Università degli Studi di Padova

Corso di Laurea in Informatica

Sviluppo e Gestione dei Progetti
A.A. 2010-2011

Docente: Dott. Filippo Ghiraldo

Best Practice nel Risk Management

proposto dal gruppo di lavoro Team42WebDivision

Bazzega Mattia - 594345

Favaro Sebastiano - 592519

Gonzo Nicola - 594890

Grigorcea Alexandr - 591543

Pasa Luca - 593258

team42webdivision@googlegroups.com

Indice

1 Abstract

Nel presente documento sono esposte delle linee guida per una gestione efficiente dei rischi. In particolare esso vuole essere un compendio delle migliori best practices finora adottate in ambito IT. Al termine del documento sono anche presentati elementi riguardanti lo sviluppo agile e TSP. Essenziali sono dunque le fonti che sono state consultate per poter determinare tali pratiche: esse vengono riportate al termine del documento, nella sezione bibliografia, e sono citate nel suo corso tra parentesi quadre. Poiché in genere non è possibile determinare la fonte di ogni singola frase, tutte le fonti utili alla stesura di un paragrafo o di una porzione di testo sono indicate alla fine di tale porzione. Talvolta è stato fatto riferimento preciso alle fonti per rimandare ad approfondimenti all'interno di esse.

2 Best practices

Le best practices sono delle attività riutilizzabili che aggiungono valore al progetto, aumentandone le probabilità di successo. Esse spaziano in numerosi campi, offrendo un vasto raggio d'azione. In sede del presente documento ci si concentrerà sulle best practices che permettono di migliorare la gestione dei rischi all'interno di un progetto IT.
Una caratteristica essenziale per una best practice è non essere statica, bensì dinamica. Questo perché nel corso del tempo ciò che una volta era il miglior modo per affrontare un problema potrebbe divenire addirittura controproducente. Per questo è necessario monitorare continuamente le best practices individuandone i punti deboli e contestualizzandole alla situazione in cui vengono applicate. Esse hanno dunque una dimensione aziendale in quanto ogni singola realtà può individuare delle pratiche proprie che rispondano nel migliore dei modi ai problemi che si manifestano. L'esperienza è dunque il vero cuore delle best practices:

"*Good practices come from experience. Experience, unfortunately, generally comes from unsuccessful practices and mistakes. People learn what not to do, all too often, by doing it and then suffering the consequences. Experience can be an invaluable resource, even when it is not your own.*" [4]

Affinché le best practices siano davvero utili per il progetto, per il processo e per l'azienda, è necessario che esse siano conosciute e applicate da tutti gli addetti ai lavori. È quindi utile formare il personale all'utilizzo di tali norme. Solo in questo modo le best practices permettono di ridurre i rischi del progetto, la perdita di capitali e di competitività.

3 Best Practice per la gestione dei rischi

La gestione dei rischi può essere divisa in alcune attività che sono state individuate all'interno di letteratura specializzata, tali attività sono qui presentate e spiegate dettagliatamente, per poter rappresentare delle linee guida da seguire nella gestione dei rischi. Il fatto che siano state reperite all'interno di documentazione di riconosciuta importanza a livello accademico, assicura che ciò che viene riportato corrisponda a best practice.

L'attività di gestione dei rischi è divisa in diverse attività secondarie. Tali attività (a loro volta suddivise in sotto-attività sono) sono:

- Analisi del contesto
- Determinazione dei rischi
  - Identificazione dei rischi
  - Quantificazione dei rischi
- Trattamento dei rischi
- Controllo dei rischi
  - Monitoraggio dei rischi
  - Comunicazione e consultazione riguardo i rischi

Fonti: 1,2,3.

3.1 Analisi del contesto

Questa è un'attività importante e deve essere eseguita all'inizio nel processi di gestione dei rischi. Analizzare il contesto vuol dire stabilire i limiti in cui i rischi vanno gestiti e definire lo scopo, come pure il budget per le attività di RM successive. Questa parte è essenziale per gli steps successivi e la sua funzione rimanda sia alla fase di valutazione sia a quella di trattamento, in quanto può permettere un'identificazione più completa, una migliore valutazione, e una selezione migliore delle strategie di risposta. Gli attributi rilevanti per l'analisi sono:

- Trovabilità dei fattori di rischio, per dire se è facile o difficile trovare il fattore di rischio;
- Responsabilità delle azioni, per dire chi se ne deve occupare, attori interni o esterni;
- Fasi dei cicli di vita del progetto, cioè i momenti in cui i fattori di rischio possono accadere;
- Controllabilità, cioè la capacità di influenzare la probabilità di occorrenza del rischio;
- Dipendenza, cioè il grado di dipendenza di un fattore di rischio dagli altri.

Fonti 1,5.

3.2 Determinazione dei rischi

3.2.1 Identificazione dei rischi

Lo scopo principale è determinare quali sono i giusti rischi da mitigare, alcuni rischi sono ovvi da trovare principalmente perché sono molto evidenti, altri meno.

Il comune approccio per il risk management enfatizza il bisogno di identificare i rischi presto. La reale sorgente di rischi sono quelli non identificati, quindi la fase iniziale di identificazione può essere considerata come un'iniziale risposta ai rischi.

Esistono 10 diversi tipi di metodologie di identificazione dei rischi, classificati in tre categorie, qui vengono presentati brevemente, ma per un approfondimento si può rimandare a [5].

- Analisi di primo ordine: Questo è un tipo di analisi semplice, ci permette di notare i fattori critici velocemente, solo scopo è di creare una mappa per individuare i rischi critici, si divide in tre modalità:
  - Selezione dei rischi;
  - Mappa a Quadrante;
  - Lista dei dieci rischi principali;
- Distribuzione dei rischi, è utile a stabilire come sono distribuiti i rischi, in genere ha come output dei grafici a torta, si divide in tre modalità:
  - Interni ed esterni;
  - Progetto, prodotto e processo;
  - Segnatura dei processi di rischio;
- Analisi di secondo ordine: Tale tipo di analisi permette di determinare l'ambito in cui si verificano i rischi, per poter determinare i rischi correlati, le attività correlate e tutto ciò che concerne il rischio direttamente ed indirettamente.
  - Analisi temporale;
  - Analisi causale;
  - Mappa dei processi;
  - Mappa ad area delle performance.

Esistono diversi modi per identificare i rischi.

