiare il comportamento degli utenti; in altre parole cerca di identificare quale sarà il reale impatto dell'utente a seguito del cambiamento:

- Cosa possono cominciare a fare?
- Cosa possono smettere di fare?
- Cosa possono fare diversamente (più velocemente, meno di frequente, più facilmente)?

Infine è necessario rispondere alla domanda relativa al **What**: cosa bisogna fare per supportare gli impatti identificati, ovvero, quali sono i deliverable del progetto che verranno messi a disposizione degli utenti. A conclusione del ramo troviamo il punto terminale che rappresenta quindi la funzionalità da realizzare.

La mappa è il risultato del lavoro del team di progetto (persone del business, esperti di dominio, tecnici). Le diverse capacità, approcci e prospettive delle persone che compongono il team vengono enfatizzate per ottimizzare il risultato creato portando al tavolo il contributo di tutti. Questo momento comune di brainstorming è l'occasione per condividere i confini del progetto e le priorità, garantendo quindi un ambiente ottimale per pianificare in maniera strategica quanto rilasciare.

La mappa in genere viene creata all'inizio di un progetto, perché permette di chiarire le idee su quanto si vuole ottenere, chi è coinvolto e come raggiungere l'obiettivo di business. Tuttavia, durante la realizzazione del progetto, la mappa può essere aggiornata in funzione di nuove informazioni ed utilizzata come strumento strategico per verificare i risultati ottenuti e di conseguenza gestire lo scope in funzione di questi.

Per la creazione della mappa si predispongono due sessioni differenti in cui:

- Definire gli obiettivi e le misurazioni attese;
- Creare la mappa;

Durante il primo incontro vanno esplicitati i reali obiettivi del progetto, dunque bisogna essere in grado di esprimere chiaramente la vision del progetto. Oltre all'identificazione degli obiettivi è fondamentale capire come misurarli. La ricerca di un'unità di misura è un ottimo occasione per verificare la fattibilità e le priorità.

Chiariti gli obiettivi, si può procedere con un secondo incontro durante il quale la mappa viene disegnata identificando gli attori coinvolti, i comportamenti impattati, le diverse alternative di deliverable e le priorità degli elementi della mappa.

Il beneficio di questa mappa sta nel collegare i deliverable al goal centrale tramite attori e impatti, evidenziando come i deliverable supportano gli impatti e come sono collegati all'obiettivo di business. Il quadro d'insieme permette di vedere in un colpo d'occhio tutti i deliverable e di identificare quali permettono di ottenere valore più velocemente, scartando ciò che non è (o non è più) necessario.

La mappa è uno strumento di pianificazione strategica perché aiuta a fare diversi ragionamenti su:

- Quali attori e quali impatti supportare;
- Riflettere sugli andamenti del progetto e sui feedback riprioritizzando / aggiungendo / eliminando deliverables;
- Quali deliverable stanno contribuendo o meno al goal;

Comunicando lo scope e il goal, l'Impact Map fa anche da Roadmap: ogni volta che si rilascia un deliverable si valuta se l'obiettivo è stato raggiunto o meno. Questa analisi permette di cambiare strategicamente direzione, interrompendo le attività su features che non stanno dando valore (eliminando waste) e/o smettendo di introdurre ulteriori funzionalità nel momento in cui si rileva l'avvenuto raggiungimento dell'obiettivo. Lo strumento favorisce quindi un approccio iterativo ed incrementale. È importante sottolineare che proprio perché concludere un progetto centrando l'obiettivo è fondamentale, lo scope della soluzione finale può essere diverso dall'idea iniziale. La mappa, riportando deliverable orientati

al supportare gli impatti e definendo degli obiettivi misurabili, si propone anche come strumento per tenere sotto controllo la qualità. In ogni momento si può verificare se quanto rilasciato soddisfa le esigenze di un attore e quanto contribuisce al raggiungimento dell'obiettivo principale.

Oltre a coadiuvare la gestione di un progetto, l'Impact Map, con i suoi goal-attori-impatti-deliverable è uno strumento per la raccolta dei requisiti.

Con l'approccio agile l'Impact Map è utile per identificare anche le User Story. Considerando il template di User Story: As <user> I want to <action> So that <reason>. Lo user corrisponde ad un attore della mappa, l'action ad un deliverable e l'impatto collegato al deliverable è proprio la reason della user story.

Come visto una parte essenziale della gestione progettuale è la stesura del piano di progetto e del piano di release, entrambi sono gli strumenti necessari per condividere con tutti gli obiettivi progettuali, le fasi e la pianificazione dell'intera implementazione. All'interno dell'implementazione l'elemento principale risiede sicuramente nella release ovvero nella parte di software che verrà sviluppata e che farà parte del software finale. All'inizio di goni release è prevista una sessione collettiva per la stima della Release dove il team di esperti con conoscenze diverse valuta la complessità in una specifica unità di misura che mostra la complessità dell'effort (Story Point). La sessione di stima è chiamata Poker Sizing Session (PSS).

In un progetto agile, la PSS viene svolta all'inizio di ogni Release ed è volta a valutare le funzionalità previste nella Release oggetto di stima. Il Referente PSS, generalmente il Service Manager, organizza la PSS per stimare le funzionalità da implementare in Story Point (SP) oppure in giorni/uomo per gli oneri aggiuntivi (questo è il caso, per esempio, di Test Aggiuntivi, PMO, Utilizzo Test Factory, che vengono stimati in giorni/uomo anche quando sono considerati tra le funzionalità da portare in sessione PSS).

Alla PSS prende solitamente parte, oltre al Referente PSS, un Referente di stima per ogni ambito applicativo impattato dalla Release. In aggiunta, se la Release lo richiede, possono essere coinvolti i seguenti ruoli di progetto:

- Il Product Owner di Business in veste di utente;
- Gli eventuali Product Owner Area IT;
- Il Referente Infrastrutture IT;
- Il Test Manager;
- Un referente della struttura Architetture Applicative e Tecnologiche;

Di seguito sono riportate le attività preliminari allo svolgimento della PSS:

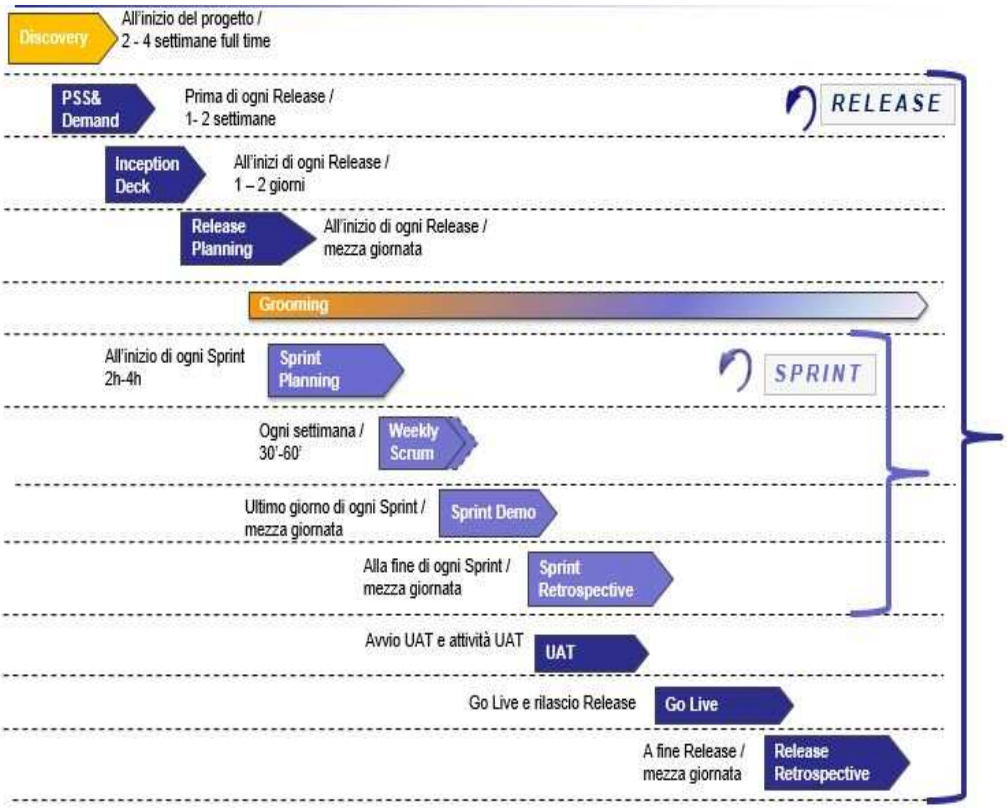
- Ogni Referente di Stima deve dichiarare che la metodologia applicata è quella Agile;
- Ogni Referente di Stima, per quanto di competenza, identifica le funzionalità oggetto di implementazione che dovranno essere discusse durante la PSS, coinvolgendo tutte le figure necessarie ad esplicitarle in maniera esaustiva;
- È possibile l'utilizzo di Catalogo di Funzionalità Standard, cioè un elenco predisposto di funzionalità risultate maggiormente ripetitive nei progetti di sviluppo software. Il Catalogo di Funzionalità Standard ha lo scopo di "standardizzare" le voci da stimare in PSS per facilitarne la comprensione durante la discussione e rendere possibili analisi statistiche e trend;
- Per ogni funzionalità deve essere indicata l'Epica di riferimento per la Release in stima, ripresa dal Product Backlog (è possibile indicare più funzionalità afferenti ad un'unica Epica);

Per quanto riguarda lo svolgimento della sessione PSS i passi seguiti sono:

- Il referente di Stima descrive le funzionalità ai partecipanti;
- Con una votazione di tutti i partecipanti viene valutata la complessità dell'attività in SP;
- Si discutono quindi le votazioni più alte e più basse e, mediamente al secondo giro di votazione, si raggiunge il voto unanime;

Successivamente alla PSS, ogni Referente di stima, identifica gli attributi necessari per completare la stima dell'effort da erogare per lo sviluppo delle funzionalità valutate (giorni esterni e interni, Fornitori ipotizzati, quote eventualmente anticipate in "avvio lavori", pluriannuali) e consolida la stima in modo che il SM possa correlare la stima all'iniziativa. L'output della PSS, unitamente alle eventuali ulteriori stime progettuali, contribuisce ad identificare il budget necessario per la realizzazione della Release e i conseguenti processi di approvazione del budget.

Nel processo di implementazione della metodologia sono presenti una serie di cerimonie e artefatti che devono essere applicati in linea con i principi della metodologia. L'attività di pianificazione è precedente all'inizio delle attività di implementazione ma alcune attività come vedremo sono cicliche e si ripetono durante tutto il corso del progetto. Un esempio di passi successivi che si susseguono dalla fase di Discovery alla fase di go-live è mostrato nell'immagine di seguito.



Le cerimonie si riferiscono alle fasi di Discovery, di PSS, di Inception, di Release Planning, di Grooming, di Sprint Planning, di Weekly Scrum Meeting, di Sprint Demo, di Sprint Retrospective, di UAT, di GoLive e infine di Release Retrospective.

Due termini che risultano spesso utilizzati nella metodologia Agile sono le User Story e le Epiche. Le Epiche sono un requisito di alto livello che successivamente viene esploso in item (User Stories) più piccole. Per quanto concerne invece le User Story, esse sono descrizioni brevi del sistema dal punto di vista dell'utente e lo scopo è di definire un bisogno di business visto dal punto di vista dell'utente utilizzando un linguaggio naturale e comprensibile sia da personale tecnico sia da persone con minori competenze tecniche. La sintassi è molto semplice e prevede un Ruolo, un Azione e un Obbiettivo che vengono declinati:

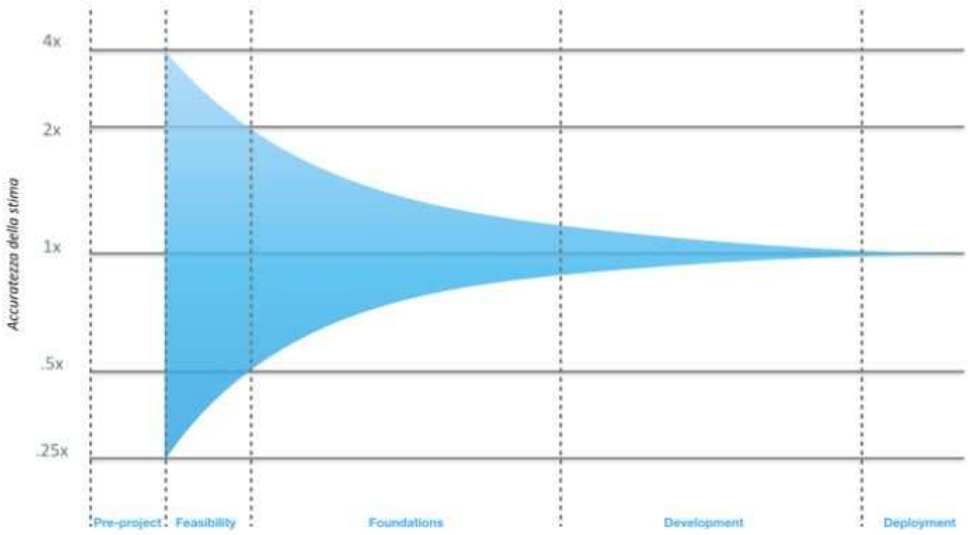
COME <RUOLO viene definito il ruolo come quello di utente finale del sistema>

VOGLIO <AZIONE viene esplicitato quello che il sistema dovrà fare>

PER <OBBIETTIVO l'utente esplicita la necessità finale ovvero l'obbiettivo da raggiungere>

Ogni User Story è fondamentale che abbia dei criteri di accettazione che devono essere verificati una volta che l'implementazione viene rilasciata per il test, questi criteri devono essere verificati affinché la storia venga considerata finita e accettata. I criteri contegno dettagli implementativi o precondizione e vengono condivisi tra il team e il Product Owner costituendo la base dei test case.

Una delle più grandi sfide nella gestione dei progetti è la loro stima. La stima come visto viene richiesto per diversi motivi da quello iniziale per analizzare la fattibilità a quello intermedio per valutare la sua esecuzione ed eventualmente la sua prosecuzione. Ogni stima deve essere quindi fatta con un obiettivo specifico ma allo stesso tempo è necessario sempre riconoscere i limiti di tale stima. Nella gestione del progetto è utile ricordare che alla stima è associato un certo grado di incertezza. Partendo dal grafico proposto di seguito che considera l'intero ciclo di vita di un progetto agile, gestire l'incertezza significa prevedere punti di controllo del progetto a fasi relativamente brevi e intensificare questi checkpoint nelle fasi iniziali in cui l'incertezza è più alta.



Deviazione dell'accuratezza	**Fase progettuale**	**Numero di punti di controllo**
Da 0,25 a 4x	Fase pre - progettuale – Avvio del progetto e analisi di fattibilità	2 incontri settimanali
Da 0,5 a 2x	Scope e obiettivi definiti – Inizio del progetto	1 incontro settimanale

Da 0,67 a 1,5x	Analisi dei requisiti completato – Fase di sviluppo	1 incontro bi - settimanale
Da 0,8 a 1,25x	Chiusura dl progetto e rilascio della soluzione	1 incontro settimanale
0	Soluzione applicativa correttamente rilasciata e fase di *baby-sitting*	Da valutare in base alla fase di follow - up

CAPITOLO TERZO – CASO DI STUDIO

Metodo Agile Applicato a una società di servizi

Prendendo in considerazione una grande società di servizi, in cui i progetti sviluppati sono per lo più multi fabbrica e multi fornitore e quindi composti da diversi cantieri, il modello adottato è uno Scrum di Scrum a due livelli: livello di Governo e livello di Execution.

Prima di affrontare l'analisi dell'approccio utilizzato è utile definire dapprima i ruoli e le figure che vengono coinvolte nell'intero ciclo di vita del progetto. Una prima suddivisione viene fatta in base al fatto che si tratti di Area di Business, Area IT o Fornitori.

Per la parte Business avremo:

- Business Sponsor: colui che ha la visione di Business del prodotto e la esprime attraverso requisiti e desiderata di alto livello;
- Product Owner Business: è individuato all'inizio del progetto, può essere uno o più di uno, recepisce con continuità le esigenze del Business e le declina in «Epic» e «User Stories». Collabora con i Product Owner dell'Area IT per definire le priorità globali e rispondere a domande puntuali sui requisiti del prodotto. Definisce i criteri di accettazione del prodotto ed è responsabile, con il Capo Progetto, del rispetto di costi e tempi.

Per l'Area IT avremo i seguenti ruoli:

- Service Manager: Responsabile della relazione col Business, supporta e facilita il Product Owner Business nelle sue attività.
- Product Owner Area IT (Capo Progetto): Ha la visione integrale dei diversi cantieri che contribuiscono alla realizzazione del prodotto, nel rispetto dei tempi di progetto. Valuta l'andamento complessivo del progetto. Collabora nella definizione e prioritizzazione delle User Stories del cantiere di competenza. È responsabile della realizzazione delle «User Stories» di competenza. Infine ingaggia il Fornitore e risponde alle sue domande.
- Product Owner Area IT (altri cantieri): È responsabile della realizzazione delle «User Stories» di competenza, nel rispetto dei tempi di progetto. Ingaggia il Fornitore e risponde alle sue domande. Collabora nella definizione e prioritizzazione delle User Stories del cantiere di competenza. Attiva l'escalation a fronte di rischi/issue di elevata severità.

- Test Owner: Supporta i Product Owner Area IT nella progettazione/definizione dei test funzionali, di integrazione e di non regressione. Supporta, inoltre, il Product Owner Business nella definizione dei test di accettazione.
- Referente Servizio Infrastrutture Tecnologiche (SIT): Supporta le attività infrastrutturali IT come disponibilità di ambienti/infrastruttura e change.
- Scrum Master: Garantisce e facilita il processo agile (rispetto cerimonie, gestione artefatti), aiuta nella gestione e verifica di tempi/costi/pianificazione. Rimuove gli impedimenti e gestisce le issue di tutti i Product Owner.

Infine nel caso di coinvolgimento di fornitori esterni i ruoli aggiuntivi identificabili con il loro coinvolgimento sono fondamentalmente due:

- Referente Fornitore il quale recepisce quanto richiesto dal Product Owner dell'Area IT, espone l'andamento delle iterazioni durante gli sviluppi ed e portavoce del team di sviluppo.
- Team Fornitore: è il gruppo di lavoro del fornitore costituito da analisti/sviluppatori/tester e Scrum Master che implementano quanto condiviso nella pianificazione.

L'gile è un approccio allo sviluppo di prodotti software che permette alle organizzazioni ed alle loro persone di adattarsi e rispondere efficacemente al cambiamento, in linea con i valori del Manifesto Agile e dei suoi princìpi ispiratori.

L'agile è una "disciplina" basata su un insieme di pratiche che abilitano clienti e sviluppatori a realizzare prodotti in maniera iterativa ed incrementale, grazie al coinvolgimento e la collaborazione sia di stakeholder interni (es. il team di sviluppo) che esterni (es. cliente/committente).

Come già illustrato l'ambito tipico dell'Agile sono contesti con alto livello di incertezza, dove i cambiamenti avvengono rapidamente. In tali contesti complessi, per consegnare un prodotto di qualità che soddisfi il cliente, è necessario un metodo alternativo alla pianificazione classica. La flessibilità e la rapidità nell'affrontare i cambiamenti spinge all'adozione di un approccio "value driven" piuttosto che "plan driven".

Con l'aggettivo agile si caratterizzano tutte quelle metodologie di gestione progettuale che utilizzano un approccio iterativo ed incrementale nell'affrontare le problematiche tipiche di un dominio complesso: contesti in cui non sono già disponibili o conosciute risposte o regole applicabili.

In ambiti caratterizzati da forte incertezza e complessità e quindi caratterizzati da molti cambiamenti, l'approccio consigliato è quello iterativo

ed incrementale: veloci cicli di feedback che permettono di integrare le nuove informazioni nel prodotto che si sta realizzando.

I cambiamenti da fronteggiare sono endemici in questo tipo di ambiti: si parte dal presupposto che l'utilizzatore del prodotto ha solo un'idea abbozzata di ciò che vuole, idea che si raffinerà facendo delle prove e sperimentando delle versioni intermedie.