Brainstorming.
Questo è l'approccio migliore per identificare i rischi, in quanto stimola le capacità creative dei partecipando riducendo la probabilità di tralasciare qualcosa. È molto utile per l'identificazione iniziale di un gran numero di rischi, particolarmente per grandi progetti, lo scopo è di trovare tutti i rischi. Il limite ideale della lista dei rischi è di circa 10 per non spendere troppa fatica sulle cose meno significative, anche se in generale è meglio trovare più rischi che meno. I partecipanti al brainstorming devono essere scelti accuratamente: devono avere molta esperienza, ed includere esperti da tutte le aree del progetto, se fosse possibile il brainstorming dovrebbe includere tutte le persone del progetto, perfino personale interessato al progetto ed esterno all'azienda.

Analisi delle esperienze su attività o progetti simili
Il processo di brainstorming può essere potenziato concentrandosi sui progetti passati simili, notando quali problemi sono stati affrontati e quali sono stati evitati.

Liste di controllo

Le liste di controllo sono costituite da una lista di rischi identificati preventivamente e da controllare. In particolare, le liste di controllo, sono efficaci se l'azienda ha una grossa esperienza nell'ambito del progetto che si sta realizzando e il progetto è piuttosto standardizzato e normale rispetto alle abitudini dell'azienda. Infatti può essere un problema per i progetti non abituali, a livello che può porre limitazioni alla creatività del brainstorming. Per progetti non abituali è preferibile iniziare con il brainstorming, e nel caso di bisogno suggerire elementi provenienti dalla checklist per stimolare i partecipanti.

Altre modalità che non vengono discusse nel dettaglio sono:

- Intervisti e gruppi di discussione;
- Analisi degli scenari;
- Sondaggi e questionari;
- Analisi delle WBS.

L'identificazione può essere utile per comprendere gli effetti sul progetto, e le cause del fallimento.

Alcune categorie in cui possono essere catalogati i rischi e che possono fornire una guida per iniziare le attività suddette sono:

- Selezione inadeguata del pacchetto software;
- Poche capacità del team di progetto;
- Sistema di comunicazione inefficace;
- Scarso coinvolgimento degli utenti chiave;
- Scarsa esperienza;
- Complessità del sistema da realizzare;
- Inadeguatezza dei processi di reingenierizzazione;
- Gestione non buona;
- Tecniche di gestione di progetto inadeguate;
- Gestione dei cambiamenti inadeguata;
- Sistema di gestione legale inadeguato;
- Servizi di consultazione inefficaci;
- Cattiva leadership;
- Performance inadeguate dei sistemi IT;
- Manutenibilità inadeguata dei sistemi IT;
- Pianificazione strategica inadeguata;
- Inadeguata gestione finanziaria.

Fonti 1,5,7.

3.2.2 Quantificazione dei rischi

La quantificazione dei rischi ha la scopo di valutare il livello di rischio tramite fattori che possano servire per sintetizzare la valutazione, per determinare la priorità e la selezione di strategie di trattamento.
In questo senso due componenti sono fondamentali: la probabilità di occorrenza, e l'impatto dell'occorrenza sul progetto.

Esistono tre approcci alla classificazione dei rischi:

- Qualitativo: utilizza parole per descrivere la grandezza delle conseguenze, e la probabilità della loro occorrenza
- Semi-Quantitativo, dove la scala qualitativa è impostata con valori dati, con lo scopo di produrre una classificazione più espansa, che in genere è ottenuta nella classificazione qualitativa, ma che però non è così precisa come in quella quantitativa
- Quantitativo, usa valutazioni numeriche. Sia per probabilità che per magnitudine.

Classificazione qualitativa
L'impatto di un rischio può essere espresso come una combinazione delle sue conseguenze o impatti sugli obiettivi del progetto. Le conseguenze sono valutate in termini del potenziale impatto sul progetto, in genere vengono individuate due tabelle, una per determinare il livello di probabilità ed una per determinare il livello di impatto. Le tabelle sono le seguenti:

Termine	Probabilità
Quasi certo	>80%
Probabile	80% - 50%
Possibile	50% - 10%
Poco Probabile	10% - 2%
Raro	<2%

Termine	Descrizione
Catastrofico	Rischio davvero estremo che porta a grandi costi aggiuntivi, grandi ritardi e significativa compromissione della reputazione dell'azienda.
Massimo	Rappresenta un evento critico, che può portare a costi aggiuntivi e ritardi.
Moderato	Grande impatto ma che può essere gestito con diversi sforzi.
Minimo	Impatto che può essere gestito con metodi di routine.
Insignificante	Impatto che può essere ignorato.

Dalla combinazione delle probabilità e dell'impatto si può dedurre la priorità del rischio secondo una tabella come la seguente:

Probabilità \ Impatto	Insignificante	Minimo	Moderato	Massimo	Catastrofico

Quasi certo	Media	Media	Alta	Alta	Estrema
Probabile	Media	Media	Media	Alta	Estrema
Possibile	Bassa	Media	Media	Alta	Alta
Improbabile	Bassa	Bassa	Media	Media	Alta
Raro	Bassa	Bassa	Media	Media	Media

Fonti: 1,7.

Classificazione Semi-Quantitativa
Questo tipo di classificazione si pone a metà tra quella qualitativa e quella quantitativa, infatti tale metodo prevede la trasformazione da scale qualitative in scale numeriche che possano indicare direttamente tramite indici numerici probabilità, impatto e priorità. In questa modalità viene effettuata la conversione da termini, o vocaboli, simili a quelli dell'analisi qualitativa a valori numerici. Vengono definiti due valori numerici, uno per la probabilità ed uno per la magnitudine delle conseguenze. Tali valori definiti rispettivamente P e C sono numeri compresi tra 0 ed 1.

A questo punto esistono diversi modi per calcolare il fattore dei rischio.

Modo 1

$$\text{Fattore di rischio} = RF = P+C - P*C.$$

Questo permette di ottenere diverse misure dei rischi, tutte comprese tra 0 ed 1. In tal modo è possibile classificare i rischi secondo i parametri ISO, secondo il quale i rischi che hanno un fattore di rischio compreso tra 0,6 e 0,8 sono quelli prioritari. Successivamente è possibile stilare una lista completa ordinando i rischi per fattore di rischio.