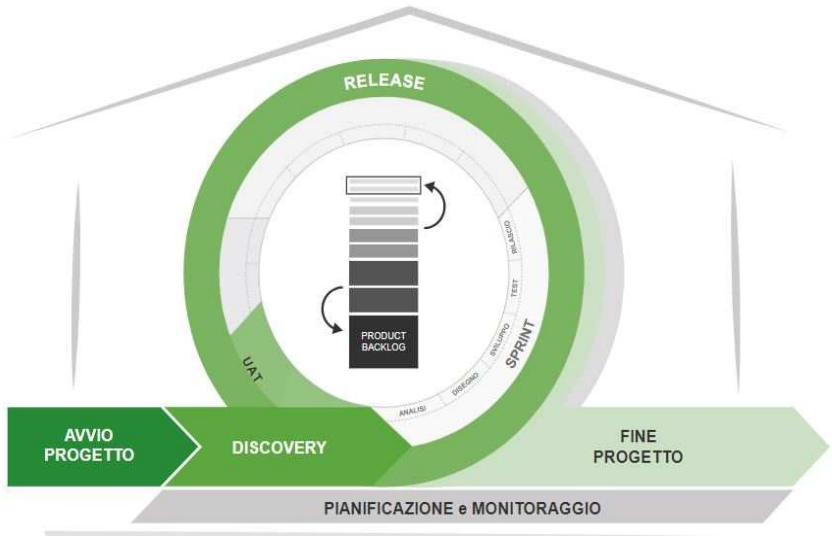
Inoltre, più sono lunghi i tempi di realizzazione e consegna, più è necessario essere pronti ad accogliere delle modifiche, sapendo che inibire il cambiamento potrebbe non essere utile: potrebbe infatti coincidere con la perdita di nuove occasioni nate grazie ai feedback intermedi dell'utilizzatore.

Nella direzione di gestire ed integrare il cambiamento, le metodologie Agili consigliano una pianificazione continua, fatta iterazione per iterazione, che tenga conto dei risultati ottenuti e dei feedback emersi nell'iterazione appena conclusa. Non essere pronti al cambiamento significa non solo un'alta probabilità di aumento esponenziale dei costi di realizzazione del prodotto, ma anche un incremento del rischio di presentare al cliente e al mercato qualcosa che non soddisfi le esigenze attuali. L'alternativa è quindi quella di prepararsi ad accoglierlo (embrace change) tramite l'implementazione di un processo che preveda dei cicli di inspect and adapt: a fronte di azioni si valuta il risultato per avere un riscontro del valore che stiamo ottenendo. Questa valutazione ci porterà a rivedere le nostre azioni correggendole per massimizzare il risultato ottenuto. In altre parole, pianifichiamo delle azioni da fare (plan), le eseguiamo (do), controlliamo il risultato ottenuto (check) e aggiustiamo il nostro processo (adjust), secondo il ciclo Plan-Do-Check-Adjust (PDCA). Parliamo quindi di sistemi retroattivi basati su feedback: l'output ottenuto si traduce in nuova conoscenza utilizzata per migliorare il processo e il risultato stesso. Questa ciclicità si riflette nel lavorare per iterazioni fisse (dalle 2 alle 3 settimane) durante le quali solo alcuni aspetti applicativi/progettuali sono oggetto del lavoro del gruppo: quelli considerati prioritari e di cui i dettagli sono noti ad un giusto livello che ne permetta l'attuazione. Al termine di ciascuna iterazione il risultato viene condiviso e ne viene verificata l'attinenza con le aspettative del cliente. Quanto emerge dai momenti di condivisione e verifica aumenta la conoscenza del gruppo sul prodotto/progetto dando la possibilità di dettagliare meglio quanto ancora da sviluppare, *prioritizzare* di nuovo gli sviluppi mancanti e agire sul processo stesso migliorandolo. Al termine di ogni iterazione abbiamo quindi un incremento di prodotto e un aumento della conoscenza del dominio e delle problematiche da risolvere. Agile è anche l'oggettivo che accompagna il termine ombrello nel riferirsi ad un insieme di metodologie e framework per lo sviluppo di prodotti. Tra queste una delle più popolari è appunto lo Scrum. Lo Scrum è un'architettura logica che prevede ruoli, cerimonie e artefatti al fine di produrre software funzionate in iterazioni che, come detto, di solito durano

2-3 settimane chiamate Sprints. Abbiamo già analizzato i ruoli previsti; per quanto riguarda le cerimonie, si parla di 4 cerimonie: Sprint Planning, Daily Meeting, Sprint Demo, Sprint Retrospective.

Nel modello Agile proposto è prevista la suddivisione dell'intero progetto in fasi ben definite con obbiettivi chiari e rispondenti tutti alla logica snella e iterativa propria dell'approccio.

Le Fasi



Come illustrato nell'immagine soprastante le fasi previste sono 6:

Avvio Progetto (Fase 1)

La fase iniziale di un progetto agile racchiude le attività volte a definire le esigenze di business e lo start delle fasi realizzative. Questa fase segue l'assessment sulla metodologia nel caso in cui l'operazione si concluda con la scelta, sia lato IT che lato business, di usare la metodologia Agile per il progetto. Nella fase "Avvio progetto" viene compresa e condivisa l'esigenza di business, chiarendo un perimetro progettuale di alto livello. A questo scopo viene individuata una lista di requisiti di business, ordinati per priorità utilizzando eventualmente tecniche agili e infine viene censita ufficialmente l'iniziativa progettuale.

L'operazione di definizione dei requisiti di business consiste nella raccolta e condivisione delle esigenze di business che hanno originato la richiesta, allo scopo di formulare una ipotesi di alto livello di perimetro progettuale.

Se l'area di business è esterna all'Area IT, le esigenze di business vengono raccolte a cura del Service Manager di riferimento.

Diversamente, se il business è interno all'Area IT, le esigenze di business vengono raccolte a cura del Coordinatore Operativo di Attività, individuato in base all'area applicativa prevalente interessata dalle esigenze.

Attraverso incontri mirati a cura del Service Manager o del Coordinatore Operativo di Attività, il Product Owner di Business, il referente applicativo prevalente e i principali stakeholder individuati, collaborano per individuare e condividere i bisogni dell'utente che hanno originato la richiesta.

La condivisione dello scopo del progetto, ovvero delle motivazioni di business (esigenza di nuove funzionalità e/o tecnologie, cambiamenti organizzativi, eventuali problemi da risolvere ecc.) e dei benefici attesi è il punto di partenza per una definizione efficace dei requisiti di business. È inoltre un momento fondamentale per fare una prima ipotesi di perimetro complessivo del progetto realizzativo (ad esempio quali e quante sono le strutture applicative potenzialmente coinvolte dalla soluzione). L'output di questa operazione è dunque l'elenco condiviso dei requisiti di business.

In un progetto agile i requisiti di business verranno ulteriormente approfonditi durante la fase di Discovery da parte del team di progetto, in particolare con il supporto della vision del Product Owner Business.

Se l'area di business è esterna alla Area IT, la richiesta viene censita dal referente Service Manager competente per l'area di business. Diversamente, se il business è interno alla Area IT, il censimento dell'iniziativa viene fatto dal Coordinatore Operativo di Attività della struttura ritenuta più impattata.

In questa fase viene formato il primo nucleo del team di progetto con i ruoli previsti dal modello Agile. Con il supporto del Team Metodologie Strumenti e Gestione viene valutata la collaborazione di un Coach Agile per un eventuale ulteriore supporto nell'applicazione della metodologia. La fase di avvio del progetto si conclude con l'incontro di Kickoff di progetto, che avvierà la fase di Discovery.

In questa fase il Product Owner e il Capo Progetto, eventualmente con il supporto del Coach Agile, prende visione dei processi abilitanti il modello agile:

- Assessment;
- Fabbisogni;
- Acquisti e fornitori;
- Pianificazione;
- Monitoraggio budget;
- Documentazione;
- Survey e Customer Satisfaction;

Nella fase di avvio, in base al contesto progettuale, è possibile iniziare indagini di mercato per individuare soluzioni e prodotti commerciali disponibili attraverso lo Scouting di Mercato.

Scouting di Mercato

L'operazione "Scouting di Mercato" è un'attività esplorativa volta ad analizzare lo scenario delle soluzioni di mercato, i relativi vendor e l'eventuale gap analysis rispetto alla richiesta di business.

Lo scouting è un'attività opzionale, generalmente effettuata con il supporto del Capo Progetto. In generale, lo scouting si effettua quando è in fase di valutazione l'adozione di un prodotto o una tecnologia nuova e si richiede un'indagine su cosa il mercato offra, per quali applicazioni e con quali referenze. In alternativa, lo scouting viene effettuato quando si ipotizza l'esistenza sul mercato di soluzioni e/o prodotti già realizzati, commercialmente disponibili, che possono indirizzare, anche in parte, i requisiti di business che in questa fase sono ancora ad alto livello.

L'analisi dello scenario viene effettuata esplorando il mercato:

- Vengono inizialmente individuate le soluzioni sul mercato;
- Successivamente si passa ad individuare i vendor attivi nell'ambito di interesse, o i relativi soggetti proprietari delle tecnologie ipotizzate (es. soluzioni di nicchia, centri ricerca e innovazione, startup che possono non essere in perimetro agli analisti perché non ancora diffuse);
- Una volta selezionato un panel di candidati, si procede con un'analisi approfondita delle soluzioni (focus sui vendor) per individuare quelle da ritenere più interessanti e ben posizionate, secondo i seguenti driver:
    - Panoramica della soluzione offerta;
    - Punti di forza della soluzione;
    - Punti di debolezza della soluzione;
    - Posizionamento nel Magic Quadrant di Gartner (Modello che rappresenta il posizionamento dei Vendor nel mercato, suddividendoli nelle categorie Leaders, Challenger, Visionaries e Niche Player e che può rappresentare un plus ai fini della selezione);

L'attività di scouting, specialmente su soluzioni di nicchia o innovative, oppure non tracciate da analisti, può essere seguita da una sperimentazione in laboratorio in ambienti completamente segregati rispetto al resto dei sistemi e poi da una sperimentazione pilota, detta anche Proof Of Concept (o POC), che può basarsi anche su dati reali di produzione, per valutare concretamente le funzionalità e le potenzialità offerte dalle soluzioni di mercato oggetto di POC.

Se vengono rilevati scostamenti tra le soluzioni individuate rispetto ai requisiti di business (a seguito o meno di una POC), il Capo Progetto con il supporto del team di progetto effettua una *gap analysis* a livello di funzionalità. Per le funzionalità non fornite dalla soluzione, vengono valutate:

- La possibilità di realizzare customizzazioni da parte del Vendor o altro System Integrator;
- La possibilità di sviluppare internamente funzionalità sostitutive integrabili con il prodotto;
- L'impossibilità di implementare la funzionalità;

Il risultato delle attività di scouting e gap analysis forniscono elementi che concorrono alla scelta finale del prodotto, alla stima dei costi (una tantum e ricorrenti), alla pianificazione del progetto e alla definizione della strategia di sourcing più adatta per il progetto.

Strategia di Sourcing

L'operazione si Sourcing consiste nella definizione della modalità di scelta delle forniture esterne necessarie alla realizzazione della soluzione IT del progetto.

Il Capo Progetto con il supporto del Service Manager, svolge le attività utili ad individuare la modalità di scelta delle forniture per il progetto, in base alle esigenze di business a livello macro o a requisiti di business più strutturati.

Gli obiettivi della strategia di sourcing per il progetto riguardano:

- L'esigenza di effettuare uno scouting preliminare;
- La tipologia di soluzione;
- Make (creazione di un prodotto da sviluppare ex novo, con l'adozione di una soluzione "custom");
- Buy (acquisto di un prodotto disponibile sul mercato, eventualmente da integrare e/o personalizzare, con l'adozione di una soluzione a "pacchetto");
- Mix delle due tipologie (soluzione "best of breed");
- La modalità di realizzazione interna e/o esterna, in base al personale impiegato;
- L'approccio di negoziazione;
- Trattativa diretta;
- Confronto di mercato (Request for Information o Request for Proposal);

La strategia di sourcing individuata fornisce le indicazioni per avviare operativamente la selezione delle forniture.

L'operazione si riferisce al reperimento dei razionali di decisione, alla finalizzazione della selezione d'acquisto (basata su offerta di mercato RFI/RFP o trattativa diretta per le forniture esterne che realizzeranno le attività progettuali), e al consolidamento della scelta make or buy.

L'attività si svolge dopo la raccolta dei macro requisiti e la definizione della strategia di sourcing per il progetto. Se la strategia di sourcing progettuale prevede di avvalersi di fornitori esterni per le attività di sviluppo, il Capo Progetto (Project Manager o Product Owner Area IT), con il supporto del team di progetto, prepara un documento di capitolato per avviare le attività di gara.

Nel caso di progetto agile, il capitolato sarà a copertura della prima Release ed espliciterà che il progetto sarà gestito in base al modello agile.

In caso di progetto Waterfall, il capitolato sarà a copertura dell'intero progetto (o fase progettuale).

In base alla scelta effettuata in strategia di sourcing, il Capo Progetto procede con una delle seguenti attività:

- Trattativa Diretta > attività negoziale diretta verso un unico interlocutore. Questa tipologia è ammessa per forniture urgenti o che per loro natura escludono una pluralità di offerenti.

oppure

- Request for Information (RFI) [opzionale] > confronto strutturato tra più Fornitori (dopo aver individuato una rosa di potenziali Fornitori) con la raccolta di un primo nucleo di informazioni finalizzato alla ricognizione delle possibili soluzioni, modelli di pricing, quotazioni indicative;
- Request for Proposal (RFP) > confronto strutturato tra più Fornitori in cui la valutazione delle offerte tecniche anticipa la valutazione delle offerte economiche;

A seguito della RFI/RFP il Capo Progetto può consolidare o meno la scelta effettuata sul tipo di soluzione make or buy individuata precedentemente nella operazione strategia di sourcing.

Di norma, l'RFI non porta alla stipula di un contratto: in questo caso la RFI si può dichiarare conclusa senza acquisizione di forniture esterne. Nel caso di RFP, l'attività si conclude con l'individuazione del miglior offerente, considerando sia la componente tecnica sia economica.

In caso di progetto agile all'RFP si applicano le seguenti logiche:

- Prima Release > assegnazione con perimetro definito da capitolato ‘componente a corpo';
- Release successive > identificazione di un “rate medio” per il progetto in questione;

Nel caso di Trattativa Diretta le attività progettuali sono assegnate all’unico interlocutore individuato.

In entrambe i casi, a seguito delle attività di RFP o Trattativa Diretta, si prosegue con l'assegnazione della fornitura e la stipula dei contratti con i Fornitori selezionati.

In conclusione generalmente le attività di RFI/RFP durano dalle 6 alle 8 settimane.

Pianificazione e Monitoraggio (Fase 2)

La fase di 'Pianificazione e Monitoraggio' è composta dalle attività volte alla previsione temporale dei rilasci e delle attività progettuali, accompagnate dalla costante verifica dell’andamento del progetto. È per questo una fase che avrà inizio a partire dalla Discovery e si concluderà con la fine del progetto.

Le attività di pianificazione e monitoraggio riguardano il governo del progetto a partire dai tre principali driver: tempo, costi, scope. In un progetto agile lo scope è sempre variabile, mentre il tempo e i costi tipicamente sono fissi. Sarà cura del Product Owner, Capo Progetto con il supporto del team di progetto, monitorare tempo, costi e scope in modo da attuare eventuali misure di adattamento, focalizzate sempre al rilascio di valore per il business. Durante questa fase, il Test Manager definisce la strategia di test, progetta e monitora i test. La pianificazione e il monitoraggio consentono di verificare l'andamento del progetto, rilevare punti di forza e debolezza del progetto, evidenziare eventuali criticità.

Come detto la pianificazione è un’attività costante durante il progetto e si articola su più livelli:

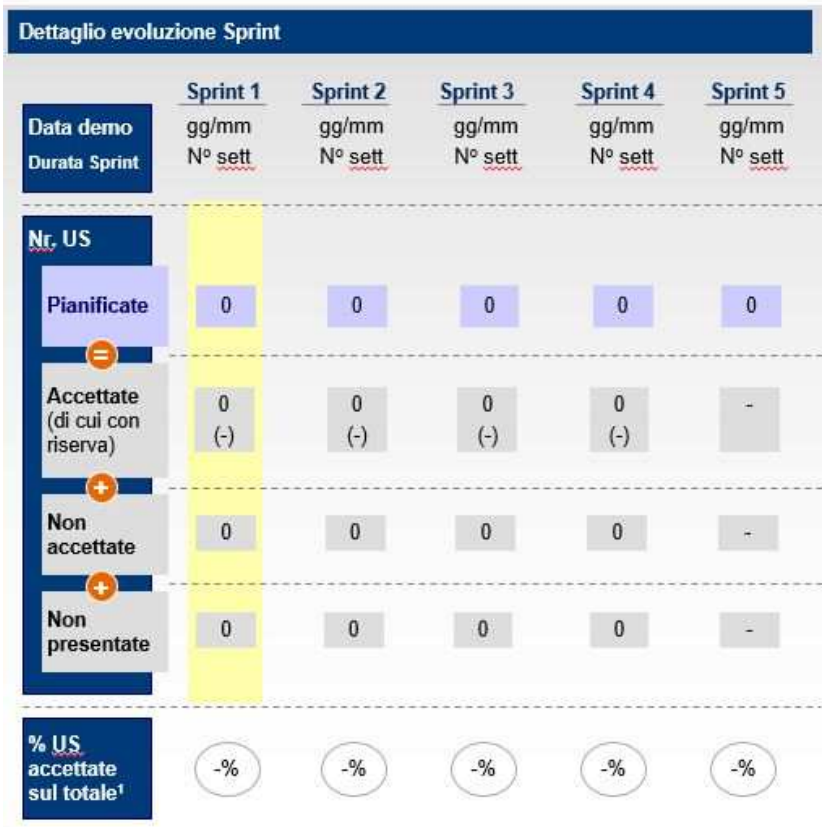
- Livello di progetto (disegno della prima Roadmap in fase Discovery);
- Livello di Release (disegno della Release, di norma in Inception e Release Planning);
- Livello di Sprint (in Sprint Planning e in pianificazione di attività e cerimonie dello Sprint);
- Livello giornaliero/settimanale (pianificazione delle attività giornaliere o settimanali rispettivamente nel Daily Meeting, Weekly Meeting);

Per la pianificazione del progetto, i Product Owner Area IT degli eventuali altri cantieri e le strutture aziendali necessarie (es. Sicurezza, Architetture, Infrastrutture), contribuiscono alla definizione del prodotto da rilasciare nel progetto.

Il Capo Progetto, con il supporto dello Scrum Master, monitora periodicamente l'avanzamento di progetto. È buona norma che la verifica dell'andamento e avanzamento del progetto venga fatta almeno una volta a Sprint, ad esempio dopo ogni cerimonia di Sprint Planning o di Sprint Review.

A supporto dell'attività possono essere utilizzati i seguenti strumenti che rappresentano il reale avanzamento del software realizzato:

- il Product Backlog che contiene i requisiti di progetto ordinati per priorità e analizzati dal team. Consente di verificare quanto e come sta crescendo lo scope;
- Burn-up che consente di verificare in quanto tempo e con che velocità vengono realizzati gli incrementi di prodotto;



Dettaglio evoluzione Sprint					
	Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5
Data demo Durata Sprint	gg/mm Nº sett	gg/mm Nº sett	gg/mm Nº sett	gg/mm Nº sett	gg/mm Nº sett
Nr. US					
Pianificate	0	0	0	0	0
Accettate (di cui con riserva)	0 (-)	0 (-)	0 (-)	0 (-)	-
Non accettate	0	0	0	0	-
Non presentate	0	0	0	0	-
% US accettate sul totale[1]	-%	-%	-%	-%	-%

- Cruscotto di monitoraggio budget che consente di confrontare il budget di progetto con lo scope variabile del progetto in corso;

Sempre in questa fase, il Capo progetto e il Product Owner, supportati dal team di progetto, prosegue il monitoraggio di rischi, vincoli, dipendenze e assunti, avviato in Discovery.

Il Product Owner e il Capo Progetto condividono gli esiti del monitoraggio del progetto con il team di progetto, in particolare con il Product Owner di Business.

Il monitoraggio del progetto, unitamente agli incontri di Product Backlog Refinement fornisce elementi per l'individuazione di possibili rischi ed opportunità. Abilita cioè il team a individuare tempestivamente il prodotto giusto al momento giusto.

Durante questa fase è necessario schedulare i meeting che coinvolgono il team di progetto, come le cerimonie di progetto (ad es. Sprint Planning, Demo, Retrospective) o gli incontri periodici di allineamento previsti (ad es. Daily Meeting, Weekly Meeting). Sarà cura dello Scrum Master condividere, pianificare e ricordare, nei tempi opportuni, gli impegni del team di progetto.

Il piano di progetto viene creato a cura del Product Owner e del Capo Progetto coerentemente con quanto emerso in Discovery e poi variato quando necessario, in particolare in Inception delle Release successive alla prima.

Gli incontri di Retrospective (a fine Sprint, Release o progetto), anche se tipicamente non riguardano i contenuti del progetto (scope, time, cost) forniscono un efficace feedback sull'andamento progettuale dal punto di vista dei processi. Danno quindi elementi significativi per il governo del progetto.

Tipicamente nei progetti agili il monitoraggio è basato sullo stato avanzamento reale dei lavori: alla fine di ogni Sprint si prende visione del software funzionante realizzato.

Oltre al burn-up chart, è possibile realizzare un burn-down chart che consente di verificare in quanto tempo e con che velocità decresce la quantità di lavoro ancora da realizzare.

Piano di Progetto

La definizione del piano di progetto comprende l'individuazione e la formalizzazione della time-line di progetto.

Il piano di progetto viene realizzato dal Product Owner Area IT e dal Capo Progetto che, utilizzando le informazioni contenute nel documento di Discovery, disegna il relativo diagramma di Gantt e lo formalizza nell'applicativo informatico.

Il disegno viene effettuato con la tecnica della pianificazione a "finestra mobile" e tipicamente la visibilità sul piano copre un orizzonte temporale di circa 3 mesi. In questo modo si ottiene un sufficiente dettaglio della Release in partenza e dei relativi Sprint che la compongono, mentre le macro-fasi delle Release successive vengono soltanto abbozzate, per poi specificarle in dettaglio in prossimità del successivo Release Planning.

Il Gantt riporta la previsione dell'andamento delle attività progettuali in termini di date ed impegno delle persone in un dato momento, consentendo di:

- Definire le date di inizio/fine delle attività necessarie per conseguire gli obiettivi progettuali;
- Evidenziare le date fondamentali del progetto, ovvero le "milestone";
- Pianificare l'impegno delle persone ed assegnarle alle singole attività;
- Identificare eventuali criticità legate ai vincoli delle diverse attività o alle dipendenze tra un'attività e l'altra;

Il Gantt definito all'inizio del progetto rappresenta una prima previsione che viene storicizzata come Baseline Iniziale. La Baseline, a livello di ogni Release, serve come punto di riferimento per verificare l'andamento progettuale e valutare gli scostamenti rispetto alla previsione iniziale, sia in termini di tempi sia in termini di impegno delle persone. La Baseline Iniziale resta quindi invariata nel tempo (e ad ogni nuova Release si effettua una Baseline Incrementale), mentre il piano di progetto è dinamico e rappresenta la situazione reale del progetto.

Allo scopo di avere un documento utile per condividere il macro-piano e valutare l'andamento del progetto con tutti gli stakeholder (ad esempio durante gli incontri di Stato Avanzamento Lavori - SAL) viene utilizzato il template Masterplan Agile in cui viene illustrato sinteticamente il quadro generale delle attività e delle date principali del progetto.

Masterplan Agile

Il Masterplan Agile è lo strumento formale per presentare lo stato generale del progetto con l'obbiettivo di illustrarlo in maniera chiara e immediata ai vari stakeholder non presenti in tutte le attività. Il Masterplan

Agile è inoltre la rappresentazione grafica che fornisce una vista di sintesi della Roadmap di Progetto (Masterplan di Progetto) e del piano di Release (Masterplan di Release).

Il Masterplan Agile (Progetto e Release) è normalmente contenuto nel documento di Fine Discovery. Per le Release successive alla prima, viene inserito nel documento di Inception. Inoltre è utilizzato anche nel documento di SAL di progetto.

L'obiettivo principale del documento è visualizzare in un unico punto le macro caratteristiche dell'intero progetto.

Il Masterplan di Progetto infatti consente di condividere con tutti gli stakeholder, in una modalità visuale e sintetica, il calendario complessivo del progetto e la sua composizione in termini di rilasci di incrementi auto-consistenti di prodotto. In altre parole, contenuti e tempi delle diverse Release. Inoltre è utile per condividere l'avanzamento di progetto nei momenti di Stato Avanzamento Lavori (SAL), come base di discussione sugli aspetti temporali e di monitoraggio delle principali milestone di rilascio delle diverse Release. Il Masterplan di Progetto è uno dei documenti fondamentali della fase di Discovery. Si sviluppa dall'inizio della fase di Discovery e si consolida alla fine della fase stessa.

Il Masterplan di Release ha invece un focus sui contenuti di una determinata Release ed ha l'obiettivo di condividere il contenuto di business e la pianificazione della Release in termini di durata e date degli Sprint, date di inizio e fine UAT e dei rilasci previsti.

Il Masterplan di Release è il principale output del Release Planning e va consolidato e condiviso nei tempi utili all'attivazione della Release:

- Per la prima Release normalmente nelle ultime settimane della fase di Discovery;
- Per le successive Release durante la fase di Inception, che si attiva temporalmente nelle settimane finali della Release precedente.

Discovery (Fase 3)

La fase 'Discovery' racchiude le attività necessarie per identificare, in base alle informazioni conosciute al momento, le caratteristiche principali del progetto da realizzare: stakeholder, obiettivi, macro requisiti del prodotto, stima dei costi, vincoli, rischi, architettura IT e Roadmap. In sintesi, la Discovery è la fase che consente di creare la big picture del progetto.

Quello che dà inizio alla fase di Discovery è il Kick Off.

l Kickoff è un incontro organizzato, indetto e condotto dal Product Owner Area IT e dal Capo Progetto.

L'incontro ha lo scopo di condividere la vision ad alto livello del prodotto da realizzare con il team di progetto.

Prima del Kickoff è auspicabile che siano state individuate le strutture realizzative maggiormente impattate e i principali stakeholder di progetto. In questo senso, è importante che vengano definiti tempestivamente, oltre al ruolo di Product Owner Area IT, del Capo Progetto, il ruolo Scrum Master e Product Owner di Business.

Prima del Kickoff deve essere dunque formato il team di progetto: se non vengono individuate tutte le persone necessarie, è bene che il team di progetto venga completato durante la fase di Discovery.

Per promuovere e valorizzare l'importanza del progetto, all'incontro di Kickoff è suggerito venga invitato il Business Sponsor.

II Kickoff si prefigge principalmente tre obiettivi:

- Condividere la vision del progetto con il nascente team di progetto;
- Far conoscere le persone del team di progetto;
- Avviare la Discovery;

Lato business può essere organizzato un Kickoff Cliente per ufficializzare l'iniziativa tra le diverse strutture di business interessate. A discrezione di chi organizza l'incontro, possono essere coinvolti referenti delle strutture applicative interessate.

Nei progetti agili, a differenza di quelli waterfall, il kickoff è un incontro che dà avvio alla fase di Discovery che mediamente può durare dalle due alle quattro settimane.

In questa fase il team di progetto si incontra per approfondire e "scoprire" le caratteristiche del prodotto che andrà a realizzare. Di seguito i principali argomenti da affrontare nella fase:

- Consolidamento del team di progetto;
- Mappatura degli stakeholder;
- Macro requisiti di progetto;
- Scelta di utilizzo di tecniche di prototipazione;
- Vincoli, assunti, rischi di progetto;
- Strategia di sourcing;
- Soluzione IT;
- Scelta di utilizzo di tecniche DevOps;
- Stima dei macro-costi di progetto;
- Roadmap di progetto;
- Planning della prima Release;

La fase di Discovery si articola in una serie di incontri organizzati e facilitati dallo Scrum Master. Lo Scrum Master pianifica gli incontri indicando chiaramente obbiettivi e agenda, invitando il team di progetto e i necessari stakeholder.

Sarebbe preferibile che almeno i primi incontri di Discovery avvenissero in co-location. In questo modo il team di progetto si conosce ed è possibile utilizzare più facilmente tecniche di visual management.

L'approccio del team di progetto, accompagnato e guidato dalla vision del Product Owner Business, sarà sempre orientato a rilasciare il più velocemente e frequentemente possibile valore di business, realizzando software di qualità e privilegiando rilasci, il più possibile ravvicinati, che permettano al business di raggiungere in modo iterativo ed incrementale gli obiettivi di business prioritari al momento.

In particolare, la definizione del contenuto della prima Release di progetto è focalizzato al raggiungimento di un Minimum Viable Product (MVP – Insieme minimo di funzionalità che portano valore al committente) ovvero il minimo valore di business rilasciabile prima e nel minor tempo possibile: l'obiettivo è garantire un obiettivo di business, anche piccolo, con un Time to Market il più possibile ridotto.

All'avvio della Discovery è opportuno stabilire con il Service Manager di riferimento la più opportuna strategia di sourcing. In particolare, durante la Discovery può emergere la necessità di effettuare ricerche di mercato per individuare eventuali soluzioni o prodotti già sviluppati che soddisfino i requisiti della soluzione progettuale che si sta immaginando. Come detto è bene individuare al più presto i Fornitori coinvolti nello sviluppo software, in modo da valutare tempestivamente il loro coinvolgimento nei lavori di Discovery e del progetto.

Una volta indirizzati e condivisi gli argomenti della Discovery, in particolare i tempi e contenuti della prima Release, è necessario attivare la procedura per la stima dei costi della prima Release, attraverso una Poker Sizing Session.

Della prima Release verranno inoltre individuati i primi requisiti di dettaglio, da implementare nel primo Sprint di sviluppo.

Al termine della Discovery viene organizzato un incontro (cosiddetto incontro di fine Discovery) in cui viene presentato al team di progetto, comprensivo di tutte le strutture realizzatrici individuate e gli stakeholder del progetto, l'esito dei lavori di Discovery.

Chiusa la Discovery, inizia ufficialmente il progetto di sviluppo realizzativo con l'avvio della prima Release.

Per lo svolgimento degli incontri di Discovery lo Scrum Master si avvale delle tecniche agili:

- Impact Mapping;
- User Story Mapping;
- Slider;
- Elevator pitch;

La frequenza e la durata degli incontri di Discovery può variare in base al contesto del progetto. Gli incontri di Discovery sono facilitati dallo Scrum Master, supportato dal Coach Agile. Durante la fase di Discovery, il Coach Agile con il supporto dello Scrum Master, valuta la maturità agile del team e si adopera per colmare eventuali gap formativi, sia con interventi di formazione one-to-one sia verso l'intero team, in base alle esigenze.

Rischi, Vincoli, Dipendenze, Assunti

L'obbiettivo è identificare e analizzare eventi o situazioni che possono influire direttamente o indirettamente sul progetto, in modo da reagire nel modo più opportuno, minimizzando gli impatti negativi e massimizzando le opportunità. Tra gli obiettivi che la Discovery si pone c'è l'identificazione di rischi, assunti, vincoli e dipendenze del progetto.

Il team di progetto individua tali elementi inizialmente durante la Discovery e successivamente durante il corso del progetto. Gli elementi che emergono vengono analizzati e gestiti principalmente a cura del Capo Progetto, con il supporto del team di progetto e degli stakeholder coinvolti.

Nel team è bene prevedere un momento specifico con l'obiettivo di rivedere rischi, assunti, vincoli e dipendenze del progetto quantomeno nelle successive Inception, durante la definizione delle nuove Release.

Tuttavia anche nei momenti di Backlog Refinement e Sprint Review (Demo) possono emergere nuovi elementi da analizzare e gestire, contestualmente all'emergere di nuove funzionalità del prodotto o in generale in base all'andamento della realizzazione del prodotto e dei lavori progettuali. Gli incontri di scoperta, definizione e analisi di rischi, vincoli, dipendenze e assunti sono facilitati dallo Scrum Master.

Di seguito una descrizione per cosa si intende per rischi, vincoli, dipendenze e assunti.

Rischi

Il rischio è un evento incerto ma rilevante che può verificarsi e causare risultati inattesi o imprevisti. Le sue componenti fondamentali sono:

- L'incertezza ovvero uno scenario o situazione che non è né impossibile né certa;
- L'impatto ovvero una conseguenza ben definita sul progetto se l'evento incerto accade;

La conseguenza che il rischio genera può provocare effetti positivi o negativi sull'andamento del progetto. Un effetto positivo viene definito 'opportunità', mentre uno negativo 'minaccia'. Poiché esiste un aspetto di probabilità associato al rischio, la sua esatta occorrenza è sconosciuta, ma rientra nei limiti del progetto.

V*incoli*

I vincoli sono situazioni o eventi che possono ostacolare temporaneamente o definitivamente l'andamento di specifici step o il conseguimento in generale degli obiettivi di progetto. Questo concetto proviene dalla cosiddetta "Teoria dei Vincoli" nell'ambito della gestione organizzativa e aziendale e lo sviluppo software. In particolare, la disciplina Agile lo ha mutuato per spiegare come un vincolo (o collo di bottiglia) rappresenti la massima potenzialità di sfruttare le risorse di progetto finché non venga rimosso o bilanciato agendo sui fattori a disposizione del team di progetto.

Tipicamente all'interno di un progetto si possono osservare 6 categorie di vincoli di cui è opportuno rilevare e condividere ripetutamente gli impatti durante tutto il ciclo di vita, in modo da cercare delle soluzioni per rimuoverli o bilanciarli:

- Scope;
- Pianificazione;
- Budget/costi;
- Qualità;
- Persone;
- Rischi.

Dipendenze

Le dipendenze descrivono la relazione tra due o più attività sequenziali all'interno di un progetto o tra progetti correlati. Queste relazioni determinano:

- L'ordine in cui il team di progetto deve svolgere le proprie attività, organizzandosi al proprio interno e relazionandosi all'esterno;
- La priorità da assegnare nello sviluppo del prodotto alle singole componenti, distinte in mandatorie e discrezionali a seconda del valore rappresentato per il business.

Assunti

Un assunto rappresenta una tesi vera e auto-consistente, si tratta di un evento che ci si può aspettare che accada durante il ciclo di vita di un progetto e che per tale motivo è opportuno identificare e condividere col team. Tuttavia, la certezza dell'assunto non è supportata da prove fattuali ma proviene dall'esperienza. Proprio come rischi, vincoli e dipendenze, anche gli assunti sono eventi la cui occorrenza è fuori dal controllo del team di progetto. A differenza degli altri, gli assunti possono rappresentare situazioni con cui inevitabilmente doversi confrontare e, se ben gestiti, opportunità volte a conseguire gli obiettivi di progetto con successo. Tra gli assunti più importanti che è opportuno governare durante tutta la vita di un progetto agile, sono prioritari:

- Disponibilità delle persone: tutti i membri chiave del team di progetto sono disponibili e hanno le competenze e le conoscenze necessarie per lavorare al progetto;
- Disponibilità del budget: l'impegno economico deterministicamente previsto insieme al Business Sponsor continua ad essere preciso durante tutto il ciclo di vita e copre tutte le spese di progetto;
- Pianificazione iterativa e incrementale: i termini guidati dalla delivery e le tappe di progetto evolvono insieme al team di progetto e continuano ad essere sostenibili grazie alla gestione delle necessità via via che emergono, per rilasciare il valore concordato col Business nei tempi;
- Performance dei fornitori: tutto il necessario è a loro disposizione per erogare il massimo del valore nelle iterazioni pianificate e i loro progressi sono apprezzabili in cicli di feedback più brevi;
- Efficiente gestione delle escalation: per avere tutto il sostegno possibile da parte degli Sponsor di progetto, è opportuno prevedere un processo di gestione delle escalation commisurato col contesto di riferimento e con gli attori chiave.

Release (Fase 4)

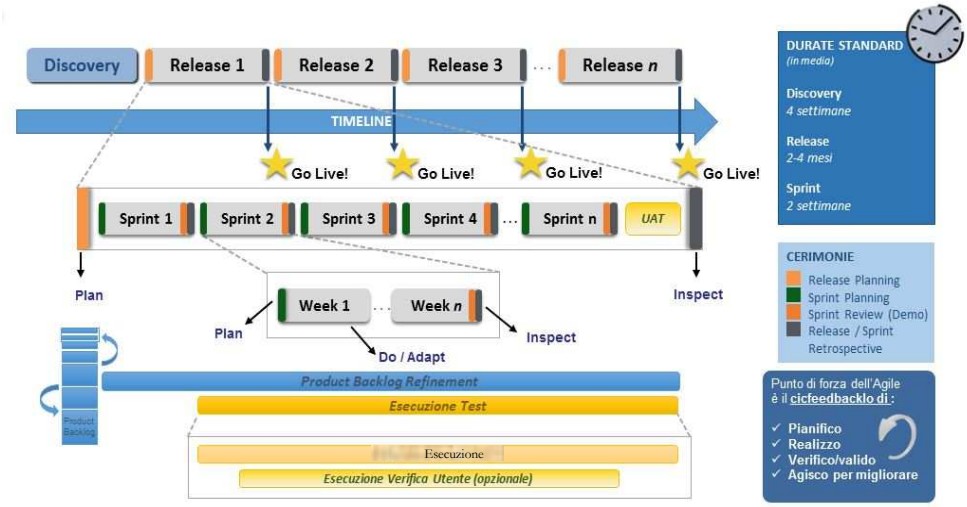
La fase 'Release' è un periodo temporale che racchiude le attività necessarie all'implementazione e al rilascio in produzione di un incremento di prodotto auto-consistente, nel contesto del progetto.

Il progetto realizzativo, attivato dopo le fasi di Avvio Progetto e Discovery, si compone di Release sequenziali, ovvero periodi temporali, anche di durata diversa, che si susseguono una dopo l'altra, mantenendo la stessa struttura di operazioni al proprio interno.

Ogni Release ha una durata di alcuni mesi (in genere 2 o 3) e corrisponde ad un rilascio in produzione consistente che produce un valore di business.

Durante la fase di Discovery (per la prima Release) o nella operazione di Inception (per le successive) il team di progetto, con gli opportuni stakeholder, analizza e condivide l'approccio progettuale, la durata e i contenuti della Release in termini di assegnazione di ruoli, partecipazione di contributori di specifiche componenti applicative, definizione dei requisiti (Temi ed Epiche).

La Release è suddivisa a sua volta in n iterazioni, ovvero periodi temporali sequenziali di durata fissa, denominati Sprint, in cui si implementa un incremento di prodotto (anche non auto-consistente) in ambienti non di produzione. All'interno della Release, dopo la sequenza degli Sprint, vengono realizzate le operazioni destinate al test utente o User Acceptance Test (UAT), al rilascio ed eventualmente al presidio post-rilascio.



Per poter attivare una Release è necessario realizzare alcuni passi preparatori:

- Stima delle Epiche ipotizzate per la Release, con Poker Sizing Session;

- Approvazione e sblocco del budget necessario;
- Attivazione dei Fornitori e formalizzazione degli appositi documenti contrattuali;
- Ingaggio dei Product Owner Area IT per le componenti applicative necessarie;
- Aggiornamento del Piano di progetto;

La Release inizia con la cerimonia di Release Planning in cui si condivide e consolida il piano della Release stessa. Durante la Release il team continua a raffinare ed eventualmente riposizionare i requisiti non ancora in sviluppo (Backlog Refinement), tracciandoli sul Product Backlog.

Mano a mano che il software viene sviluppato durante gli Sprint, vengono realizzate le attività di Test che normalmente si articolano durante l'intera Release, a partire dai primi sviluppi realizzati.

In ottica di continous improvement del team, a fine Release il team di progetto effettua la cerimonia di Release Retrospective per individuare azioni di miglioramento (cosa fare, rifare o evitare) da applicare nella Release successiva.

Al termine della Release, lo Scrum Master con il supporto del Coach Agile, attraverso le Survey Agile, raccoglie i feedback sul metodo utilizzato da parte del team di sviluppo, dei Fornitori, del Product Owner di Business e dello Scrum Master stesso.