Modo 2

$$\text{Fattore di rischio} = RF = P*C$$

Questa è un'altra modalità di calcolo piuttosto comune, ed in questo caso non è necessario restringere P e C all'intervallo 0 e 1. Il problema di questo tipo di calcolo, rispetto a quello precedente è che valori bassi di probabilità per rischi alti, o viceversa portano a fattori di rischio molto bassi, e quindi potrebbero non essere considerati come importanti. Pertanto la modalità raccomandata è quella spiegata precedentemente. Fonte: 7

Classificazione Quantitativa
La valutazione quantitativa interviene quando riusciamo a specificare in maniera piuttosto precisa i costi e le conseguenze di un rischio. L'analisi quantitativa è molto più precisa e rivela molto di più sul rischio, ma è pure più difficile da realizzare. L'analisi quantitativa permette di determinare precisamente le risorse da allocare per i progetti a rischio. Non tutti i rischi sono ugualmente dannosi, e tentare di stimare il danno è utile solo quando lo si riesce a fare in maniera precisa. Per la classificazione quantitativa possono essere utilizzate tabelle oppure fogli di calcolo. In particolare per tale tipo di analisi va individuato per ogni rischio una o più metriche precise riferite al tipo di rischio, e tali metriche sono utili a definire il livello di tale rischio, e conseguentemente i costi che possono derivarne. Pertanto è necessario definire preventivamente le metriche, ed le modalità di interpretazione di tali metriche definendo conseguentemente quali

azioni intraprendere in base ai diversi valori delle metriche. A tale scopo è utile definire alberi decisionali, che ad ogni range di misurazioni possono associare degli effetti per poter valutare i casi peggiori e migliori, ed avere sempre presente cosa dev'essere realizzato nei diversi casi. L'analisi quantitativa è uno degli aspetti più complessi del risk management e per una sua trattazione esaustiva si rimanda a [7] e [11].

Fonti: 11, 8, 7.

3.3 Trattamento dei rischi

Lo scopo di questa attività è quello di creare una politica che permetta di gestire il rischio. Esistono diversi modi per trattarlo:

1. Diminuirne la probabilità. Un buon mezzo per minimizzare l'eventualità che certi rischi si manifestino sono i contratti. Nel caso l'azienda in oggetto abbia necessità di un particolare prodotto o servizio è possibile attraverso di essi vincolare i fornitori a garantire un'alta qualità dello stesso e la certezza che esso rispetti le aspettative. In questo modo si può quasi annullare il rischio che la mancanza del prodotto o sevizio possa compromettere la pianificazione prevista delle attività.

2. Ridurne l'impatto sul progetto. Certi rischi come quelli climatici o quelli relativi alle variazioni economiche non possono essere evitati e molto spesso non è neanche possibile diminuire la probabilità che essi avvengano in quanto non sono sotto il diretto controllo dell'azienda. In questi casi è però possibile attuare delle strategie che mitighino le conseguenze del rischio nel caso esso si concretizzi. Un buon modo per seguire questa linea è quello di stipulare polizze assicurative.

3. Trasferire l'onere della gestione del rischio a servizi di terze parti. Ciò ovviamente porta ad altri costi per l'azienda, ma può garantire buoni risultati affidandosi a persone di competenza. È utile sottolineare che spesso tale gestione del rischio non viene trasferita completamente a terzi bensì viene condivisa con essi in quanto l'azienda vuole comunque poter osservare e partecipare alla gestione per avere maggiori garanzie e tranquillità. Tale forma è spesso indicata con il nome di *risk sharing*.

4. Mantenere il rischio. In certi casi non è possibile attenuare le probabilità o l'impatto del rischio magari perché i costi sarebbero eccessivi o i benefici irrisori.

Il deliverable di questa fase è un RMP (Risk Management Plan) con elencate tutte le strategie scelte per affrontare il rischio.
Per individuare le soluzioni possibili ad un rischio è consigliabile seguire queste vie alcune delle quali sono state già esposte nel corso del presente documento:

1. Brainstorming;

2. Analisi storica dei progetti passati

3. Riunioni con il personale, in particolare con quei membri del team che verrebbero direttamente colpiti nel loro lavoro se il rischio dovesse concretizzarsi.

È utile individuare più di una soluzione per ogni rischio. Quindi per ognuna di esse bisogna individuare i costi per la sua adizione, i vantaggi e gli svantaggi che apporta. Nel documento RMP è consigliabile scrivere per ogni rischio tutte queste possibili vie che sono state trovate, indicando qual'è quella migliore e per quali motivi. Infine si devono assegnare le responsabilità e allocare le risorse necessarie per attuare la strategia scelta.

Le best practices che sono state individuate per la suddetta fase sono le seguenti:

1. Aggiornare periodicamente l'RMP. Esso è infatti un documento in continuo divenire anche una volta che è divenuto ben strutturato in quanto anche i rischi variano molto spesso.

2. Per la realizzazione dell'RMP è utile guardare alla storia dell'azienda, controllando quali siano state le soluzioni migliori ai rischi del passato.

3. Per redigere l'RMP è utile appoggiarsi a persone che abbiano competenza nel settore a cui appartiene il rischio, in modo da svilupparne una gestione quanto più fruttuosa possibile.

4. Trattare prima di tutto i rischi che si ritengono distruttivi, anche se la loro probabilità di manifestarsi è molto bassa. Solo di conseguenza è bene trattare i rischi molto probabili.

   Fonti: 8.

3.4 Controllo dei rischi

Questa attività ha lo scopo di attuare ed incrementare l'efficacia e il rendimento della politica di risk management adottata. In particolare si vuole far in modo di migliorare continuamente le soluzioni adottate per fronteggiare i vari rischi.
Una buona pratica è quella di valutare se le strategie di gestione dei rischi introducano loro stesse altri rischi che devono essere controllati.

3.4.1 Monitoraggio dei rischi

In questa fase lo scopo principe è controllare la situazione del rischio, così da attuare la giusta politica non appena si manifestino i suoi primi sintomi. Inoltre nel caso il rischio in questione modifichi la sua natura cambiando probabilità ed impatto, è necessario aggiornare la politica con cui viene gestito, al fine di non farsi cogliere di sorpresa.
Una buona pratica per questa fase è quella di ottenere dei feedback su come è stato gestito il rischio, così da migliorare continuamente il RM adottato. Per realizzare ciò è utile inserire tale attività all'interno di un processo PDCA di miglioramento continuo. In quest'ottica ogni qual volta si profila all'orizzonte un possibile rischio, si deve pianificare (Plan) una strategia di gestione dello stesso. Quindi deve essere applicata (Do) la soluzione individuata che può mitigare l'impatto del rischio e/o minimizzarne la probabilità di avvenimento. Completata quest'ultima fase si devono valutare i risultati ottenuti (Check) per capire cosa è andato bene e cosa invece deve essere migliorato. Infine è necessario attuare le modifiche stabilite (Act) migliorando così la gestione del rischio.
Le best practices per questa attività sono le seguenti:

1. Iniziare a monitorare i rischi solo quando si ha veramente una politica di gestione degli stessi ben strutturata e organizzata. Essa verrà inserita nel RMP, documento che deve essere approvato sia dal leader del progetto che da chi lo finanzia.