In funzione delle esigenze di progetto, all'interno della Release è possibile effettuare rilasci intermedi (ad es. rilasci silenti, singole funzionalità auto-consistenti ecc.). In questo caso il team di progetto deve valutare i rischi associati al rilascio in produzione effettuato parallelamente ad altre attività di sviluppo in corso, garantendo le necessarie attività di test (System e UAT).

La gestione economica viene effettuata a livello di Release, nel quadro complessivo dell'intero Progetto.

Sprint (Fase 5)

Lo 'Sprint' è un periodo temporale di durata fissa, in cui viene sviluppato un incremento di prodotto dimostrabile. Lo Sprint è la componente base di una Release Agile, composta di n Sprint sequenziali.

Lo Sprint ha una durata prefissata (generalmente da 1 a 3 settimane) costante in tutta la Release.

In fase di pianificazione della Release, agli Sprint può essere associato uno Sprint Goal ipotizzato in quel momento, cioè un obbiettivo che riepiloga sinteticamente, anche solo con un titolo, gli argomenti che verranno trattati in quello Sprint. Esempi di Sprint Goal possono essere: consegna di specifiche funzionalità, completamento di un'epica, risoluzione di specifici difetti.

Lo Sprint Goal viene riesaminato nel corso del progetto e consolidato all'inizio dello Sprint. Lo Sprint inizia con la cerimonia di Sprint Planning in cui il team di progetto seleziona dal Product Backlog, in base all'obiettivo dello Sprint, gli item (di norma User Story) che verranno sviluppati durante lo Sprint, creando lo Sprint Backlog.

I contenuti dello Sprint vengono condivisi con il team di progetto e in particolare con i referenti dei Fornitori (se coinvolti) per la valutazione della sostenibilità.

Durante lo svolgimento dello Sprint, i contenuti degli item dello Sprint Backlog sono in fase di realizzazione (sviluppo e Unit Test) da parte del team di sviluppo e dunque non possono essere modificati. Mentre il team di sviluppo realizza il codice, il team di progetto, che include i Referenti Fornitori coinvolti, continua le attività di Backlog Refinement per gli Sprint successivi.

Durante lo Sprint, il team con il supporto dello Scrum Master può valutare la pianificazione di Weekly Meeting allo scopo di condividere aggiornamenti sull'andamento delle attività e su eventuali impedimenti riscontrati.

Durante lo Sprint è inoltre possibile condividere le informazioni sull'avanzamento delle User Story aggiornandone lo stato nel Product Backlog (es. da Planned a Doing) o su una Kanban board.

A fine Sprint, durante la cerimonia di Sprint Review (o Demo), viene mostrato l'incremento di prodotto funzionante, sviluppato durante lo Sprint (anche se non ancora auto-consistente), al Product Owner di Business e a tutto il team per raccogliere i feedback su quanto realizzato.

L'incremento di software sviluppato nello Sprint risiede in un ambiente precedente alla produzione, di norma l'ambiente di 'System'.

L'output dello Sprint che viene accettato dal Product Owner Business è pronto per essere testato a cura dei tester, coordinati dal Test Manager. Durante lo Sprint viene quindi generalmente testato l'output degli Sprint precedenti.

Lo Sprint si conclude con la Sprint Retrospective, la cerimonia in cui il team di progetto riflette sull'andamento dello Sprint appena trascorso. L'obiettivo della cerimonia è individuare possibili azioni di miglioramento da applicare nello Sprint successivo.

Durante lo Sprint è possibile valutare e condividere integrazioni al contenuto dello Sprint Backlog, aggiungendo User Stories da portare in Review, se la capacity del team di sviluppo lo consente, nel rispetto del principio di sostenibilità.

Durante lo Sprint, il team di sviluppo lavora alla realizzazione delle User Story pianificate nello Sprint, mentre il team di progetto, che include i referenti fornitori del software, rifiniscono le User Story da pianificare nello Sprint successivo.

A fine Sprint è bene valutare l'avanzamento della Release, in termini di contenuti e costi, utilizzando i grafici burn-up chart e burn-down chart..

Gli Sprint sono di durata fissa nella Release e, auspicabilmente, nell'intero progetto.

Fine Progetto (Fase 6)

La fase di Fine progetto è composta dalle attività conclusive del progetto.

Il Product Owner Area IT e il Capo Progetto con il supporto del team di progetto verifica la conclusione delle attività progettuali, inclusa l'eventuale fase di follow-up seguita al rilascio della soluzione progettuale.

Il Product Owner Area IT e il Capo Progetto con il supporto del team indirizza eventuali temi rimasti da gestire (punti aperti, problematiche, rischi ecc.) e, a situazione stabilizzata, sancisce l'avvenuto passaggio della soluzione dalla fase progettuale alla gestione ordinaria di manutenzione.

A questo scopo, il Capo Progetto valuta l'organizzazione di un Meeting di chiusura con gli stakeholder del progetto per condividere principalmente i seguenti argomenti:

- Obiettivi raggiunti, risultati e deliverables;
- Investimenti sostenuti in termini di tempi e costi ed eventuali scostamenti;
- Eventuali punti aperti, problematiche e rischi da indirizzare e gestire;
- Ufficializzazione del passaggio alla gestione ordinaria;
- Chiusura dei lavori;

Inoltre, lo Scrum Master con il supporto del Capo Progetto invita il team ad una Retrospective di progetto, per raccogliere le lesson learned dal progetto appena concluso.

Al termine del progetto, se l'area di business è esterna all'Area IT, il Service Manager di riferimento raccoglie il feedback dell'utente sul progetto complessivo, attraverso la Customer Satisfaction Agile.

A differenza delle Retrospective effettuate durante il progetto (es. Sprint, Release), quella finale ha lo scopo di individuare le azioni di miglioramento che potranno essere applicate in progetti futuri.

Per chiudere il progetto, anche ai fini operativi della reportistica aziendale, il Capo Progetto, con il supporto del team, deve attivare gli opportuni interventi previsti nella Checklist di Chiusura progetto.

Metodo Agile funziona?

La metodologia Agile, nata inizialmente per lo sviluppo di software, è largamente utilizzata per i benefici che riguardano le interazioni nel processo di sviluppo, la collaborazione necessaria nella fase di ingegnerizzazione, il contatto con il cliente o utente finale. L'obiettivo è sempre stato la ricerca di un modello per la pianificazione e l'implementazione di progetti che specificatamente potesse adeguarsi alle singole necessità.

Un approccio strutturato sull'analisi di questo strumento di supporto alla produzione dovrebbe analizzare criticamente i benefici e valutare concretamente il risultato migliorativo ottenuto. Se si volesse valutare per parametri questi benefici si potrebbe iniziare a discutere alcuni temi come efficienza dell'approccio, la soddisfazione degli utenti coinvolti, il rispetto degli obbiettivi organizzativi e quello che può essere definito come il team experience e per concludere percentuale di successo/insuccesso.

L'attenzione rivolta alle performance del progetto a volte oscurano l'attenzione da porre sulla percentuale di successo. Si può concordare sicuramente nel definire di "successo" un progetto che si sia svolto nei tempi, coti e abbia raggiunto gli obbiettivi posti, ma come questo può essere considerato nel caso in cui gli obbiettivi non siano in linea con le necessità del cliente finale? Un esempio può essere Kodak, la multinazionale specializzata nella produzione di apparecchiature per immagini, che non investendo su prodotti digitali è andata incontro a un netto declino della sua presenza sul mercato. Sicuramente negli anni la Kodak ha intrapreso progetti di sviluppo condotti e conclusi con successo, ma una scelta strategica sbagliata e una poca attenzione ai desideri del mercato non gli ha permesso di cogliere quanto il mondo della fotografia digitale stava sostituendo quello tradizionale.

La metodologia Agile nel panorama dello sviluppo software è in contrasto con il più classico approccio Waterfall: obbiettivi flessibili, interazioni con il cliente, congelamento del software il più tardi possibile, un approccio incrementale piuttosto che un processo standardizzato. Un approccio più standard utilizza come strada per il raggiungimento del risultato una sequenza logica di step predeterminati in un piuttosto rigido processo di sviluppo con potenziali conseguenze su: soddisfacimento del cliente e flessibilità.

Una domanda da porsi potrebbe essere: ma davvero i progetti Agile sono più di successo?

Ebbene, partendo dal presupposto che qualsiasi deviazione progettuale può nascondere rischi, le deviazioni all'interno di un piano di progetto non

possono essere evitate, sia che il progetto segua un approccio tradizionale o evolutiva a cicli multipli. Non potendo evitare tali deviazioni quello su cui si può intervenire è la metodologia di gestione delle conseguenze derivanti da questa deviazione. Se si considera che in alcuni casi si arriva ad avere che 50% delle attività di design avviene in fasi diverse da quella di design, è evidente come non sia possibile blindare la fase di design appena essa viene conclusa nelle prime fasi del progetto bensì è opportuno prevedere un approccio che tenga conto di queste modifiche e le gestisca si come eccezioni ma senza impatti critici.

Lo sviluppo deve concentrarsi su alcuni punti cardini:

- Gli individui e le interazioni nel processo;
- Collaborazione del cliente;
- Risposta ai cambiamenti.

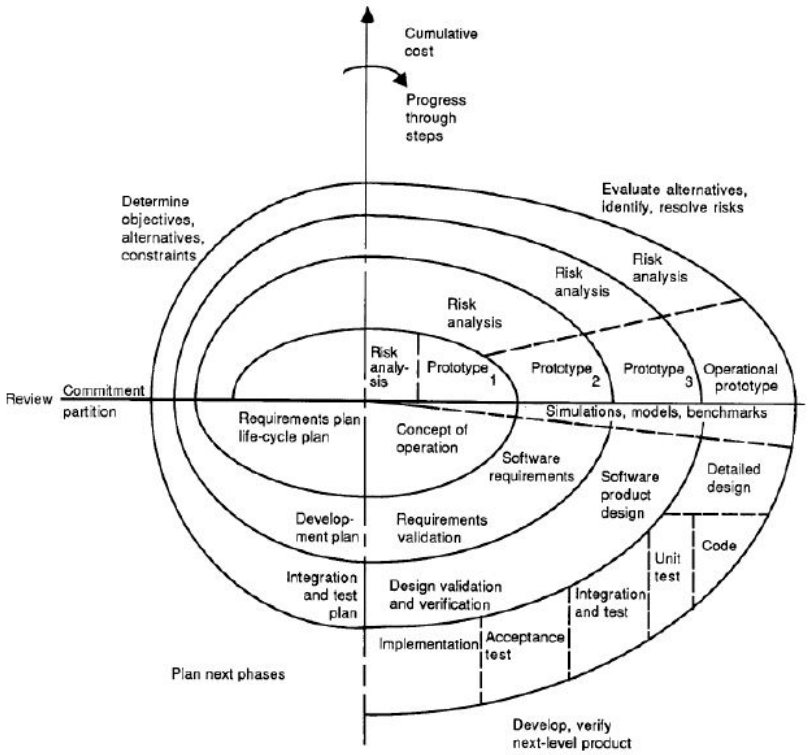
L'utilizzo di documentazione deve essere ridotta al minimo in modo da non pregiudicare la complessità e reattività ai cambiamenti. Ebbene i cambiamenti sono la sfida che deve essere supportata adeguatamente, e riguardano generalmente tre ambiti:

- Obbiettivi;
- Risorse e strumenti;
- Relazione con altri progetti e prodotti.

Bohem nel 1996 identifico in questi cambiamenti quelli che maggiormente si riscontrano nello sviluppo di progetti IT ed in particolare identifico alcune conseguenze di una documentazione troppo dettagliata:

- Dettagli superflui, che non descrivono il prodotto da rilasciare come prototipo, e quini poco utili.
- Ricchezza di contenuti superflui, perché non conoscendo il prodotto si tenta di inserire il "più possibile" nell'unico istante in cui possono essere descritte le specifiche.
- Mancanza di visibilità, ci si focalizza sulla conoscenza e visione attuale non prevedendo eventuali cambiamenti che potrebbero occorrere nel corso del progetto.

Bohem propone inoltre un approccio ciclico mostrato nella figura di seguito dove ogni attività viene ripetuta nel tempo e con diversi livelli di dettaglio.



È utile sottolineare che il metodo Agile non tralascia e non critica la pianificazione anticipata e complessiva, ma il concetto fondamentale su cui ci si concentra è sul rendere questa pianificazione non un ambiente immutabile, bensì strutturare il processo di sviluppo in modo che si possano prevedere interazione e cambiamenti derivanti da influenze esterni. In uno studio condotto da Koskela e Abrhaamsson nel 2004 si è rilevato come il 42% del tempo in un progetto con metodologia tradizionale si sia impiegato nella pianificazione tanto che si può arrivare ad ottenere che l'impatto negativo risultante dal troppo tempo speso per la pianificazione è uguale all'effetto negativo che si otterrebbe nel caso di scarso impegno in questa attività di pianificazione.

Punto essenziale e chiave di svolta è la comunicazione con i clienti, tanto che da uno studio condotto nel 2015 è emerso che incontri e riunioni di allineamento aiutano a ridurre la "confusione" sulle attività da svolgere e nell'ambito IT le parti da sviluppare.

Analizzando per mezzo di un confronto per punti è possibile elencare alcune differenze presenti tra una metodologia Agile e una tradizionale.

	Metodologia Tradizionale	Metodologia Agile
Approccio di base	Il processo è analizzato con estrema specificità, la pianificazione condotta al dettaglio	Gli sviluppi seguono una pianificazione di più alto livello che a sua volta segue una logica di continui miglioramenti e test in linea con i cambiamenti esterni
Gestione	Stretto controllo e indirizzamento in base al piano	Comunicazione e scambio di informazioni tra i vari team
Comunicazione	Formale	Informale
Modello di sviluppo	A blocco in base al ciclo di vita	Basato sui rilasci (numerosi e intermedi)
Struttura organizzativa	Complessa	Organica e flessibile, utile alla gestione cooperativa
Controllo della Qualità	Avviene a posizioni avanzate che richiedono un considerevole sforzo per il test	Controlli continui e intermedi

Successo del Progetto

È necessario a tale riguardo definire preliminarmente cosa può essere definito un successo e cosa no. Normalmente la misura del successo si basa sul prodotto e come questo possa essere di sufficiente qualità tanto da poter produrre il beneficio e il risultato atteso o desiderato. Sorge a questo punto normale concordare su quale debba essere il punto in cui deve essere fatta questa misura. Tradizionalmente il lavoro del Project Manager si conclude quando il progetto è consegnato al cliente e viene tralasciata la parte più ampia ma naturalmente consecutiva dell'utilizzo del prodotto. Esiste una lacuna, che nel processo evolutivo è stata colmata, definendo per successo quel progetto che rende sodisfatto il cliente, sostenendo che può essere definito di successo un progetto che sebbene non abbia rispettato gli obbiettivi iniziali abbia reso il cliente soddisfatto. L'identificazione degli obbiettivi può creare delle distorsioni nel caso in cui gli obbiettivi iniziali non siano rispettati e il cliente finale sia soddisfatto, questo può essere evitato solo attraverso un processo attento a inizio del progetto nella stesura

del documento di scope, tuttavia questa circostanza può accadere molto più probabilmente nel caso in cui utilizzando un approccio tradizionale, l'utente non viene coinvolto. Il caso contrapposto è quello in cui gli obbiettivi vengono raggiunti ma il cliente non è soddisfatto.

Certamente la definizione di un progetto di successo passa per la valutazione di tre dimensioni: Tempi, costi e obbiettivi. L'obbiettivo come si è visto ricopre parte fondamentale nella definizione di successo, ma normalmente abbiamo anche:

- Efficienza del Progetto = raggiungimento dei costi, degli obbiettivi e dei tempi prefissati
- Successo degli stakeholder = soddisfacimento delle aspettative derivanti dal progetto (la migliore definizione di successo).

Una valutazione più ampia di progetto di successo può essere indirizzata svolgendo un'analisi su tutte le fasi di sviluppo analizzando valore e sprechi. In questa ottica un progetto di successo può essere definito come una serie ordinata di attività volte al raggiungimento degli obiettivi, pianificando attività che aggiungono valore con la loro esecuzione e che riducono (o monitorano) eventuali sprechi che possono verificarsi. Con un overview su quali sono le barriere che ostacolano l'aggiunta di valore possiamo osservare che potenziali barriere riguardano:

	Attività	**Implicazioni**
1	Cambi di scope continui	Delivery non chiare e definite
2	Requisiti non dettagliati a sufficienza	Mancanza di una visione d'insieme/tempi
3	Limitata capacità di rilascio e prioritizzazione	Tempi
4	Aspettative non realistiche degli stakeholders	Stress e alti carichi di lavoro
5	Problemi tecnici negli ambienti di test	Spreco di risorse
6	Difficile individuazione del valore	Costi
7	Utente con necessità non chiare	Spreco di risorse

Qualità tra monitoraggio e miglioramento

Nell'ambito dell'approccio Agile, l'approccio di sviluppo scelto e quello di DevOps. Il DevOps è un approccio allo sviluppo del software che abilita la collaborazione tra i team in termini di Development, Operation e Quality Assurance. La finalità principale è la distribuzione continua, rapida e frequente del software, avendo come obiettivo aumentare l'affidabilità, la resilienza, la stabilità e la sicurezza degli ambienti di produzione, abbattendo la "barriera" tra i team di sviluppo e i team di operation. L'approccio DevOps è applicabile sia in progetti di sviluppo software in metodo Waterfall che Agile. In particolare l'approccio DevOps è complementare al metodo Agile perché facilita i processi di integrazione e rilascio applicativo assicurando la "consegna" di nuove funzionalità a valore aggiunto al termine di ogni Sprint.

L'approccio DevOps si sviluppa attraverso quattro ambiti, per ognuno dei quali vengono proposte delle soluzioni e/o servizi:

- Continuous Integration;
- Continuous Testing;
- Continuous Delivery;
- Continuous Monitoring;

Continuous Integration

La Continuous Integration è il processo che consente ai diversi team di sviluppatori di lavorare simultaneamente seguendo la metodologia Agile o Waterfall, validando costantemente le modifiche che vengono apportate al codice. Assicura che il software prodotto da diverse fonti sia consistente, frequentemente integrato ed abilitato a gli sviluppi paralleli. L'applicazione della Continuous Integration consiste nell'esecuzione dei cosiddetti Continuous Integration Script all'accadere di eventi predefiniti che prevedono l'attuazione di una serie di operazioni automatiche quali: l'esecuzione dei test unitari automatici, la compilazione ed il deploy. Al termine dell'esecuzione vengono prodotti i report per il team di sviluppo contenenti:

- I log prodotti durante la fase di build;
- L'esito del deploy;
- I risultati dei test unitari;
- L'indice di copertura dei test sul codice;
- Le eventuali violazioni delle 'Regole di buona programmazione';

Continuous Testing

Il Continuous Testing è il processo che consente l'esecuzione di test predefiniti (funzionali, di performance, di sicurezza, etc.) in un ambiente con frequenza elevata o continuativa, mano a mano che il software viene sviluppato e/o rilasciato. Il processo ha l'obiettivo di ridurre i tempi ed i costi relativi all'esecuzione dei test, per farlo sono presenti diversi servizi:

- Service Virtualization per servizi esterni alle applicazioni:

  La Service Virtualization consente di simulare il comportamento di una componente software, abilitando la possibilità di anticipare il test end-to-end di una applicazione soggetta a change già negli ambienti di test a monte.

  La Service Virtualization consente di effettuare i test richiamando servizi virtuali invece che produttivi/reali esistenti (normalmente disponibili a valle o solo in produzione) ma anche simulando i servizi non ancora disponibili in fase di sviluppo/test. In questo modo, è possibile verificare l'integrazione tra alcune componenti software in fase di caricamento il prima possibile, frequentemente e continuamente quando serve. Lo strumento utilizzabile per il servizio di Service Virtualization è un applicativo commerciale e permette di virtualizzare servizio come Web Services, Business Services e Chiamate DB.

- Test Automatici:

  Il modello condiviso Agile-DevOps prevede, per la componente di Continuous Testing relativa ai test automatici, di elevare il livello di automazione del Testing in ambiente di System con l'obiettivo di definire i casi di test, di eseguire test automatici di non regressione.

- Mobile Testing Automation:

  Nell'ambito del Continuous Testing, il Mobile Testing Automation consente il test delle applicazioni mobile e la gestione efficiente e centralizzata dei dispositivi fisici (smartphone/tablet) e delle versioni delle App.

Continuous Delivery

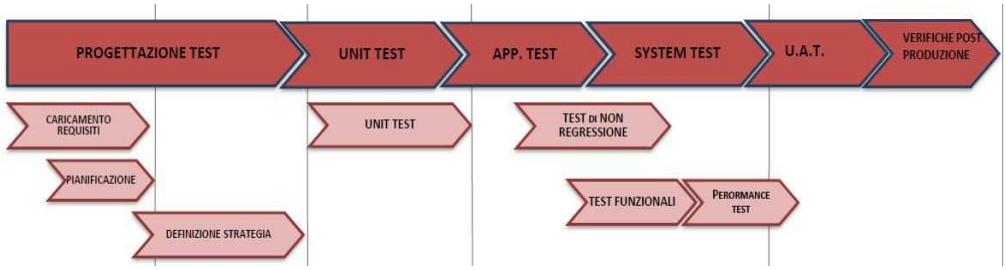
Il Continuous Delivery è il processo che prevede l'automazione del rilascio del software in tutti gli ambienti previsti: dall'ambiente di Sviluppo fino a quello di Produzione. Il processo copre anche le esigenze di gestione delle configurazioni, refresh dei dati (per gli ambienti di test) ed esecuzione automatica dei test. È considerato il processo più critico e fondante per l'adozione delle logiche DevOps.

Continuous Monitoring

Il Continuous Monitoring è il processo che prevede di fornire dati e metriche (alle fasi precedenti del ciclo di vita):

- Sulle applicazioni lungo tutti gli stati evolutivi (dal Requirement fino alla funzionalità rilasciata);
- Sull'uso delle applicazioni in Produzione da parte degli Utenti Finali;
- Sulle funzionalità, le modalità d'uso e le performance;
- Sul processo e sugli Ambienti di test.

Per quanto riguarda la fase di test sono previsti livelli successivi di test in base alla fase del ciclo di vita del software in cui ci si trova: Unit Test, Application Test, Integration Test, System Test e User Acceptance Test (UAT).



Gli **Unit Test** hanno lo scopo di testare singoli moduli, senza interazione con altri moduli o altri sistemi, con basi dati e casistiche "forzate". Gli Unit Test hanno l'obiettivo di verificare la stabilità e la conformità alle specifiche stabilite in fase di analisi, a livello di singolo modulo sviluppato. Lo Unit Test si articola in due passi il primo consiste nella verifica del corretto funzionamento delle singole componenti sviluppate rispetto ai requisiti di progetto definiti e secondo consiste nell'individuazione e correzione di eventuali anomalie. Gli Unit Test vengono eseguiti nella macchina dello sviluppatore, in modalità stand-alone e le eventuali anomalie vengono individuate e corrette direttamente dallo Sviluppatore.

Gli **Application Test** hanno lo scopo di verificare che l'intera applicazione, in modalità stand-alone, cioè senza l'integrazione con tutte le altre applicazioni e sistemi esterni, sia sufficientemente stabile e risponda ai requisiti di progettazione. Rispetto all' Integration Test, l'applicazione è più

stabile e completa. Gli Application Test vengono eseguiti in ambiente "Application" dove potrebbero essere disponibili alcune integrazioni con applicazioni esterne. Le eventuali anomalie individuate e corrette direttamente dallo sviluppatore o segnalati tramite il Test Manager. Il processo per la gestione delle anomalie dipende da quanto stabilito dal progetto per l'Application Test. L'esecuzione dell'Application Test prevede il completamento delle seguenti attività:

- Attribuzione dei casi di test alle specifiche risorse del team;
- Esecuzione dei casi di test assegnati;
- Registrazione dell'esito dei test e delle eventuali anomalie riscontrate;
- Correzione delle anomalie ed esecuzione dei test per la verifica del corretto funzionamento dell'applicazione;
- Valutazione dell'esito complessivo della fase;

In caso di esito positivo dell'attività, il TM provvede a formalizzare il superamento della fase di Application test e il superamento di questa fase viene seguita dalla richiesta di promozione in ambiente di "System", dove verranno eseguite le successive fasi di test.

Gli **Integration Test**, test di livello successivo rispetto allo Unit Test, hanno lo scopo di verificare la corretta interazione fra i singoli moduli sviluppati, interni ad una singola funzionalità dell'applicazione. In questa fase vengono verificate le interfacce e i flussi di comunicazione fra i singoli moduli. Gli Integration Test vengono eseguiti generalmente nella macchina dello sviluppatore o nell'ambiente di sviluppo dell'applicazione. Le eventuali anomalie vengono individuate e corrette direttamente dallo Sviluppatore. Durante gli Integration Test si verifica la corretta integrazione e comunicazione fra i diversi moduli rispetto ai requisiti di progetto ed al relativo disegno tecnico e si individuano e correggono di eventuali anomalie.

Il **System Test**, ha lo scopo di verificare la corretta interazione tra tutte le applicazioni coinvolte nella soluzione, e che l'applicazione risulti sufficientemente stabile e rispondente ai requisiti funzionali anche dopo l'integrazione all'interno del restante ambiente applicativo. In questa fase vengono verificate tutte le tipologie di interfacce esterne tra le applicazioni coinvolte, sia quelle sviluppate ex-novo dal progetto, sia quelle esistenti che hanno implementato nuove interfacce di comunicazione. Il System Test viene eseguito in un ambiente dedicato molto simile alla produzione, dove è garantita l'integrazione con le applicazioni e i sistemi esterni necessari al completamento del System Test. Le eventuali anomalie vengono generalmente individuate da parte di un Tester e segnalate tramite il Test Manager. Il System Test è caratterizzato da attività di:

- Organizzazione temporale e procedurale delle attività;
- Progettazione ed assegnazione dei test-case da eseguire;
- Esecuzione dei test-case e raccolta delle eventuali anomalie;

- Gestione delle eventuali anomalie emerse (correzioni sul software e nuova esecuzione dei test case).

Il superamento della fase di System Test è legata al raggiungimento delle soglie sui KPI concordati per il passaggio alla fase di test successiva. L'esito positivo delle attività di System Test consente il passaggio alla fase dei User Acceptance Test (UAT).

Gli **User Acceptance Test (UAT)**, hanno lo scopo di verificare, prima del rilascio in produzione, che l'applicazione risponda alle aspettative e ai requisiti condivisi con l'utente committente Product Owner Business. La fase UAT viene attivata dopo il superamento positivo del System Test e ha l'obiettivo di validare, da parte dell'utente, il software realizzato. A seguito dell'approvazione utente, può essere eseguito il passaggio del software in produzione. La progettazione e la successiva esecuzione dello User Acceptance Test possono essere svolte in due differenti modalità, in base a quanto concordato nel progetto e quindi direttamente da parte del Business Owner o da parte di una funzione aziendale delegata dal Business Owner. Il Test Manager verifica che l'ambiente per il collaudo utente soddisfi tutte le esigenze di chi effettuerà l'attività di User Acceptance Test (UAT). La fase UAT è caratterizzata dalle seguenti attività:

- Progettazione ed assegnazione dei test-case da eseguire;
- Predisposizione dei dati di input corretti, delle profilazioni e degli ambienti;
- Esecuzione dei test-case e raccolta delle eventuali anomalie;
- Gestione delle eventuali anomalie emerse (correzioni sul software ed esecuzione dei test case);
- Monitoraggio dello stato di avanzamento e delle eventuali criticità e condivisione con il team di progetto;
- Certificazione e accettazione software.

In caso di incompletezza del test o di inadeguatezza degli esiti, va definito un piano di contingency che ne consenta il rilascio in produzione.

Nell'ambito dell'esecuzione dei test quello che si può verificare è l'insorgere di errori dovuti ad un'azione umana che produce un risultato non corretto (errore nella scrittura del software), guasti o bug ovvero manifestazioni di errori nel software, fallimenti ovvero deviazioni del comportamento del software da quanto atteso. È opportuno sottolineare che il fatto di non trovare difetti con il testing non implica automaticamente che il software sia corretto in quanto il test riduce la probabilità che si manifestano difetti non trovati ma non è possibile effettuare un test omnicomprensivo, ma per ridurre i rischi è opportuno avviare la fase di test già nelle fasi iniziali del progetto. Oltre alla classificazione dei test vista in base alla fase del ciclo di vita del software esiste una distinzione rispetto alla particolare caratteristica da verificare:

- Test Funzionale con cui si verifica che i singoli requisiti funzionali siano stati soddisfatti, tipi i test che possono essere fatti sono quelli di sicurezza che verifica le funzionalità in relazione alla rilevazione di minacce come virus e accessi incontrollati o test di non regressione con cui si verifica che l'applicazione, relativamente alle componenti non coinvolte nella modifica, continui a funzionare correttamente a fronte di modifiche proprie e di componenti con cui interagisce;
- Test Non Funzionali con cui si verifica che i singoli attributi di qualità siano stati soddisfatti come test di prestazione e di carico con cui si misura il grado di erogazione delle funzionalità con vincoli di tempo di risposta in relazione a determinati carichi di lavoro, test di usabilità con cui si determina il grado con il quale il software viene compreso o piacevolmente usato, test di affidabilità con cui si misura la capacità del software di eseguire correttamente le funzionalità richieste per un specifico tempo o per uno specifico numero di operazioni, o ancora test di robustezza con cui si verifica il grado di robustezza di un componente ovvero quanto continua a funzionare correttamente in presenza di input invalidi o condizioni ambientali di stress.

Nell'attenzione alla qualità la Product Risk Analysis è l'analisi per determinare il rischio a cui ci si espone nel caso in cui un requisito non venga soddisfatto completamente o in parte. Il rischio è legato alla specifica funzionalità del prodotto. Questa analisi viene effettuata tramite un'intervista al Business Owner per comprendere il grado di rischio dei requisiti di soluzione individuati. Questo approccio supporta il calcolo di copertura dei requisiti rispetto ai test effettuati e di conseguenza si determina la quantità di rischio residuo all'uscita dai test. Come deve essere classificato la mancanza di un requisito (o il fallimento del test relativo)? Bisogna definire le priorità, valutando cosa è strategico nell'ambito dei test e delle funzionalità. Semplificando e ricorrendo a un esempio relativo a un mutuo rivolto alle giovani coppie, è necessario capire quale è il rischio di tener fuori ad esempio la parte relativa alla chiusura del mutuo che presumibilmente servirà dopo qualche anno dall'apertura del mutuo e la parte relativa all'apertura ed erogazione del mutuo evidentemente questi ultimi due requisiti hanno una priorità maggiore.

Un altro passo essenziale per monitorare l'andamento del progetto e il suo sviluppo in linea con le aspettative del cliente è raccogliere il suo grado di soddisfacimento. Per fare questo si attua un'operazione che consiste nella raccolta del feedback del committente di business sul progetto complessivo, questo avviene attraverso un'intervista di **Customer Satisfaction.**

Il questionario riguarda una serie di domande sia sui contenuti rilasciati, sia sulla gestione progettuale nel suo complesso. In particolare vengono

poste tre domande specifiche sul metodo Agile, aggiuntive rispetto al questionario Waterfall:

- Ritieni che durante il progetto sia stato possibile variare i requisiti per ottenere il prodotto/servizio giusto?
- Ritieni che le Sprint Demo abbiano consentito di dare feedback per migliorare iterativamente il risultato?
- L'impegno investito è stato commisurato al valore generato dal progetto?

I risultati della Customer Satisfaction Agile sono confrontabili con quelli della Customer Satisfaction dei progetti Waterfall.

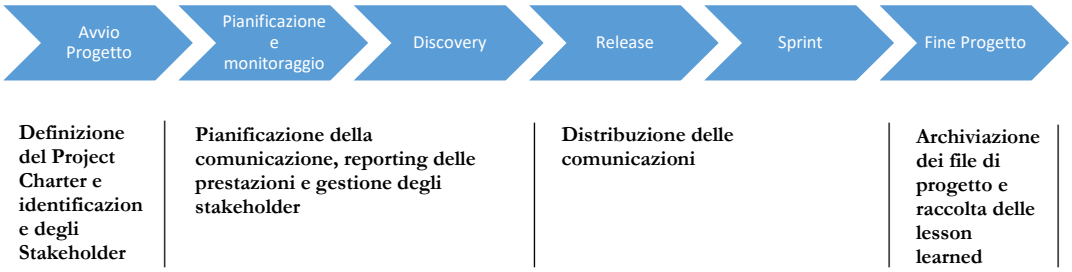
La comunicazione un processo essenziale

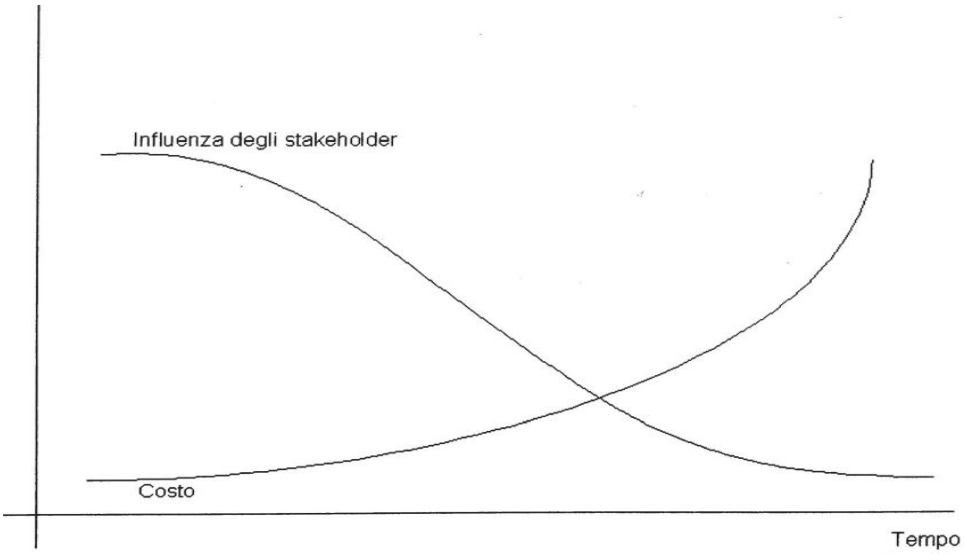
In contesti complessi e articolati dove spesso vanno ad evolversi i progetti IT un requisito essenziale per la riuscita del progetto è il coordinamento e la comunicazione tra tutte le persone coinvolte. È usuale concordare su alcune caratteristiche principali che definiscono la figura professionale del capo progetto come un vero e proprio manager e leader, dando un a particolare evidenza ai temi della comunicazine e dell'integrazone di tutti gli attori sociali coinvolti al fine di promuovere con e tra di loro le indispensabili basi relazonali che consentono uan proficua collaborazione. Allo stesso modo nel contesto attuale il team è sempre meno "fisico" e sempre più "logico", ovvero i componenti stanno perdendo in buona parte la caratteristica di essere collocati e assegnati al progetto in modo univoco. Nella maggior parte dei casi i gruppi di lavoro sono composti da persone che operano in più progetti contemporaneamente a volte prestando il proprio contributo per pochi giorni o per specifiche attività, senza avere mai l'occasione di incontrare di persona né il capo progetto né gli altri componenti del team. Un'altra caratteristica sempre più usuale è che per le caratteristiche tecnologiche e specialistiche che contraddistinguono molti progetti, il team non è più composto prevalentemente da esecutori ma sempre più da figure professionali che hanno specifici skills e una visibilità aziendale pari se non superiore a quella del capo progetto. Questa caratteristica ha favorito la tendenza degli specialisti a stare nel proprio ambito e non contaminarsi con l'ambiente circostante, ossia con funzioni specializzate limitrofe e partecipanti allo stesso progetto. Questa situazione pone in evidenza il ruolo che il capo progetto deve a volte esercitare nei confronti delle risorse assegnate ogni volta che le stesse devono comunicare con altri, con le inevitabili ripercussioni sull'efficienza complessiva. I progetti dunque sono realizzati da persone e il fattore umano rappresenta sempre l'elemento determinante per il successo di ogni progetto in qualunque organizzazione, settore di mercato e geografia esso venga intrapreso. Le persone devono potersi concretamente relazionare tra loro per giungere ad una proficua collaborazione e in ciò, come già detto, la comunicazione ha un ruolo determinante. Comunicare significa coinvolgere i vari attori del progetto in modo da renderli parte attiva, significa anche informare ed essere informati su motivi e contenuti del progetto e sul suo aggiornamento.

Una buona comunicazione deve essere organica, chiara e concisa. La comunicazione però non è da intendersi come solo mezzo di scambio di informazioni sull'andamento del progetto, ma anche la gestione dei documenti, la raccolta dei lesson learned, le tecniche di comunicazione e di gestione dei dati, cosi come l'individuazione e la gestione degli stakeholder dal punto di vista comunicativo. Un progetto con un forte controllo sui costi, sui tempi e sulla qualità senza un efficace processo di comunicazione con molta probabilità si troverà a dover affrontare problemi come il disallineamento tra gli stakeholder.

È chiaro che le informazioni e la condivisione degli stessi avviene a seguito di riunioni e report. Tuttavia un punto necessario da chiarire a priori e prima che venga attuata e pensata una strategia comunicativa efficace è la definizione dei fabbisogni informativi degli stakeholder, in altre parole comprendere i fabbisogni e i desiderata degli stakeholder definendone i canali di comunicazione che devono essere utilizzati.

La pianificazione della comunicazione è un qualcosa che viene dall'esperienza quotidiana per cui quando vogliamo comunicare qualcosa a qualcuno pianifichiamo quello che volgiamo dire. Una volta definita l'informazione che si vuole fornire o richiedere, il passo immediatamente successivo è l'individuazione dell'interlocutore e l'individuazione dell'approccio cui rivolgersi all'interlocutore e infine il momento migliore. Gli stakeholder sono sicuramente tra i più importanti interlocutori con cui instaurare una efficace comunicazione in quanto gli stakeholder primari sono i portatori di un interesse nell'attività in esecuzione, più in generale sono gli individui che posso influenzare o essere influenzati dall'attività di organizzazione in termini prodotti e processi lavorativi. Gli stakeholder però sono prima di tutto persone coinvolte attivamente nel progetto e che sono chiamate a prendere specifiche decisioni. Tralasciando per un attimo la differenza tra approccio Agile e approccio Tradizionale, non essendo in questo caso una differenza particolarmente impattante, lo stakeholder esercita un'influenza sul progetto. L'influenza esercitata ha una diretta ripercussione sui costi del progetto: sarà massima nella fase iniziale e diminuirà progressivamente con l'avvicinarsi della conclusione. L'impatto sul costo delle modifiche è all'inizio del progetto, poiché nella fase di pianificazione le modifiche possono essere gestite senza incidere gravemente sui costi, aumenterà progressivamente all'avvicinarsi delle fine, poiché, con molta probabilità comporterà rilavorazioni e ripanificazioni con un impatto potenzialmente maggiore sui costi. È giusto sottolineare tuttavia che questo principio ha una validità ragionevole e generale, la differenza tra l'approccio Agile e quello tradizionale può risultare nella distribuzione del grafico riportato di seguito, nel caso specifico questo grafico mostra le attività relative alla comunicazione previste in tutte le varie fasi del progetto:





Gli stekeholder coinvolti sono numerosi e vari, la loro classificazione avviene in funzione dell'interesse e del potere che possono esercitare sul progetto. Il potere sul progetto vien esercitato in base alla posizisone gerarchica ricoperta e alla competenza tecnica. Il livello d'interesse è invece dato dal gardo di incidenza che il progetto esercità sulla realtà dello stakholder. È possibile dunque in base a questa differenza tra interesse e potere costruire una tabella che illustri questa suddivisione posizionando:

- Stakeholder Primari, i quali hanno una relazione formale con la società come (team di progetto, Capo Progetto, Sponsor);
- Stakeholder Secondari, il quale contributo non è essenziale;
- Stakeholder Istituzionali, i quali forniscono un contributo di supporto e di controllo aziendale (ufficio legale, ufficio acquisti);
- Stakeholder Operativi, i quali coinvolti in modo significativo all'interno del progetto non hanno però un potere, è il caso fornitori e consulenti;

Potere Alto	Istituzionali	Primari
Potere Basso	Secondari	Operativi
	Interesse Basso	**Interesse Alto**

Per quanto riguarda invece la frequenza dell'aggiornamento e la profondità della relazione, varia in modo similare per quanto visto con il potere e l'interesse:

- Funzionali, la comunicazione deve essere ben curata, per ciascuno stakeholder la comunicazione dovrebbe essere accurata, continuativa e

personalizzata. L'obbiettivo è di mantenere sempre aggornati questi stakeholder in maniera da evitare blocchi o ritardi nella comunicazione;
- Informati, sono gli stakeholder che devono essere aggiornati sistematicamente sugli aspetti salienti del progetto con una comunicazione sintetica e immediata;
- Performer, sono gli stakeholder istituzionalu, necessiatno di un'informazione chiara e concisa, il loro ruolo funzionale è essenziale nelle fasi critiche del progetto in cui è necessario ricevere il loro appoggio;
- Attivi, sono evidentemente gli stakeholder operativi, oltre ad essere opportunamente informati, soprattutto nelle fasi iniziali del progetto è necessario tenere conto del loro punto di vista.