2. Avere un buon sistema informatico che permetta di tracciare correttamente i rischi.

3. Effettuare un tracciamento della situazione in modo periodico, magari a grana settimanale. Se il progetto risulta molto vasto o addirittura composto da molti team il quantitativo di materiale prodotto può risultare molto importante, è quindi utile avere un addetto che periodicamente riassuma la situazione generale dei rischi evidenziando i punti chiave di tutto ciò che è stato inserito nel sistema informatico. È quindi comodo effettuare periodicamente delle riunioni di aggiornamento nelle

quali informare gli stakeholders sullo stato del progetto. Un report su un progetto potrebbe essere così composto:

(a) Breve descrizione sulle conquiste e le milestone raggiunte dall'ultimo report. In questo punto è anche intelligente citare le persone che maggiormente si sono distinte raggiungendo risultati degni di nota. In questo modo si motivano i componenti.

(b) Attività previste per il prossimo periodo.

(c) Problemi riscontrati e soluzioni proposte.

(d) Risorse necessarie attualmente e stime per quelle che potrebbero essere utili nel futuro.

(e) Nuovi rischi che sono stati individuati.

Fonti: 4

3.4.2 Comunicazione e consultazione riguardo i rischi

Una buona pratica per gestire i rischi è quella di comunicare con tutti gli stakeholders del progetto lo stato attuale dello stesso. In questo modo ognuno è aggiornato in tempo reale sulla situazione e può avvisare gli altri sull'evolversi di un rischio. È quindi comodo stabile un canale di comunicazione privilegiato che venga periodicamente controllato da tutti.
L'informazione che viene condivisa sui rischi è bene che riguardi i seguenti punti su di essi:

- Esistenza di un rischio
- Natura;
- Forma;
- Probabilità;
- Impatto;
- Se si tratta di un rischio accettabile;
- Modo con cui il rischio viene trattato.

Senza una forma di comunicazione i rischi di un progetto rischiano di non essere identificati causando gravi problemi.
La comunicazione può rivelarsi molto difficile da realizzare in quanto essa è affetta da numerose barriere come la distanza, la lingua, la cultura, il tempo. A complicare ulteriormente questo scenario c'è il problema di fondo che spesso le persone cui cui è necessario comunicare non conoscono il dominio del discorso in quanto hanno una formazione diversa dalla nostra. Per questo è necessario adottare delle best practices quando si comunica, in particolare quando si tratta di RM. Alcuni suggerimenti che si possono dare sono:

1. utilizzare una forma chiara, e precisa. Solo in questo modo è infatti possibile valicare le ambiguità intrinseche del linguaggio naturale. Così facendo si possono limitare le incomprensioni, minimizzando i chiarimenti necessari in futuro: esporre chiaramente qualcosa fa perdere tempo nell'immediato, ma garantisce di guadagnarne a lungo termine.

2. Se il progetto ha un respiro internazionale è bene che tutti i documenti vengano redatti in inglese. Viceversa è meglio utilizzare la propria lingua, in questo modo si minimizzano gli errori e si velocizza il processo di redazione.

3. È sempre comodo avere un luogo online nel quale tutte le informazioni vengano costantemente aggiornate. Ciò permette di avere 2 grossi vantaggi

   (a) Un backup dei documenti

   (b) La possibilità da parte di tutti di accedere a tali informazioni senza barriere geografiche o temporali grazie alla facilità con cui al giorno d'oggi è possibile accedere alla rete.

4. Adottare una comunicazione periodica imponendo degli aggiornamenti fissati in modo che tutto l'evolversi della politica di RM sia sempre ben tracciata.

5. Essere brevi, ovvero fornire un'informazione ricca di contenuti e chiara, ma non prolissa in dettagli inutili: non è importante la quantità delle informazioni, ma la qualità delle stesse.

   Fonti 1,6

4 Metodologie di sviluppo del software

Esistono molte possibili vie per realizzare un progetto IT, in questa sezione esponiamo esclusivamente quelle Agile e TSP come richiesto dal progetto.

4.1 Agile

Le metodologie agili nascono alla fine del '90 come reazione ai modelli di ciclo di vita precedentemente utilizzati e caratterizzati da un eccessivo uso di documenti. Essi si basano su questi quattro principi:

1. Dare più importanza agli individui e ai membri del team piuttosto che ai processi. In questo modo si favorisce l'emergere del valore di ognuno, garantendo un miglior risultato finale;
2. È più importante avere un software che funziona piuttosto che una documentazione ben fatta. Se ne deduce che il tempo impiegato per documentare è ridotto al minimo;
3. La collaborazione con il cliente è essenziale per realizzare ciò che veramente gli è necessario;
4. Adattarsi ai cambiamenti è essenziale, anche a costo di violare il piano stabilito.

Un altro concetto che sta alla base dei metodi agili è quello di ''user story'', che indica un compito significativo che l'utente vuole svolgere con il software richiesto. Ogni *user story* è definita da tre elementi che si possono elencare di seguito:

- un documento di descrizione, che riporta ciò che bisogna fare pezzo per pezzo;
- una raccolta di tutti i dialoghi con gli stakeholder per fissare la comprensione comune;
- una modalità per dire che quello che è stato fatto è giusto, cioè per confermare che il software realizzato soddisfi gli obiettivi;

L'agile tenta dunque di minimizzare la burocrazia favorendo lo sviluppo di un progetto migliore. Ci sono però alcuni aspetti negativi che fanno da sfondo a questo ciclo di vita:

1. Dando maggior importanza alle persone che ai processi, si ritiene che tutti i membri del team siano molto competenti nel loro settore, cosa molto difficile;
2. Minimizzare la documentazione è un grosso pericolo in quanto essa risulta spesso essenziale per portare a termine un buon progetto. Ad esempio per poter procedere velocemente e con sicurezza nella codifica è bene avere una progettazione realizzata correttamente e documentata con precisione. Discorso ancora più importante vale per la verifica e la validazione del software: esporre in un documento tutti i test che si sono fatti permette di comprendere meglio quanto si sia fatto e quanto si dovrebbe ancora fare. Inoltre tale documento è presentabile al cliente che quindi può meglio convincersi della qualità del progetto realizzato;
3. Esigere un stretta collaborazione con il cliente spesso è pura utopia in quanto egli stesso molto raramente si dimostra così disponibile o competente nell'ambito dello sviluppo;
4. Non seguire un piano o violarlo molto spesso può causare dei ritardi.