Frequenza Alta	Funzionali (Istituzionali)	Performer (Primari)
Frequenza Bassa	Informati (Secondari)	Attivi (Operativi)
	Profondità Bassa	**Profondità Alta**

In progetti di grandi dimensioni e caratterizzati da elevata criticità per assicurarsi che tutti gli stakeholder sappiano quali siano le proprie responsabilità può essere utile avere a disposizione una matrice delle responsabilità anche nota come RACI Matrix. Lo scopo di questa matrice è il documentare per ogni deliverable, gli stakeholder del progetto responsabili per la creazione, revisione, approvazione o supporto. La suddivisione dei ruoli prevede:

- **(R)**esponsible, è responsabile dell'esecuzione/completamento di determinati compiti, è un esecutore sia individualmente che come parte di una team. Può delegare l'esecuzione ad altri individui o team, ma non può delegare la propria responsabilità.
- **(A)**ccountable, è responsabile per il risultato del compito, convalida o controlla altre entità rilevanti che completano il loro compito, definendo qualità / tempo / budget.
- **(C)**onsulted, viene consultato per compiti specifici o attività, fornisce raccomandazioni, guida e supporto ad altri ruoli su vari argomenti o domini.
- **(I)**nformed, viene informato regolarmente o ad-hoc, secondo il piano di comunicazione definito, su azioni, decisioni o milestone.

IT STREAM								
		1	2	3	4	5	6	7
Solution Architecture	Solution architecture design confirmation	C	C	C		C	A/R	C
	Tracking, review & governance	A	C			C	C	
	Maintenance	C	C	C			A/R	
	Business requirements & scoping	C	C	A/R		C	C	C
Requirements	Analysis		C	R		R	A/R	
	Functional specification		C/R	R		C	A/R	
	Technical specification		C/R	R		C	A/R	
	Tracking, review & governance	A	C	C	I		C	I
Development	Development environment		R				A/R	
	Coding & unit testing		R				A/R	
	Tracking, review & governance	A	C		I		C	I

Questa matrice risulta essere da supporto per la comunicazione. Attraverso questa tabella è possibile individuare la tipologia delle informazioni che devono essere trasmesse. In altre parole analizzando nel dettaglio la riga evidenziata dell'esempio sopra citato avremo che per definizione dei requisiti di business tutti gli stakeholder sono coinvolti (fatta eccezione per il numero 5 perché struttura di supporto PMO), lo stakeholder principale è il business e dunque avrà la responsabilità dell'attività cosi come del risultato. Tuttavia essendo una attività tra le più importanti e trasversali all'interno di un progetto complesso, vengono coinvolte tutte le altre strutture con il ruolo di supporto ovvero di "consulted".

Fino ad adesso si è trattato degli stakeholder senza sottolineare la differenza ad esempio Product Owner o Capo Progetto, termini introdotti nei capitoli precedenti, questo perché di fatto la loro rilevanza e differenza subentra nel momento dell'assegnazione delle responsabilità e ruoli all'interno del progetto. Tuttavia identificando prima per via concettuale la differenza tra gli interessi, il potere e il coinvolgimento permette di apprendere il loro ruolo e sarà più semplice in un secondo momento collegarlo ai vari ruoli già identificati.

Paramenti di valutazione del progetto

Lo sviluppo software non può essere un processo perfetto in tutte le sue fasi, anzi è molto probabile che nel corso della sua esecuzione intervengano fattori critici che se non opportunamente trattati posso creare problematiche sul progetto che risultano spesso in ritardi, fallimenti o aumento dei costi. L'obiettivo deve essere quindi quello di identificare dei parametri di valutazione del progetto, da monitorare in modo da avere consapevolezza nel modo più oggettivo possibile dell'andamento del progetto. È opportuno sottolineare due differenze presenti nelle valutazioni di successo. La prima differenza distingue il progetto di successo, inteso come misura rispetto agli obiettivi, e la gestione progettuale di successo intesa come misura rispetto a tradizionali indicatori di prestazioni in materia di costi, tempo e qualità. La seconda distinzione riguarda la differenza tra criteri di successo intesi come i parametri di successo (o fallimento) che verranno giudicati, e i fattori di successo intesi come quegli input al sistema di gestione che conducono direttamente o indirettamente al successo del progetto. L'approccio spesso utilizzato è quello di analizzare progetti che hanno riportato performance non soddisfacenti con quelli che ne hanno riportate di buone. Nel primo caso ovviamente si cercherà di capire quali sono le attività che hanno impattato negativamente sul progetto in modo da non replicarle in progetti futuri, al caso contrario nel caso di progetti di successo in cui si deve capire cosa ha funzionato per poterlo replicare. I problemi che si possono incontrare vengono divisi in quattro categorie:

- Problemi organizzativi;
- Persone;
- Processi;
- Tecnici.

Considerare la serie di parametri che vengono elencati di seguito in base alla categoria a cui sono associati sono utili a comprendere come la metodologia si possa adattare al progetto in questione, quali verifiche adottare e se l'esecuzione del progetto può presentare potenziali rischi di non successo.

Organizzativi

- Mancanza di un executive sponsorship - **verifica proposta**: numero di incontri in cui gli sponsor del progetto vengono coinvolti, 1 ogni tre mesi.
- Mancanza di coinvolgimento dei managers - **verifica proposta**: numero di incontri e decisioni in cui i manager vengono convolti,1 mensile.

- Cultura organizzativa troppo tradizionale (struttura rigida piramidale) - **verifica proposta**: numero di figure del team con responsabilità, 1 per ogni stream strategico.
- Cultura organizzativa troppo politica (limite alla delega delle decisioni) - **verifica proposta**: numero di figure del team delegati a prendere decisioni operative, 1 per ogni stream strategico.
- Organizzazione troppo grande (mancanza di responsabilizzazione) - **verifica proposta**: numero di team coinvolti e valutazione delle loro attività a valore aggiunto, 1 team per ogni stream strategico eventualmente condiviso con al massimo due stream.
- Mancanza di un'organizzazione agile - **verifica proposta**: numero di pratiche agile applicate.

Persone

- Mancanza delle necessarie competenze - **verifica proposta**: valutazione delle competenze dei team in linea con le necessità tecniche del progetto, elenco delle competenze necessarie e verifica della presenza di specifiche figure che coprano tutti i ruoli necessari.
- Mancanza di competenze nella gestione del progetto - **verifica proposta**: valutazione delle competenze ed esperienze del capo progetto.
- Mancanza di lavoro in team - **verifica proposta**: numero di meeting di allineamento tra i membri del team 1-2 a settimana.
- Ostacoli dal gruppo di lavoro
- Non buona relazione con il cliente - **verifica proposta**: numero di incontri in cui vengono esposte le valutazioni del cliente e quante di queste vengono accettate, in via di principio dovrebbero essere prese in considerazione tutte le valutazioni del cliente, tuttavia è necessario valutare ed aiutare l'utente a verificare l'effettiva consistenza delle sue valutazioni.

Processi

- Scope del progetto non definito correttamente - **verifica proposta**: aree progettuali coperte e percentuale di copertura dei gap.
- Requisisti del progetto non definiti in maniera esaustiva - **verifica proposta**: numero di requisiti identificati per ogni release e parametri di accettazione.
- Piano di progetto non dettagliato - **verifica proposta**: livello di dettaglio del progetto, con particolare focus sulle attività a più alta priorità e che sono da predecessori a più attività successive.

- Mancanza di un meccanismo di tracciatura agile dei progressi - **verifica proposta**: numero di sprint e retrospective analysis, la release non deve essere mai più ampia di 2-3 settimane.
- Mancanza del coinvolgimento del cliente - **verifica proposta**: numero di incontri con il cliente, 1 ogni due settimane di allineamento, in aggiunta agli incontri per la validazione della release.
- Mancanza di un definito ruolo del cliente - **verifica proposta**: identificazione nella matrice Raci del ruolo dell'utente non solo nella validazione a fine processo ma anche in tutte le fasi intermedie.

Tecnici

- Mancanza di un completo set di pratiche agile corrette da utilizzare - **verifica proposta**: valutazione insieme allo Scrum Master della metodologia e degli strumenti implementati.
- Tecnologia e strumenti inappropriati - **verifica proposta**: valutazioni insieme al team progettuale dell'adeguatezza degli strumenti utilizzati.

Considerando ora i parametri di successo, questi sono divisi nelle medesime categorie viste per i fattori che possono produrre un fallimento. Le voci che sono elencate sotto come parametri di successo sono quegli attributi che aiutano a dare una percezione complessiva di buona riuscita del progetto. Molti di queste valutazioni sono strettamente collegate con quelli che possiamo chiamare fattori di rischio, ciò che bisogna fare è considerare queste come linee guida o checklist che supportano la supervisione e la guida del progetto al fine di ridurre i rischi di insuccesso

Organizzativi

- Supporto esecutivo strutturato
- Organizzazione dove la metodologia agile è diffusamente accettata
- Assegnazione dell'intero team

Persone

- Membri del team con competenze e motivazione
- Team capace di gestirsi autonomamente
- Buona relazione con l'utente

Processi

- Adozione di un approccio Agile-oriented nella gestione del team, dei processi, del progetto e della configurazione dello stesso
- Ottima comunicazione tra i componenti del team con incontri di focus giornalieri
- Presenza strutturata del cliente

- Cliente con una chiara responsabilità e con autorità decisionale

Tecnici

- Si persegue un design semplice senza complessi artefatti
- Attività di risoluzione bug e riprocessamento rigorosi ma snelli
- Consegne prioritarie per le specifiche più importanti
- Fase di test di integrazione (ma anche UAT e system tests) pianificata accuratamente

A queste categorie è possibile poi aggiungerne una quinta relativa al progetto che deve essere strutturato per supportare cambiamenti di scope, cosi come situazioni di dinamicità e sprint accelerati in casistiche specifiche. Per questo specifica situazione si può valutare il tempo di risposta a una richiesta di cambiamento proveniente dall'utente e quanto in termini percentuali il cambiamento impatta sul progetto.

La differenza che esiste tra l'approccio tradizionale di misurazione dell'andamento del progetto e l'approccio agile, è che nell'approccio tradizionale la misurazione dei progressi progettuali è basato su un piano predefinito e su obiettivi misurabili che possono contrastare con il valore principale dell'agile di implementare tutti cambiamenti sorti in corso d'esecuzione. L'approccio tradizionale propone una serie di metriche globali che non sempre si adattano bene al principio di semplicità dell'agile. Per semplificare questo contrasto tra approccio agile e tradizionale possiamo elencare le principali differenze nelle misurazioni:

- Tradizionale: controllo diretto con un punto di vista esterno; tracciamento rispetto ai risultati finali, agli obiettivi misurabili definiti, seguendo un piano e un programma di grandi dimensioni.
- Agile: team di controllo con attenzione al cliente, orientati alla semplicità, alla tendenza, al feedback e alla risposta al cambiamento.

La maggior parte delle informazioni è relativa alla qualità, come i report sui testi automatici. Questo set di informazioni sono necessarie oltre al monitoraggio delle attività anche come supporto alla comunicazione e alle decisioni da prendere.

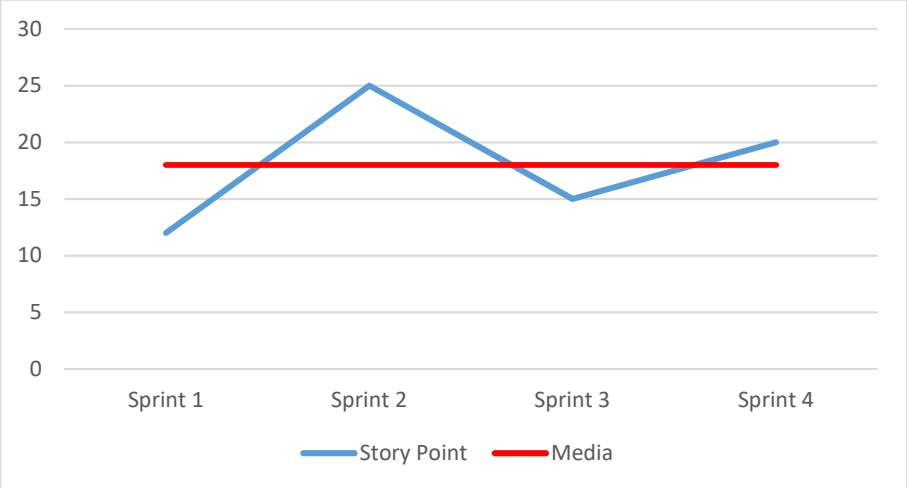
A questo punto è opportuno proporre una serie di metriche da utilizzare per monitorare il progetto durante la sua esecuzione.

La prioritizzazione delle attività è una delle attività principali nella pianificazione. Per effettuare la prioritizzazione viene utilizzato l'effort stimato per ciascuna attività e il loro relativo peso (inteso come attività principale rispetto ad altre seguenti), ma è possibile anche utilizzare un diverso approccio ovvero valutare quali attività sono a maggior valor aggiunto per il cliente, e porre queste a più alta priorità nella schedulazione del backlog. L'effort o il lead time viene utilizzato insieme alla velocity per

valutare se si è in grado di completare in tempo per il rilascio le attività che sono state previste. La velocity media viene anche utilizzato come parametro per il minimo set di task da rilasciare per ciascun sprint. Per il calcolo della velocity si devono prendere in considerazione soltanto le user story che sono state completate e validate al termine di ciascuno Sprint. Quindi per esempio, se in un caso abbiamo 15 user story point la velocity sarà pari a 15. Questo valore preso singolarmente non ha una importanza singola specifica. Per capire quanti story point possono essere verosimilmente completate negli Sprint successivi, bisogna fare una media aritmetica del punteggio totale di ciascuna iterazione. Ad esempio nel caso di 4 iterazioni con rispettivamente 12, 25,15 e 20 story point avremo:

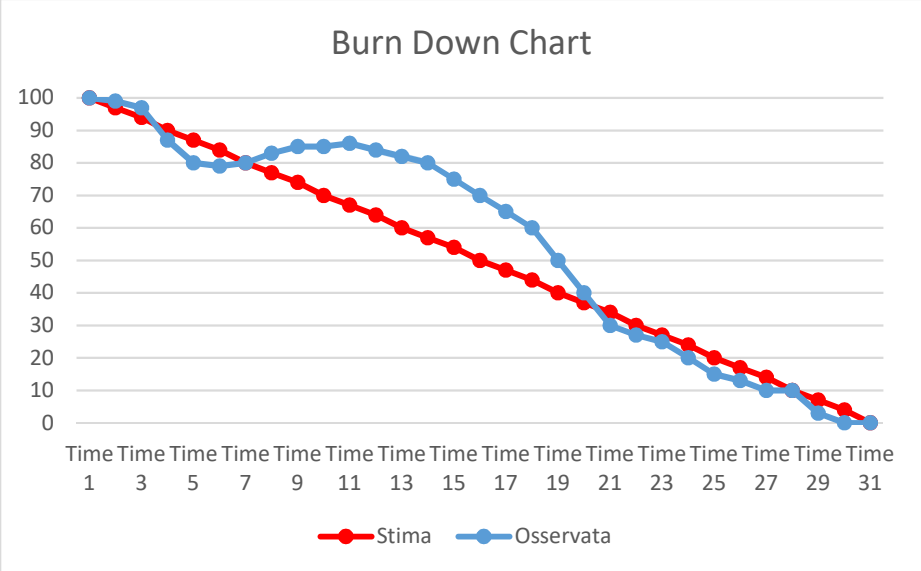
velocity media = (12 SP + 25 SP + 15 SP + 20 SP) / 4 = 18 SP

La nostra velocity media, in relazione ai quattro Sprint, è 18 SP. Per avere una stima attendibile della capacità di un team, e per fare previsioni sui rilasci in maniera più efficace, conviene prendere in considerazione almeno 4 o 5 Sprint. La velocity non deve comunque limitare le ambizioni di un team. Se la media è di 18 SP, non vuol dire che bisogna agire al ribasso e pianificare soltanto per quella quantità di story point, anzi può essere stimolante provare a migliorare il proprio punteggio ad ogni iterazione. È inoltre opportuno sottolineare che la velocity relativa di ogni team può variare a seconda della natura del team e delle attività a loro carico. Di seguito è proposto un esempio dove la linea blu mostra l'andamento delle user story rispetto a un valore medio (linea rossa). Le oscillazioni attorno al valore medio sono accettabili in quanto riflettono le dinamiche progettuali. Considerazione diversa è nel caso si ampi scostamenti dai valori medi.



Per stimare i progressi del progetto può essere utilizzato come valore il numero di test passati, questa stima può dare evidenza sulla quantità di lavoro completato. Altra metrica utilizzabile è il numero di user story

completate o in altri casi la dimostrazione dei progressi all'utente con un codice funzionante in un periodo ben definito. Un ulteriore strumento utile per visualizzare immediatamente l'andamento del progetto è l'utilizzo del burn-down chart che raffigura la quantità di attività rimanenti per il completamento del progetto. È possibile a questa sovrapporre una stima o desiderata, in maniera tale da valutare l'andamento rispetto alla stima e capire se gli scostamenti sono dovuti a ritardi o per esempio a cambi richiesti. Nel grafico seguente la linea osservata (blu) dovrebbe seguire il più possibile la linea rossa derivante dalla stima del lavoro rimanente. Una situazione positiva sarebbe quella in cui la stima è spesso sopra la linea delle osservazioni in quanto vorrebbe dire che l'andamento e il completamento delle task sta avvenendo più velocemente del previsto. Situazione assolutamente non positiva sarebbe il caso contrario, tuttavia è possibile che in certi casi ci siano momenti del progetto in cui la quantità del lavoro rimanente sia superiore a quella stimata e tralasciando il caso di problematiche sorte nel corso dell'esecuzione del progetto, un ulteriore spiegazione potrebbe essere la richiesta di cambiamenti o la richiesta di nuove funzionalità che devono essere incorporate nella fase di sviluppo. Nel caso di rilevanti scostamenti rispetto alla pianificazione che possono mettere a rischio il raggiungimento degli obiettivi nei tempi previsti e opportuno tempestivamente valutare la possibilità di ridurre le task ancora da eseguire, assegnando le priorità secondo le logiche di valore per l'utente e effort illustrati precedentemente o ancora valutare con maggiore attenzione eventuali nuove richieste provenienti dall'utente.



Un ulteriore buona pratica a beneficio della trasparenza è la presentazione periodica degli avanzamenti del progetto collegati con

l'andamento dei costi. È possibile anche nel caso dei costi presentare una stima dell'andamento nel corso del progetto insieme all'effettiva spesa sostenuta. In termini manageriali la variabile costo è uno delle variabili più visibili in cui ogni scostamento significativo deve essere opportunamente trattato, previsto e illustrato.

Per valutare la **qualità** del prodotto o delle parti di esso rilasciate viene valutato il numero di cambi richiesti (change request – CR) o il numero di richieste di manutenzione pervenute dal cliente. Allo stesso modo è possibile prevedere uno specifico limite al numero di work in progress (wip), in quanto i wip contengo potenzialmente un numero indefinito di difetti che possono essere scoperti e risolti solo nel momento in cui vengono eseguiti i test sulla specifica funzionalità e congiuntamente i test di non regressione sul sistema. Uno specifico punto di attenzione emerge nel momento in cui il numero di test scritti e correttamente eseguiti con successo non aumenta. Per l'approccio nella gestione del wip o del backlog è preferibile implementare un approccio pull piuttosto che pull in modo da delegare il team nella gestione dello stesso, in modo da garantire sempre il *continous delivery*.

Il tempo di processamento e il tempo di coda sono utilizzati per identificare gli **sprechi** per cui un requisito attende per lungo tempo prima del completamento delle sue specifiche. L'approccio suggerito deve essere quello di creare una proposta di alto livello da inviare al cliente senza una analisi dettagliata delle specifiche. L'utente può quindi usare la proposta di alto livello per valutare se procedere con l'analisi più dettagliata o scartare quello specifico requisito. Inoltre un lungo tempo di processamento può suggerire uno spreco dovuto alla circostanza in cui lo specifico requisito è posizionato nel mezzo tra il marketing e l'unità di sviluppo, in questo caso è necessario aumentare la collaborazione tra questi due team in maniera particolare nel caso di requisiti complessi. Un ulteriore spreco è identificato nel caso di lunghe code o lunghi tempi di processamento che fanno permanere lo specifico sviluppo troppo a lungo nella fase di test, in questo caso il suggerimento è di rilasciare il prima possibile una versione beta e nel contempo continuare con i test mancanti. Il rischio reale di ritardi soprattutto a ridosso della deadline di fine progetto è quello di trasferire molti requisiti nella fase di system test. Questo approccio oltre ad essere estremante contrario alla logica di rilascio che passa per una serie di test utili a garantire la qualità del prodotto può nascondere problematiche che se non riscontrate in questa fase possono verificarsi negli step più a valle con impatti potenzialmente maggiori.

Un ulteriore indicatore della qualità è l'andamento dei **bug** durante tutte le fasi di sviluppo. In definitiva questo parametro mostra se il numero di difetti (nel backlog) aumenta, diminuisce o rimane stabile. Nel caso in cui tale valore non abbia un andamento in linea con il progetto il PM può agire immediatamente per identificare la problematica. In linea con il progetto si

intende la situazione in cui l'andamento deve essere sicuramente decrescente ovvero difetti devono riguardare solo parti nuove rilasciate e non parti già testate e completate, in caso contrario è evidente che esiste un problema.

Per quanto riguarda invece una metrica che a volte viene tralasciata riguardante la **motivazione del team** si possono adottare strategie per aumentare la motivazione del team. Il numero di difetti può essere mostrato in tempo reale negli spazi in cui le persone lavorano, questo aumenta la motivazione nella risoluzione dei difetti e nell'evitarne di nuovi. Inoltre è possibile prevedere specifici incontri in cui vengo mostrati per ciascun team la parti in debito di sviluppo, in queste circostanze il confronto è necessario per definire la miglior strategia o soluzione per eliminare questo debito di sviluppo e allo stesso tempo il team si impegna a prendersi carico di un debito in particolare che dovrà essere completato entro la riunione successiva. Un ulteriore pratica suggerita al fine di motivare il team è l'inserimento di differenti milestone progettuali intermedie con le quali si può celebrare e motivare il team e il lavoro da loro svolto.

Nell'ottica di valutazione del progetto è possibile illustrare anche tutte quella serie di elementi che sono considerati degli sprechi a livello progettuale. L'idea di fondo è che la qualità del progetto si può raggiungere concentrando l'attenzione solo sugli elementi che aggiungono valore ed eliminando tutte le fonti di spreco. La serie di sprechi che spesso vengono identificati riguardano:

- Lavorazioni parzialmente completate, si riferiscono a tutte quelle attività sviluppate parzialmente che però non sono state implementate, testate, integrate e che rischiano di diventare obsolete.
- Extra-Processi, si riferisce ai quei processi non necessari o che possono essere rimandati alla fine del progetto (report, documentazione).
- Extra-Funzionalità, la tendenza potrebbe essere quella di aggiungere funzionalità al sistema che non vengono strettamente richieste ma che potrebbero risultare utili, sebbene possa sembrare una buona iniziativa, questo può creare ritardi sulle funzionalità core.
- Task-Switching, come è facilmente osservabile lo scambio di risorse frequenti tra diverse attività crea una diminuzione dell'efficienza lavorativa.
- Attese, è uno dei più importanti sprechi che si verificano nell'ambito dello sviluppo software. Come già visto è necessario ridurre e prevedere processi strettamente monitorati in modo tale che nessuna delle attività o azioni previste aspetti più del dovuto in passaggi a basso valore aggiunto (approvazioni, verifiche etc.).

- Motion, è valutato come tempo medio per l'identificazione della risposta a uno specifico quesito. La pratica suggerita e spesso adottata è quella che si può chiamare regola 48: se il sottoposto non risolve una specifica questione nelle successive 48 ore, è compito del suo capo risolverla tempestivamente. Questo non significa rispondere sempre di "si" ma vuol dire risolvere il problema con una risposta e un piano chiaro.

Volendo identificare un'unica variabile di controllo che possa tracciare ed eventualmente sottolineare una criticità è il **numero di richieste di manutenzione**. Purtroppo in questo caso non è possibile suggerire un valore standard di riferimento perché questo valore è strettamente legato alla complessità generale del progetto, dal numero di applicazioni coinvolte o dal numero di sistemi utilizzati. In via di principio è ovvio che tale valore dovrebbe essere il più basso possibile e dopo il rilascio, quindi durante il Business As Usual (BAU) il numero di difetti dovrebbe essere sempre molto baso o assente. Nell'ottica degli sprechi, i difetti sono universalmente considerati degli sprechi, ma considerando il modo dello sviluppo applicativo, i difetti sono qualcosa che qualsiasi sviluppatore aspetta di trovare una volta rilasciato il prodotto. Il punto focale a questo punto è identificare un processo solido che gestisca tutte le richieste che pervengono dall'utente. Solitamente vengono utilizzate piattaforme dedicate di bug tracking i quali sistemi permettono l'accesso al cliente e allo sviluppatore in modo tale che una volta aperta la segnalazione questa possa essere seguita da ambo le parti fino alla risoluzione. Con questo tipo di approccio è ovviamente necessario identificare delle metodologie per assegnare la corretta priorità alle differenti richieste. Un approccio spesso utilizzato nella situazione di applicazioni che offrono servizi direttamente ai clienti di una società la priorità viene assegnata rispetto al potenziale impatto (numero di clienti impattati, numero di transazioni di pagamento, tipologia del cliente). Congiuntamente sono stipulati contratti specifici con i fornitori degli applicativi, in cui vengono elencati per ciascun servizio i livelli si SLA (Service Level Agreement), riguardanti i livelli massimi di disservizio ritenuti accettabili ed entro i quali non vengono previste sanzioni e vengono anche concordati i tempi di risoluzione per le funzionalità con errori identificate.

Considerando il parametro che traccia il numero di bug è possibile utilizzare il dato proveniente dalle serie storiche per crearne uno aggiuntivo. Quello che si vuole cercare di identificare è un parametro di affidabilità del software. Con questo obiettivo è possibile analizzare le serie storiche e basandosi su questi dati si può individuare una stima sulla probabilità di malfunzionamento del software in un determinato intervallo di tempo (es. tempo intercorrente tra due bug critici con impatti sull'utente).

In conclusione è necessario che i parametri finora visti vengano utilizzati concretamente per presentare lo stato del progetto. La richiesta di

documenti specifici, per il tracciamento delle performance cosi come degli sprechi, da parte dei capi risulta essere l'approccio migliore per rendere utile questo tipo di misurazione. Il concetto di spreco è alla base dell'approccio Lean e Agile, nel approccio tradizionale lo strumento utilizzato per rimuovere gli sprechi è quello di condurre specifici assessment ed implementare processi migliorativi. Nella metodologia agile questo approccio continua ad essere valido ma la migliore assicurazione sulla qualità deriva dalle attività di apprendimento previsto durante tutta la fase progettuale, avendo quindi un approccio non solo bottom-up ma anche top-down. La misurazione deve essere il punto di partenza per intraprendere un percorso di miglioramento e non per rafforzare il controllo.

Come scegliere tra le diverse metodologie

Al lancio di tutte le attività progettuali esiste un set di attività preliminari che devono essere condotte in quanto prerequisito per l'intero progetto. Questa serie di attività che vanno dall'analisi dei costi all'analisi di fattibilità sono gli elementi fondamentali per avere basi certe e condivise con cui avviare tutte le successive fasi progettuali. Un passo altrettanto importante e anche esso preliminare è la scelta della metodologia con cui gestire tutte le attività di sviluppo, sia essa un approccio tradizionale, Agile o ibrido. Il motivo di questa scelta preliminare è ragionevole, considerando che l'approccio di gestione progettuale influisce su tutte le fasi e le attività. Tenuto conto il coinvolgimento profondo della metodologia in ogni parte del progetto è fondamentale che questa scelta segua un processo ragionato, che attraverso valutazioni oggettive produca come risultato la scelta della metodologia che meglio si adatta al progetto da avviare. Spesso questa analisi viene condotta o attraverso specifici questionari che permettono di guidare la scelta o attraverso il confronto con progetti simili. Come vedremo di seguito la scelta della metodologia più adatta passa per l'identificazione di caratteristiche del progetto che per la loro specificità si adattano meglio a un approccio piuttosto che a un altro.

Complessità generale del progetto

La stima della complessità del progetto è una parte fondamentale nella scelta della metodologia. La valutazione della complessità passa sicuramente dalla definizione della dimensione del progetto, ma la complessità riguarda sicuramente il numero di sistemi coinvolti e il tempo per l'implementazione. In maniera più analitica le 6 informazioni necessarie per il calcolo della complessità sono:

- Dimensione economica complessiva;
- Dimensione temporale complessiva;
- Dimensione organizzativa complessiva;
- Indeterminatezza dei requisisti;
- Innovazione;
- Rigidità dei vincoli realizzativi.

La suddivisione è finalizzato ad identificare certamente la complessità dell'iniziativa, abilita ad attività progettuali più complesse in linea con la criticità del progetto e fornisce delle linee guida sulle metodologie da utilizzare. La definizione della complessità rispetto alle 6 informazioni classifica lo stesso in tre differenti livelli: Alto, Medio e Basso.

I range proposti sono riportati nella tabella seguente:

	Range		
Dimensione economica	Durata <= 6 Mesi	6 Mesi < Durata <= 11 Mesi	Durata > 11 Mesi
Dimensione temporale	Costo <= 0,4 mln	0,4 mln < Costo <= 0,9 mln	Costo > 0,9 mln
Dimensione organizzativa	Fabbriche <= 3	3 < Fabbriche <= 7	Fabbriche > 7
Indeterminatezza dei requisisti	Variabilità <= 10%	10% < Variabilità <= 30%	Variabilità >30%
Innovazione	Modifica a processi (es. modifica processo organizzativo)	Modifica a prodotto o a tecnologia	Nuovo prodotto, servizio, processo o tecnologia
Rigidità dei vincoli realizzativi	1 vincolo fissato	2 vincoli fissati	Più vincoli fissati

Tale suddivisione riporta come progetto ad alta complessità un progetto dalla durata rilevante e con importanti immobilizzazioni di capitale. Al fine della valutazione dell'implementazione agile le informazioni sicuramente più rilevanti sono riportate nell'ambito dell'indeterminatezza dei requisiti, dal grado di innovazione, dalla dimensione organizzativa. La rigidità dei vincoli invece è un'informazione molto importante a livello progettuale, minore nella scelta della metodologia in quanto essendo dei vincoli progettuali (in termini di tempo, funzionalità, caratteristiche, richieste del regolatore) devono essere gestite al livello di progetto anche attraverso l'uso del *reverse engineering* se utile in modo che tali vincoli vengano rispettati. Per quanto concerne la scelta della metodologia un elemento da tenere in considerazione è sicuramente la valutazione rispetto alla indeterminatezza dei requisiti e al grado di innovazione. L'approccio agile assicura quella flessibilità necessaria per gestire situazioni di variabilità dovute alla mancanza di completa informazione o dalla necessità di esplorare in maniera iterative soluzioni innovative.

Un altro strumento utilizzato per valutare la complessità del progetto è considerare le linee di codice (lines of code – LOC), le quali tuttavia possono essere stimate solo a conclusione degli sviluppi ma può offrire un ottimo parametro, se relazionato al budget per lo sviluppo, sul costo medio di sviluppo. Collezionando queste informazioni su diversi progetti e con differenti linguaggi software di scrittura è possibile registrare tutti questi valori che possono essere utilizzati come supporto per la stima di altri progetti, capire se si è speso troppo o troppo poco. Queste informazioni

nel loro insieme sono realmente utili per avere una valutazione oggettiva di supporto alle scelte.

Uno ulteriore strumento con cui viene definita la complessità del progetto nella fase iniziale è la valutazione dei Function Points con cui si assegna un valore specifico per:

- Input Esterni, dati che provengono dall'esterno del perimetro dell'applicazione, punteggio 4 (per item);
- Output esterni, dati che vengono inviati all'esterno del perimetro dell'applicazione, punteggio 5 (per item);
- Interrogazioni esterne, è la combinazione di input (processati) e output (che producono una risposta), punteggio 4 (per item).

La somma dei valori individua la complessità generale del progetto. Il range basso/medio/alto può essere individuato come segue: Basso <100, Medio 100<X<160, Alto >160.

Riferendosi alla sezione precedente dove è stata trattata la problematica relativa all'identificazione di KPI utili a monitorare l'andamento del progetto, è possibile utilizzare la medesima argomentazione per la scelta della metodologia da utilizzare. Quello che si vuole proporre è di fare un'analisi comparativa tra elementi di successo e gli elementi di fallimento. Nella valutazione si chiede di definire quali elementi di entrambe le liste (successo e fallimento) si prevede possono essere presenti nel progetto e quali no. In conclusione se il numero (pesato) degli elementi di fallimento sono maggiori di quelli di successo allora ovviamente è necessario ripensare all'applicazione della metodologia Agile ma forse anche all'esecuzione del progetto stesso.

Parametro	**Agile**	**Tradizionale**	**Priorità**
Complessità del progetto	Medio/Bassa	Alta	Alta

Variabili costo e tempo

Un ulteriore analisi che viene richiesta è la valutazione delle variabili costo e tempo. L'obiettivo è stimare quanto i driver costo e tempo siano variabili o meno, in una scala che va da 0 a 5 (da fisso a variabile). L'osservazione porterà a risultati ovviamente diversi in quanto considerando che la metodologia agile prevede di incorporare richieste differenti che pervengono dall'utente o dal mercato, l'implementazione di tali cambiamenti possono avere un impatto sui tempi di implementazione ed ovviamente sui costi degli stessi. La valutazione viene interpretata nel seguente modo: per risultati da 0 a 1 la metodologia suggerita è quella tradizionale, con risultati da 2 a 3 si richiede di fare un'analisi su quali sistemi o processi possono essere impattati in termini di costo e tempi oltre alla

stima della potenziale contingency necessaria, con valore da 4 a 5 i driver non sono considerati fissi e quindi l'approccio proposto è la metodologia agile. Con riferimento a queste variabili bisogna considerare che essendo i tempi e costi tra i due più importanti driver progettuali, spesso si è spinti nel definirli fissi perché nessuno auspica a una situazione di variabilità di queste due voci, quindi sebbene nell'ambito dell'analisi dell'approccio a queste due voci viene dato un peso inferiore l'attenzione che si richiede di avere è sulla stima della contingency. Un ulteriore domanda che viene posta in merito al progetto è la valutazione del time to market. In questo caso la richiesta che viene fatta è solo di indicare con un'affermazione negativa o positiva quanto il TTM è importante. Come visto il progetto agile è orientato al prodotto e all'utente e a consegnare il prodotto al momento in cui viene richiesto, in altre parole quando il time to market è fondamentale.

Parametro	Agile	Tradizionale	Priorità
Variabili costo e tempo	4-5	0-1	Bassa

Aree di coinvolte

Successivamente viene analizzato l'impatto del progetto sull'intero sistema. Si valutano le aree di business, applicative e il numero di fornitori coinvolti. La scala proposta per ciascuna area, applicazione e fornitore coinvolto va da 0 a 4. Questa analisi è utile per capire il perimetro progettuale. Come visto un progetto agile poco si adatta sistemi con il coinvolgimento di troppo attori dove il coordinamento e l'agilità perde d'efficacia a causa della complessità organizzativa e anzi potrebbe creare in alcuni casi maggiori difficoltà. in quest'ottica l'approccio agile viene suggerito nel caso di un totale fino a 9, negli altri casi per un totale di 12 l'approccio più adatto potrebbe essere quello tradizionale, eventualmente coadiuvato da un approccio agile nelle singole aree di progetto.

Parametro	Agile	Tradizionale	Priorità
Aree progettuali coinvolte	0-3	3-4	Media

Livello di incertezza

Nel caso di incertezza progettuale alla luce di quanto discusso nei capitoli precedenti l'incertezza è uno degli elementi che abilità alla metodologia agile. La scala utilizzata ha valori alto/medio/bassa. In corrispondenza di livelli di incertezza media e alta l'approccio suggerito è quello agile, tradizionale nel caso di bassa incertezza. In questo caso la metodologia agile è possibile applicarla in tutti e tre casi in quanto in caso

di incertezza non è possibile pianificare l'intero progetto per tutta la sua durata utilizzando gli strumenti waterfall in quanto situazioni impreviste possono avere impatti non prevedibili sulla pianificazione e questo è ancora più vero nel caso di pianificazione iniziale del progetto dove la visibilità su intera vita dell'iniziativa non può essere esaustiva.

Parametro	Agile	Tradizionale	Priorità
Livello di incertezza	Media/Alta	Bassa	Media

Revisione dei requisiti

Il quesito posto riguarda la possibilità o meno della modifica dei requisiti nel corso dell'esecuzione delle attività. La revisione dei requisiti può essere prevista anche all'inizio del progetto quando si pensa di implementare alcune funzionalità innovative, o l'obbiettivo di alcune di queste non è chiaro per cui è necessario continuare a investigare sulla soluzione da adottare. L'approccio agile prevede l'adeguamento dei requisiti durante il corso del progetto siano queste derivanti da richieste specifiche del cliente o dovuti ad analisi aggiuntive. La scala utilizzata va da 0 a 3, dove 0 prevede la invariabilità dei requisiti, mentre da 1 a 3 si prevede la revisione dei requisiti rispetto al loro numero, 1 basso, 2 medio, 3 alto numero di requisiti che si prevede rivedere. Fatto 100 il numero totale delle funzionalità previste se la previsione è del 15% di cambi il valore assegnato sarà basso, 25% medio, sopra il 40% alto. Una quota del 35% si considera sia fissa in quanto non è ragionevole pensare che il progetto possa avere la totalità dei requisiti da rivedere. La spiegazione è immediata se si considera al concetto degli sprechi e delle funzionalità base. In prima istanza è molto probabilità esistono delle funzionalità base da supportare che si è certi di sviluppare in una certa direzione ed inoltre per quanto riguarda gli sprechi, potrebbe essere un rischio importante iniziare degli sviluppi con la previsione che vengano rivisti. Rivedere dei requisiti può voler dire progettare e sviluppare nuovamente parti già lavorate.

Parametro	Agile	Tradizionale	Priorità
Revisione dei requisiti	1-3	0-1	Alta

Rilasci incrementali del prodotto

Un progetto che prevede rilasci incrementali si adatta bene alla metodologia agile in quanto condivide la logica di rilascio. Bisogna valutare anche se si prevede l'utilizzo di un prototipo per la realizzazione del prodotto o se la soluzione è di tipo custom o comunque utilizza delle soluzioni custom. In tutti questi casi se la valutazione è affermativa allora è da preferire l'approccio agile e valutare eventuali trade off. L'utilizzo del prototipo è molto vicino all'idea di release dell'approccio agile per cui a fine delle release si prevede di rilasciare una parte del prodotto/codice che

l'utente può validare. La soluzione custom prevede l'interazione con l'utente in modo da essere abilitatati direttamente da lui allo sviluppo e alla validazione in base alle sue richieste. Con un prodotto standard non ci si aspetta di avere particolari cambiamenti ed è quindi consigliato una metodologia tradizionale che non prevede tutte le pratiche agili che se non utili possono anche in questo caso produrre ritardi. Infine l'utilizzo di tecnologie innovative o semplicemente non applicate precedentemente possono nascondere parti non conosciute per cui un approccio incrementale di scoperta iterativa aiuta a diminuire i problemi dovuti all'incertezza. Sicuramente tutte queste informazione nella valutazione non hanno tutte lo stesso peso, quelle a più alto peso sono sicuramente relative al tipo di soluzione (custom o standard) e alla presenza o meno di rilasci incrementali.