Lo sviluppo agile ha quindi i suoi pregi, ma anche i suoi difetti. È necessario decidere con attenzione se adottare tale ciclo di vita.
Possiamo suddividere l'agile in due filoni diversi:

- metodologie per l'organizzazione del lavoro. In questa sede ricadono principalmente *Scrum* e *Kanban* che hanno la caratteristica di poter essere applicate praticamente a qualunque processo produttivo, non solo a progetti IT.
- metodologie di sviluppo software che si affiancano a quelle del punto precedente, ma sono strettamente pratiche e legate appunto a progetti IT. La più famosa è l'*eXtreme Programming*.

L'agile è dunque un termine che racchiude al suo interno numerose sfaccettature e movimenti diversi. Nonostante ciò possiamo già qui individuare due best practices che garantiscono buoni risultati per qualunque progetto che adotti queste metodologie.

Sviluppo prototipale La realizzazione di un progetto in modo Agile prevede la creazione di molti prototipi, che per incrementi successivi divengono il software desiderato dal cliente in base alle sue correzioni. Ciò porta a numerosi vantaggi come esplorare meglio i requisiti che magari emergono in corso d'opera e trovare diverse soluzioni al problema. Gli studi sui prototipi permettono inoltre di migliorare l'usabilità e l'architettura dell'applicazione, cosa difficilmente ottenibile a prodotto finito. Un altro vantaggio di questo tipo di sviluppo è la possibilità di esercitare gli utenti all'utilizzo del sistema prima ancora che esso sia terminato, ottenendo così un loro feedback.
Lo sviluppo prototipale ha però il limite di avere costi molto maggiori nelle fasi iniziali che possono essere ammortizzati in fase di manutenzione. Essa dovrebbe essere infatti molto ridotta se i prototipi realizzati sono confluiti in un buon progetto finale.
Questa tecnica permette quindi di gestire meglio i rischi legati ai requisiti, in particolare se essi sono ambigui o difficili da capire.

Mago di Oz Una best practices per creare dei prototipi molto velocemente e a basso costo è quello di creare dei modelli dell'interfaccia grafica su carta proponendoli agli utenti che poi esprimono il loro parere e danno preziosi suggerimenti. Un'estensione di questa tecnica è il prototipo ''mago di Oz'' dove viene realizzata solo un'interfaccia grafica con la quale l'utente può interagire, ma si tratta di una semplice simulazione, infatti le richieste sono inviate ad una persona che le interpreta e restituisce quanto desiderato dall'utente. Questa tecnica permette di ridurre il rischio di creare una grafica poco comprensibile per l'utenza finale che quindi risulta poco soddisfatta, introducendo alti costi per migliorare l'esperienza d'uso del prodotto finale.

4.1.1 Metodologie per l'organizzazione del lavoro

Iniziando a parlare del modello agile *Scrum*, si può dire che esso è un approccio innovativo per completare dei progetti complessi. In origine, tale modello era stato progettato per i progetti di sviluppo software, ma si applica al meglio anche per qualsiasi altro ambito di lavoro.
Per spiegare tale framework, si può innanzitutto considerare la seguente figura che illustra macroscopicamente il suo funzionamento.
Come si può osservare, in primo luogo il team di progetto crea una lista di priorità delle cose da fare (*product backlog*) per portare a termine l'obiettivo prefissato. Questo primo passo è necessario in quanto si comprende il problema tramite l'equivalente di *user story*. Dalla cima di questa pila viene preso un sottoinsieme delle cose da eseguire (*sprint backlog*) e lo si pone dentro ad un'agenda dove si lavora a ciclo rapido. Si

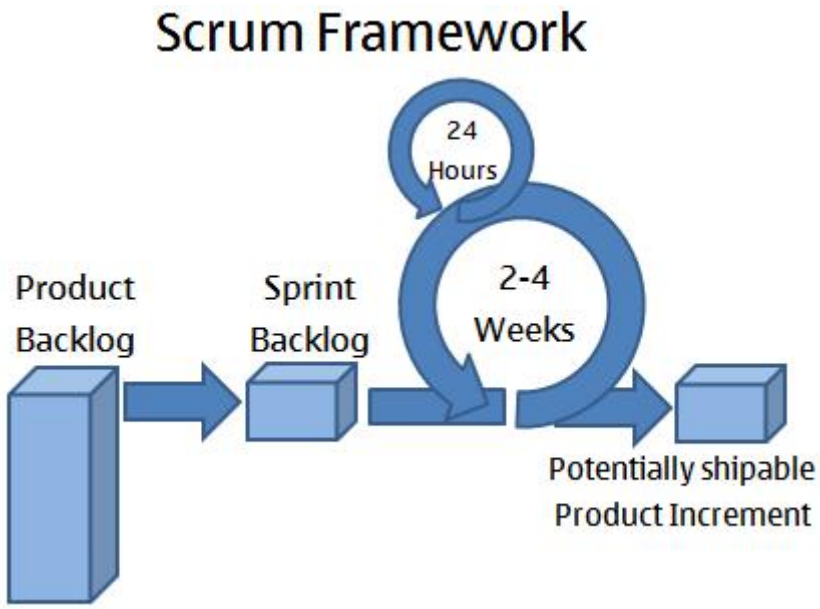

Figura 1: Funzionamento del modello agile *Scrum*.