Parametro	**Agile**	**Tradizionale**	**Priorità**
Rilasci incrementali del prodotto	SI	NO	Medio
Si prevede l'utilizzo di un prototipo	SI	NO	Basso
È prevista una soluzione custom	SI	NO	Alto
Si prevedono tecnologie innovative o nuove per la società	SI	NO	Medio

Il team

Precedentemente si è discusso del numero di aree progettuali coinvolte. Aree progettuali differenti prevedono solitamente team diversi e probabilmente anche dislocati. Il numero di risorse nel team per un approccio agile deve essere limitato nelle sue dimensioni e possibilmente allocato fisicamente in uno stesso spazio in modo da agevolare il processo di comunicazione e di condivisione delle informazioni. L'impiego di differenti team magari dislocati geograficamente sicuramente non sono l'ambiente più adatto ad implementare l'approccio agile. Un ultimo punto da considerare per quanto riguarda il team è la sua preparazione o conoscenza delle pratiche e delle metodologie agili, in mancanza di queste è sicuramente necessario prevedere un periodo di apprendimento e considerare un potenziale rischio.

Di seguito è riportata una tabella riassuntiva delle caratteristiche rilevanti e della differenza tra l'approccio agile e quello tradizionale.

Caratteristica	Approccio Tradizionale	Approccio Agile
Requisiti	Requisiti iniziali chiari e bassa probabilità di cambiamenti di questi	Requisiti non chiari e innovativi
Utenti	Non coinvolto	Alto coinvolgimento e collaborazione
Documentazione	Documentazione formale	Conoscenza tacita
Dimensioni progetto	Progetti grandi	Progetti medi/piccoli
Supporto Organizzativo	Organizzazione presente	Preparazione dell'organizzazione all'utilizzo del nuovo approccio
Team	Team dislocati e soggetti a fluttuazioni	Piccoli team localizzati nello stesso luogo
Impatti sui sistemi	Alta criticità a seguito del fallimento	Minor criticità ed impatti sui sistemi
Piano di progetto	Lineare	Complesso e iterativo

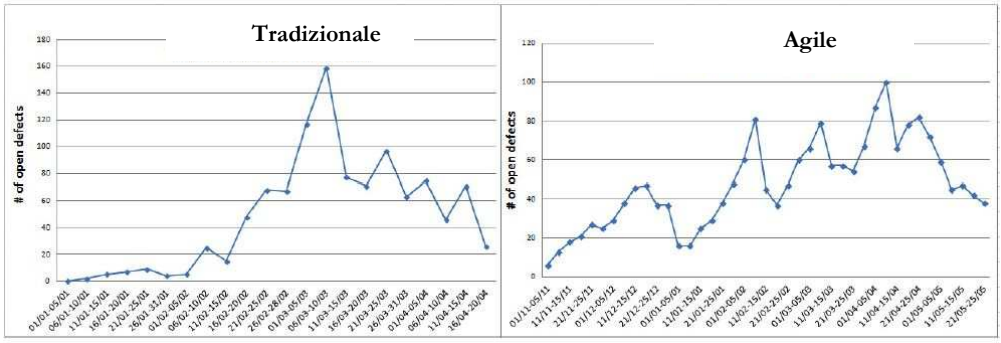
La valutazione della complessità è inoltre funzionale a un ulteriore stima. In via di principio la complessità si trasferisce anche sull'implementazione che viene eseguita. Questa complessità in termini pratici si rendere visibile anche attraverso l'analisi dei bug identificati durante le fasi di sviluppo, maggiore complessità potrebbe tradursi in potenziali maggiori bug, dovuti certamente non alla qualità del lavoro bensì alle difficoltà intrinseche nel progetto. Di seguito è proposta un'analisi che mostra dal punto di vista dei bug cosa cambia in un approccio tradizionale e in un approccio agile. Questa analisi oltre a fornire interessanti evidenze può essere presa in considerazione nella scelta della metodologia da utilizzare.

Difetti Approccio Tradizionale

Il numero di difetti lungo l'arco temporale ha una distribuzione mostrata in figura (Tradizionale). Il periodo preso in esame riguarda la fase di sviluppo e test con un parziale accavallamento delle attività di test con quelle di implementazione. All'inizio dei test i difetti salgono rapidamente a quota 60 prima della fine dell'implementazione avvenuta a fine febbraio. Il numero di difetti sale a 160 durante la fase esclusiva di test. Tale situazione ha richiesto un coinvolgimento straordinario per il raggiungimento degli obbiettivi progettuali nelle tempistiche previste, con una successiva riduzione dei difetti sino al momento del rilascio del prodotto al cliente.

Quando nuove richieste arrivano durante la fase di test, non possono essere gestite senza riaprire l'analisi dei requisiti e la fase di Design. Dati i

costi rilevanti generati dall'introduzione di una nuova funzionalità dopo la chiusura della fase di design, l'opzione a minor costo sarebbe quella di inserire il nuovo requisito nella versione successiva.



Difetti Approccio Agile

Il numero di difetti lungo l'arco temporale ha una distribuzione mostrata in figura (Agile). I difetti sono distribuiti uniformante lungo l'arco temporale, tenendo conto che le implementazioni sono state rilasciate per Sprint con il picco di rilasci a marzo e il conseguente picco di difetti a 100. Le iterazioni successive al picco per il rilascio del prodotto in linea con i requisiti del cliente sono stati inferiori 2 contro 3 e con minori picchi e risoluzioni già nelle fasi iniziali del progetto. All'arrivo di nuove richieste, è stata assegnata una priorità data dal Product Owner e sono state inserite nel Backlog.

Dal punto di vista della soddisfazione del team, è subito evidente come l'utilizzo della metodologia Agile abbia influito sulla collaborazione tra i componenti del team, oltre alla percezione che il tempo perso per attività inutili sia stato ridotto notevolmente concentrando lo sforzo ad attività a valore aggiunto. Tutto questo è da imputare principalmente alla distribuzione dei difetti durante l'arco temporale del progetto questo perché il team ha potuto distribuire in maniera più ottimale il carico di lavoro evitando periodi di altro stress osservati nel caso di un approccio tradizionale, allo stesso tempo la possibilità di lavorare in team ovvero lavorare ad obbiettivi comuni ha fatto crescere la soddisfazione del team progettuale.

Il processo di test e l'identificazione dei difetti è tra gli strumenti utili a verificare la qualità del prodotto, come già visto la certificazione della qualità è un processo imprescindibile per questioni di efficienza, ma anche dal punto di vista dei costi. Considerando che una volta che il prodotto viene rilasciato ogni carenza di qualità viene segnalato attraverso l'apertura di specifici ticket, ogni ticket di primo livello ha un costo medio di processamento di 10 €/cad. per passare ai 40 €/cad. per ticket di un livello

più alto. La somma di tutti i ticket all'anno porta a un costo di milioni di euro l'anno. Questo vuol dire che la non qualità ha un costo che può essere controllato e previsto.

Conclusioni e Implicazioni

Nella moderna industria del software, rilasciare software ad alta qualità è l'obbiettivo principale per gli sviluppatori. La mancanza di panificazione e di un approccio non sistematico di sviluppo, può produrre quasi certamente un risultato finale di bassa qualità ed alti costi. In quest'ottica il ruolo che gioca la metodologia è fondamentale, in quanto influenza la qualità del prodotto rilasciato.

Trovare e correggere errori nelle fasi iniziali del progetto produce un grande vantaggio rispetto a un approccio tradizionale in quanto viene evitata la costruzione di caratteristiche e funzionalità avendo come base degli errori. Il rischio di costose riprogrammazioni nelle fasi avanzate del progetto è tenuto al minimo e il focus è posto alle attività che aggiungono valore e ai miglioramenti qualitativi.. Inoltre distribuire la capacità di sviluppo e l'ambienti di test in maniera equa su tutta la durata del progetto crea un ambiente più sostenibile evitando picchi che possono produrre inefficienze, questo rende evidente come la qualità nel processo di sviluppo è molto importante

Lo studio finora riportato ha mostrato anche che l'adozione della metodologia Agile non è semplice e le società, soprattutto quelle più grandi, complesse e con una lunga storia nell'applicazione della metodologia Waterfall, hanno bisogno di pianificare attentamente le attività di adozione e implementazione della metodologia. Inoltre l'approccio alla costruzione di una cultura Agile, può richiedere tempo, nel quale le attività devono essere strettamente controllate, risolvendo problematiche al loro nascere, con l'obbiettivo di creare un approccio che sia in linea con tutti i requisiti di letteratura ma che soprattutto sia a supporto dello specifico ambiente e società in cui la metodologia deve essere applicata.

I benefici da cogliere nella metodologia Agile applicata al progetto di sviluppo sono numerosi, considerando che impattano sull'intero ciclo di sviluppo. Questi benefici riguardano in definitiva la frequenza dei rilasci, la soddisfazione del cliente, la trasparenza con l'utente e all'interno del team, la flessibilità, la maggiore produttività, la maggiore qualità del software e minore necessità di prevenire applicando un approccio iterativo in situazioni di incertezza.

È stato riscontrato che gli sviluppatori mostrano un maggiore soddisfacimento del loro lavoro con il prodotto realizzato con metodologia Agile, così come è emerso che la metodologia Agile è anche funzionale all'apprendimento on the job cosi come si ha un aumento della produttività del team. La percezione della qualità è maggiore anche se in questo caso non è possibile dire con assoluta certezza che questo sia un effetto della

metodologia utilizzata, quello che è certo è che tale percezione deriva da quel processo iterativo che abilita a numerosi controlli, test e aggiustamenti nel corso del progetto. La variabile umana è molto importante se si considera che sebbene si tratti di un settore tra i più innovativi e tecnologici, poco può essere delegato a strumenti senza supporto umano. Test possono essere automatizzati, i software possono analizzare la correttezza del codice ed evidenziare eventuali malfunzionamenti, ma la parte di analisi, scrittura e risoluzione dei bug richiede ancora un alto coinvolgimento degli sviluppatori

Un aspetto determinante è la valutazione dei parametri identificati essenziali per i manager che devono prendere decisioni strategiche. Queste valutazioni devono essere un riassunto oggettivo sull'andamento del progetto. Inoltre bisogna sottolineare che tutti i benefici di un approccio strutturato in linea con i parametri o buone pratiche viste in precedenza non possono essere abilitate esclusivamente dal project manager o dal team di progetto, è invece necessario che tali azioni abilitanti vengano intraprese da dei manager operativi che rendano sicura la collaborazione tra il gruppo di lavoro con l'utente e gli sponsor di progetto. Realizzare un progetto di successo è sicuramente più difficile che raggiungere un successo dal punto di vista della gestione del progetto. Il project e programme management traduce i processi, le decisioni e la strategia in progetti, programmi e risorse (allocazione delle stesse). Ad un livello immediatamente inferiore invece lo operations management ha l'obbiettivo di gestire le operations per ottimizzare i benefici di prodotti o servizi.

Un progetto sviluppato con successo migliora il time to market producendo margini e migliorando le vendite o la posizione competitiva cosi come una fase design ben eseguita riduce costi operativi. Il principio che guida tutta l'iniziativa di monitoraggio è che senza monitorare non si può migliorare.

La metodologia Agile restituisce un ROI prima rispetto all'approccio tradizionale in quanto dati tecnici e specifiche sono gestibili, mentre i dati di qualità sconosciuta in quanto arrivano solo alla fine del ciclo di sviluppo non lo sono. Nell'approccio tradizionale subito dopo la specifica dei requisiti parte la revisione degli stessi. In questo processo centinaia di requisiti sono valutati dal team in un periodo limitato. Ci sono due grossi problemi con questo approccio: in primo luogo, rivedere l'intero set di requisiti alla ricerca di difetti non è un modo né efficace né efficiente per migliorare i requisiti, secondo, eventuali difetti che vengono rilevati dopo settimane o mesi di lavoro richiedono la riscrittura dei requisiti difettosi e l'identificazione di tutti i requisiti correlati che potrebbero essere interessati. Considerando che questo sforzo è differito, non solo si traduce in sforzo maggiore ma aumenta anche la probabilità di errori. Un feedback iniziale sul numero di richieste corrette è più prezioso di un feedback tardivo.

I rischi, in linea con la loro stessa natura, sono qualcosa che può mettere in discussione gli obiettivi del progetto e non è possibile sempre prevedere o conoscerli a fondo. L'approccio tradizionale prevede l'utilizzo di metodi per la riduzione del rischio ma spesso non sono utilizzati nella maniera più rigorosa. Il metodo agile utilizza la lista dei rischi o il piano di gestione dei rischi per identificare e gestire iterativamente potenziali rischi che possono ridurre la probabilità di successo del progetto. Impiegare strumenti specifici (Risk Burndown list) per apprendere le informazioni rilevanti e correggere se necessario, è la qualità intrinseca della metodologia agile di incorporare i cambiamenti.

Delle linee guida nell'impianto del progetto che ha impatti su tutta la vita, suggeriscono quali sono le raccomandazioni per indirizzare il progetto nella giusta direzione sin dalle prime fasi di ideazione. Come visto nel corso dell'analisi fatta queste raccomandazioni riguardano lo scopo e gli obiettivi del progetto che devono essere stabiliti in maniera rigorosa e chiara con la corretta qualità dell'informazione cosi come la definizione del team progettuale nelle sue dimensioni. Qualsiasi barriera comunicativa che impedisca la libera ed efficiente condivisione delle informazioni a tutti i livelli progettuali deve essere tempestivamente identificata ed eliminata. Sviluppare sistemi complessi è possibile, è richiesto dalla trasformazione digitale in corso, l'abilità deve essere quella di scegliere sempre l'approccio che assicuri il risultato in termini di costi e tempi, ma soprattutto che il risultato soddisfi pienamente le aspettative del cliente.

In definitiva nell'analisi dell'approccio da utilizzare bisogna considerare che l'approccio può essere adattato al progetto ma non è mai possibile adattare il progetto all'approccio scelto, bisogna rilevare con cautela quali sono le caratteristiche rilevanti che devono guidare la scelta della metodologia.

In base a quanto analizzato finora la metodologia Agile non è solo una questione di "moda" ma effettivamente persegue dei principi che si adattano molto bene all'attuale ecosistema, che si è stravolto con l'impiego massivo della tecnologia. Non è possibile affermare con certezza che un progetto gestito in modalità agile sia in assoluto preferibile rispetto alla gestione per il tramite di un approccio tradizionale, quello che emerso che esistono una serie di parametri da considerare nella scelta che indicano quale approccio riesce a trattare quella determinata caratteristica in maniera più rigorosa. A volte potrebbe essere opportuno anche optare per soluzioni miste o customizzazioni che meglio vadano incontro alle specifiche necessità ma la cosa importante è considerare le raccomandazioni sugli obiettivi, il team e lo scope che non possono essere considerate come accessorie.

Bibliografia e Sitografia

- Tanel Tenso, Alexander Horst Norta - Enhancing Requirements Engineering in Agile Methodologies by Agent-Oriented Goal Models, IEEE 2017
- Ekaterina Nemkova - The impact of agility on the market performance of born-global firms: An exploratory study of the 'Tech City' innovation cluster, Journal of Business Research 80 (2017) 257–265
- Cezary Orłowskia - Quantitative Assessment of the IT Agile Transformation, Cezary Orłowski et al. / Procedia Engineering 182 (2017) 524 – 531
- Agile project management approach and its use in big data management, Patrícia Franková et al. / Procedia Computer Science 83 (2016) 576 – 583
- Does Agile work? A quantitative analysis of agile project success, P. Serrador, J.K. Pinto / International Journal of Project Management 33 (2015) 1040–1051
- Moving from traditional to agile software development methodologies also on large, distributed projects, Georgios Papadopoulos / Procedia - Social and Behavioral Sciences 175 (2015) 455 – 463
- Manager della qualità, E. Leonardi, Egea 2015
- Method for Adaptation and Implementation of Agile Project Management Methodology, Arturs Rasnacis and Solvita Berzisa / Procedia Computer Science 104 (2017) 43 – 50
- Mixed agile/traditional project management methodology – reality or illusion?, Mario Špundak / Procedia - Social and Behavioral Sciences 119 (2014) 939 – 948
- Innovare i sistemi di controllo, M. Morelli L. Zoni, Egea 2013
- Metodi quantitativi per il project management, L. Bianco M. Caramia, Hoepli 2010
- Alistair Cockburn, Addison Wesley - Agile Software Development
- A study of value in agile software development organizations, H. Alahyari et al. / The Journal of Systems and Software 125 (2017) 271–288
- Il valore strategico della comunicazione nel project management, A. Bassi M. Tagliafico
- An exploratory study of waste in software development organizations using agile or lean approaches: A multiple case study at 14 organizations, H. Alahyari et al.
- Agile Software Development Process at Financial Institution in Kosovo, Edmond Hajrizi et al. / IFAC-PapersOnLine 48-24 (2015) 153–156
- Extreme Programming and Agile Processes in Software Engineering, P. Abrahamsson M. Marchesi, Giancarlo Succi (Eds.)
- Analysis on the Developing Trend of China's Banking Industry, Qu Wei-ming, Sun Ying-na

- Agile Compass: A Tool for Identifying Maturity in Agile Software-Development Teams, R. M. Fontana, S. Reinehr, A. Malucelli
- Defect Analysis in Large-Scale Agile Development Quality in the Agile Factory Model, Bernard Doherty
- Digital Technologies in Transformation of Classical Retail Bank into Digital Bank, Mirko Saji, Du**š**anka Bundalo, Zlatko Bundalo, Dra**ž**en Pa**š**ali
- Technological Innovations in the banking sector in India: an Analysis, Malini A, Dileep G Menon
- Agile Software Development: Adaptive Systems Principles and Best Practices, P. Meso, R. Jain
- Agile vs. structured distributed software development: A case study
- Hans-Christian Estler, M. Nordio, C. A. Furia, B. Meyer, J. Schneider, Springer Science Business Media New York 2013
- The "real" success factors on projects, Terry Cooke-Davies
- Extreme Programming and Agile Software Development Methodologies, L. Lindstrom, R. Jeffries
- Empirical studies of agile software development: A systematic review, T. Dyba, T. Dingsoyr
- A comparison of lifecycles - Agile software processes vs. projects in non-Agile software companies, S. Saarnak, B. Gustafsson
- I 10 grandi temi che guidano le banche verso la digital transformation, Mauro Bellini
- Digital Banking: le sfide per il sistema bancario tra esigenze dei consumatori, rivoluzione digitale e nuovi competitor
- A survey study of critical success factors in agile software projects, T. Chow, D.B. Cao / The Journal of Systems and Software 81 (2008) 961–971
- Software Metrics, Everald E. Mills
- Introduction to Function Points, Tom Cagley
- Empirical Study of Agile Software Development Methodologies: A Comparative Analysis, G. S. Matharu, A. Mishra
- Survey on Agile Metrics and Their Inter-Relationship with Other Traditional Development Metrics, S. Misra, M. Omorodion
- Harvard Business Review (edizione settmebre-dicembre 2015)
- http:// www.intesasanpaolo.com/intranet
- http://intranet.intesasanpaolo.com/scriptIni20/web/direzione-centrale-sistemi-informativi
- Lean Product and Process Development, A. C. Ward, Lean Enterprise Cambridge
- Agile scheduling and Control for Industrial Product Service Systems, H. Meier, E Uhlmann, N. Raue, T. Dorka
- https://airbrake.io/blog/software-design/behavior-driven-development (Behaviour Driven Development)

- https://en.wikiversity.org/wiki/Crystal_Methods
- https://books.google.it/ (Crystal Methods)
- https://www.impactmapping.org/ (Impact Map)
- https://www.amazon.it/Impact-Mapping-software-products-projects-ebook/dp/B009KWDKVA (Impact Map)
- https://www.mountaingoatsoftware.com/agile/user-stories (Impact Map)