innescano, quindi, due cicli: uno rapido (quotidiano) e un altro che è al più mensile. Il team di progetto, dunque, ha una certa quantità di tempo (*sprint*) per completare questa parte di lavoro, generalmente 2 o 4 settimane, ma di questa ne viene accertato ogni giorno il progresso (*daily scrum*).
Il ciclo principale (*ScrumMaster*) ha il compito di tenere concentrato il team sull'obiettivo prefissato. Il ciclo rapido, invece, raccoglie l'esperienza del giorno; tale esperienza viene distribuita a tutti gli altri componenti del team di lavoro. Il suo scopo è dunque quello di porre nel backlog un pezzo di soluzione di cui ne è stata valutata la correttezza. Se non funziona, al ciclo rapido successivo vengono adottate delle opportune decisioni ed apportate le dovute modifiche.
Al termine del ciclo dalla durata di 2 o 4 settimane, il lavoro prodotto (incremento / deliverable) dovrebbe essere potenzialmente pronto per essere venduto ad un cliente o mostrato agli stakeholder. Il ciclo termina, inoltre, con una revisione di quanto fatto, retrospettiva, che passa in rassegna ciò che è stato prodotto passo passo.
All'inizio dello *sprint* successivo, il team seleziona un'altra parte di lavoro da eseguire, creando un sprint backlog, ed inizia a produrre nuovamente.
Questo ciclo si ripete finché tutti gli ''item'' del product backlog sono stati completati, il budget richiesto è esaurito, oppure è arrivata la scadenza del progetto. La variabile che determina la sua conclusione, è interamente specifica del progetto.
Si comprende inoltre che la pila è dinamica e di priorità. È dinamica perché i vari item possono essere cancellati o aggiunti in ogni momento durante il progetto, facendo sì che essa decresca o meno. Quando la pila finisce ogni pezzo aggiunto incrementalmente funziona. È di priorità, facendo in modo che gli item con priorità più alta vengano completati prima. È, inoltre, ''raffinata progressivamente'', nel senso che gli item a priorità minore sono volutamente meno dettagliati, più a grana grossa.
Come si nota, dunque, questo modello agile si fonda sul principio del ''caos organizzato'', proprio perché è fortemente iterativo.

Un'altra metodologia per l'organizzazione del lavoro è data dal modello *Kanban*. Quest'ultimo è un metodo giapponese per lo sviluppo agile, meno strutturato rispetto a *Scrum*.

È un approccio per incrementare e migliorare i processi e i cambiamenti dei sistemi per le aziende. Più precisamente, è un modello per introdurre delle modifiche attraverso dei miglioramenti incrementali. Per far ciò, vengono portate a galla delle problematiche riguardanti le operazioni o i processi del sistema e viene stimolata la collaborazione tra i membri del team per effettuare un miglioramento continuo sul sistema.

Il funzionamento della metodologia Kanban si basa su un sistema di tipo ‘‘pull’’ (detto anche modello *Just In Time*). Quest'ultimo si fonda sul concetto del supermarket. I clienti comprano i prodotti posizionati negli scaffali e gli operatori rimpiazzano i prodotti mancanti nelle quantità prestabilite e senza mai lasciare i clienti senza prodotti. Il processo funziona, quindi, al contrario del metodo di produzione ‘‘push’’, basato su grandi lotti stimati dalle vendite.
La parola kanban è un termine giapponese che letteralmente significa ‘‘cartellino’’ ed indica un elemento del sistema Just In Time di tipo pull di reintegrazione delle scorte mano a mano che vengono consumate.

Il lavoro viene organizzato tramite una *kanban board*, che sono dei tabelloni in cui vengono presentate delle *kanban card*. Queste ultime come già accennato in precedenza, sono essenzialmente dei cartellini con cui un reparto produttivo segnala al magazzino o ad un fornitore di essere a corto di materie prime, e chiede di essere rifornito. Nei progetti agile esse riguardano i compiti da svolgere e sono realizzate con dei post-it.
Si ha quindi un tabellone con una timeline, che nella sua forma più semplice può essere rappresentata da tre sole colonne, intitolate, ad esempio, attività da fare, attività in corso e attività terminate. I membri del team potranno aggiungere delle kanban card o cambiare colonna alle kanban card già presenti.
Si possono, inoltre, utilizzare varie kanban board per fornire informazioni sullo stato del progetto a diversi livelli. Ad esempio si potrebbe avere una kanban board per la release del prodotto, una per ciascuna iterazione, ed addirittura una giornaliera contenente le attività da svolgere al massimo livello di dettaglio.
Un esempio di kanban board costituita da diverse colonne può essere mostrato di seguito:

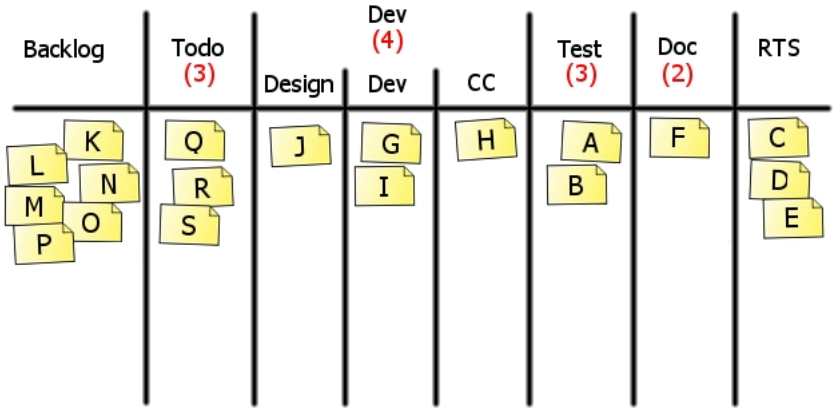

Figura 2: Esempio di *kanban board*.

Per applicare al meglio questo metodo agile, quindi, si può far riferimento a queste regole:

- visualizzare il flusso di lavoro (*workflow*):

  - suddividere il lavoro in parti (item), scrivere ogni item su una card e apporla sul muro;
  - utilizzare delle colonne che abbiano dei nomi per illustrare dove sia ciascun item all'interno del workflow.

- limitare il *Work In Progress* (*WIP*), cioè assegnare dei limiti espliciti su quanti item possono essere in lavorazione per ogni stato del workflow;
- misurare il *lead time* (tempo medio per completare un item, talvolta chiamato anche ''cycle time''), ottimizzare il processo per rendere il lead time quanto più piccolo e prevedibile possibile.

Similitudini tra *Scrum* e *Kanban* La relazione che esiste tra *Scrum* e *Kanban* è legata al termine *strumento di processo*. Quest'ultimo è composto da due parole molto significative: *strumento*, ossia qualunque cosa possa essere utilizzata come mezzo per realizzare un compito o un fine e *processo*, cioè il modo in cui si lavora.
Sia Scrum che Kanban sono infatti degli *strumenti di processo*, in quanto consentono di lavorare in modo più efficace, in una certa misura, dicendo cosa fare.
Altre similitudini tra i due modelli sono le seguenti:

- entrambi sono metodologie Agile, enfatizzando il rispondere ai cambiamenti piuttosto che seguire un piano;
- entrambi utilizzano una programmazione di tipo ''pull'';
- entrambi limitano il WIP (Work In Progress);
- entrambi promuovono un miglioramento dei processi, eseguendo un'ottimizzazione continua ed empirica degli stessi;
- entrambi si concentrano sulla realizzazione di software che sia rilasciabile rapidamente e spesso; in entrambi, il piano di rilascio viene costantemente ottimizzato basandosi su dati empirici (velocità / lead time);
- entrambi richiedono la suddivisione del lavoro in parti.

Differenze tra *Scrum* e *Kanban* Le due metodologie agile qui esposte presentano diverse differenze, che possono essere brevemente riassunte nella seguente tabella:

Scrum	***Kanban***
Prescrive iterazioni *timeboxed* (l'idea generale è infatti quella di mantenere la stessa lunghezza per tutte le iterazioni in un certo periodo di tempo e determinare così una cadenza).	Le iterazioni *timeboxed* sono opzionali. È possibile avere cadenze separate per la pianificazione, il rilascio e il miglioramento dei processi. Può essere *event-driven*, piuttosto che timeboxed.
Utilizza la *velocità* come metrica di default per la pianificazione e il miglioramento dei processi.	Utilizza il *lead time* come metrica di default per la pianificazione e il miglioramento dei processi.
Gli item devono essere ridotti in modo da poter essere completati all'interno di uno sprint.	Non viene prescritta nessuna grandezza particolare per gli item.

Il *WIP* viene limitato indirettamente (per sprint).	Il *WIP* è limitato direttamente (per stato di workflow).
Non è possibile aggiungere elementi a iterazione in corso.	Si possono aggiungere nuovi item ogni qualvolta ci sia capacità disponibile.
Una *Scrum board* viene resettata ad ogni sprint.	Una *Kanban board* è persistente.
Presenta un *product backlog* con priorità.	La priorità è opzionale.
Il team si impegna a realizzare una quantità di lavoro specifico nel corso dell'iterazione corrente.	L'impegno è opzionale.

Tabella 4: Differenze tra *Scrum* e *Kanban*.

Considerazioni sulle due metodologie citate Dopo aver dato una breve panoramica dei due metodi agile, si possono fare ora alcune considerazioni sull'uso dell'uno piuttosto che dell'altro.
Come si è potuto osservare, in pratica Kanban ha solo un paio di regole esplicite (visualizzare il workflow, limitare il WIP), mentre Scrum è molto più prescrittivo.
Si può dire dunque che Kanban si adatta più facilmente a team di lavoro e organizzazioni che normalmente resistono a pratiche agili più tradizionali (per cultura, passate esperienze fallimentari, ecc). Infatti tale modello non impone ruoli, non impone iterazioni, non forza gente a cambiare drasticamente la maniera in cui svolgono le loro attività correnti.
In questo contesto il cambiamento può essere profondo ma è un'evoluzione sottile ed organica. Il problema è che senza una guida esperta, il team non ha facili riferimenti o ''regole'' da seguire. La responsabilità ricade sul team.
Tuttavia un team ha comunque bisogno di ruoli definiti e di tutta una serie di attività che, seppur non prescritte, si rivelano essere utili.
Quindi un team agli inizi, può avere vita più facile partendo da Scrum, sempre che ovviamente la natura del lavoro non si adatti a Kanban (per esempio un lavoro non pianificabile, come la manutenzione).

fonti: 20, 21, 22.

4.1.2 Metodologie per lo sviluppo del software

L'XP (*eXtreme Programming*) è una metodologia di sviluppo software per team medio-piccoli che variano dalle 2 alle 10 persone. Essa viene teorizzata all'interno del libro *''Extreme Programming Explained: Embrace Change''* da Kent Beck dove si riporta:

''l'XP è un modo leggero, efficiente, a basso rischio, flessibile, predicibile, scientifico e divertente per sviluppare il software''

L'XP vuole offrire delle linee guida e delle pratiche per realizzare nello sviluppo quotidiano il manifesto agile. Viene utilizzato in particolare nei progetti in cui i requisiti sono molto volatiti e quindi il rischi di insuccesso è estremamente elevato, in quanto è facile sviluppare qualcosa che non soddisfi il cliente. L'aggettivo *extreme* vuole sottolineare come ogni aspetto del senso comune viene portato ai massimi livelli:

- i test vengono effettuati continuamente, così da migliorare la qualità interna dell'applicativo. Così si possono ridurre il numero di bug che si consegnano con esso, minimizzando i problemi che possono esserci con l'utenza finale e il lavoro di manutenzione.
- è risaputo che le revisioni del codice permettono di migliorare significativamente il prodotto finale, quindi l'XP propone di utilizzare la programmazione a coppie, in modo che sin dalla prima redazione, il codice sia già controllato da un'altra persona.
- si vogliono utilizzare al massimo le soluzioni più semplici in quanto risultano molto spesso quelle più efficienti ed efficaci.
- i test d'integrazione hanno dimostrato la loro grande utilità, per questo l'XP suggerisce di realizzarli più volte ogni giorno.
- le iterazioni brevi sono molto utili poiché permettono di avere obiettivi chiari, precisi e semplici da raggiungere, garantendo una prosecuzione sicura del progetto.

L'XP propone una struttura di valori, principi e attività. I primi sono ciò che dovrebbero guidare il team di sviluppo in tutti i progetti in cui è coinvolto. Essi hanno un respiro multidisciplinare ovvero potrebbero essere applicati ad ogni settore produttivo. I principi sono invece i valori intercalati nella produzione del software. Infine le attività sono delle linee guida che permetto di realizzare veramente i principi.

Valori I valori hanno lo scopo di dare una linea guida per definire dei criteri di sviluppo che permettono di rimanere sulla retta via. Essi sono

- **Comunicazione**: essa è essenziale per capire i cambiamenti che a diversi livelli possono avvenire. Ad esempio durante la codifica si possono fare delle scelte che potrebbero cambiare l'architettura, quindi i programmatori sono tenuti a discuterne con i progettisti. Al contrario i clienti potrebbero esigere nuove funzionalità che devono essere chiaramente esposte ai progettisti.
- **Semplicità**: è essenziale rendere le cose il più semplice possibile, anche nell'ottica dei costi. È meglio realizzare qualcosa di elementare adesso e pagare di più nel futuro per migliorarla piuttosto che spendere molto adesso per fare qualcosa che poi potrebbe non essere utilizzata. Ciò ovviamente vale solo se non si è sicuri che quello che si realizza sia definitivo. In uno sviluppo prototipale questa è veramente una regola aurea.
  C'è una grande relazione tra la comunicazione e la semplicità: più un sistema è semplice meno c'è da comunicare e con meno difficoltà, riducendo così le incomprensioni.
- **Feedback**: è essenziale per capire il vero stato attuale del sistema. In campo IT ciò può essere concretizzato adottando una politica di test molto seria, che garantisca la buona riuscita di quanto si sta sviluppando. Maggiori sono i feedback più risulta semplice comunicare per dimostrare che qualcosa va bene o meno. Ad esempio avere un test che dimostra un'errore di programmazione aiuta notevolmente la comunicazione tra le parti.
- **Coraggio**: è necessario per provare nuove vie che - a volte - possono generare una soluzione migliore.

Principi Essi derivano dai quattro valori esposti nel paragrafo precedente e vogliono concretizzarli nel lavoro quotidiano. Si possono individuare i seguenti principi:

- **feedback rapido**: quando un componente del team svolge un'operazione è necessario capire il prima possibile se essa ha un risultato positivo, così da applicarla nuovamente oppure da eliminarla o perfezionarla. Quindi il feedback deve essere la norma di ogni attività e deve avvenire appena possibile, subito dopo che l'attività si è conclusa.
- **adottare la semplicità**: bisogna trattare ogni problema in modo da ottenere delle soluzioni ridicole, che richiedano poco tempo per essere realizzate. Ciò va contro il principio del riuso tipico dell'ingegneria del software, che vuole realizzare le cose per il futuro. L'approccio dell'XP è però totalmente diverso: bisogna realizzare e fare un buon lavoro per il problema che oggi deve essere risolto. In futuro - se necessario - si potrà incrementare la complessità della soluzione. Così si abbassano i costi produttivi.
- **cambiamenti incrementali**: i grandi cambiamenti fatti tutti una sola volta non funzionano. bisogna adottare delle piccole modifiche, fatte una alla volta, realizzando così passo passo il grande cambiamento che si era programmato.
- **abbracciare il cambiamento**: non aver paura di modificare il sistema, se ciò porta a valore aggiunto.
- **qualità**: è essenziale non solo per consegnare un progetto ben fatto, ma anche per favorire il lavoro del team: è bello sviluppare per un sistema che funziona bene e che rende soddisfatte le persone che vi partecipano.
- **esperimenti concreti**: quando si devono fare delle scelte radicali è sempre bene verificare concretamente con delle prove che la nuova soluzione sia corretta.
- **responsabilità accettata**: nello sviluppo tradizionale il capo del team decide a priori chi deve fare cosa. In questo modo le persone si sentono frustate in quanto si viene obbligati a svolgere una data mansione. Una soluzione è esporre all'intero team ciò che deve essere fatto. Poi saranno i singoli membri a decidere cosa fare, spartendosi i lavori e accettando individualmente le responsabilità. Ciò permette anche di ottenere un lavoro di qualità superiore in quanto ognuno cercherà di scegliere - se possibile - i compiti che più lo interessano e quindi svolgerà un lavoro migliore.

Attività Nei progetti IT possiamo individuare 4 attività essenziali per la realizzazione del software. Esse sono:

- **codifica**: è ciò senza la quale il prodotto finale non esiste. È bene però sottolineare che una codifica per quanto buona possa essere, non sarà mai in grado di generare un prodotto di qualità. Essa è dunque solo una parte, una fase di un buon prodotto.
- **test**: sono essenziali per dimostrare che ciò che si è programmato è corretto e funziona. È utile realizzare due tipi di test: quelli di unità e quelli funzionali. I primi servono ai programmatori per verificare che il codice realizzato sia veramente quello che avevano in mente, i secondi servono ai clienti per capire se il progetto fa veramente quanto richiesto.
- **ascoltare**: i programmatori molto spesso non conoscono nulla riguardo al business che stanno costruendo, ne hanno una chiara conoscenza del dominio del discorso su cui si costruisce il progetto. Per questo è necessario che essi sappiano porre le domande adeguate ai clienti e siano in grado di ascoltare ciò che gli viene detto.

- **progettazione**: ciò permette di dare una certa logica all'architettura dell'applicazione. Un buon design permette di espandere l'applicazione con estrema facilità.

best practeces Possiamo quindi definire delle pratiche che hanno dimostrato di apportare molto valore:

- adottare cicli di rilascio molto brevi (al massimo qualche mese). In questo modo le cose da fare per ogni consegna sono poche, chiare e facilmente raggiungibili. Ciò permette di far progredire costantemente lo sviluppo del progetto. Risulta inoltre comodo realizzare per prima i requisiti più importanti, così le consegne man mano che progrediscono divengono meno importanti e se dovessero essere rinviate, il cliente avrà un danno minore. Adottando questa strategia si ottiene anche il vantaggio di testare maggiormente le parti più importanti del programma in quanto vengono realizzate per prime.

- tipicamente i progetti vengono cancellati perché non rendono abbastanza al cliente che li ha commissionati. Per ridurre questo rischio, XP propone al cliente di individuare le cose più importanti e che hanno veramente una buona possibilità di successo: si partirà da queste, in modo da generare un prodotto finito e utilizzabile e/o commercializzabile il prima possibile.

- avere uno stretto rapporto con il cliente garantisce una chiara identificazione di cosa egli voglia veramente, evitando fraintendimenti.

- adottare stime da parte dei membri del team. Ciò permette di capire quanto tempo sia necessario per raggiungere l'obiettivo prefissato. Ognuno è responsabile delle stime che ha fatto per realizzare il suo lavoro. Attraverso dei feedback sul tempo realmente impiegato si favorisce l'incremento delle capacità di stima di ogni membro. Oltre ciò per migliorare l'esperienza lavorativa si favoriscono gli incontri tra i membri, in modo che i programmatori si sentano meno soli nello sviluppo cosa purtroppo molto comune.

Pair Programming Per la programmazione può esse utile scegliere di sviluppare il codice in coppia: due persone lavorano sullo stesso computer alla realizzazione dello stesso codice. Ciò potrebbe sembrare uno spreco di risorse, in realtà porta ai seguenti vantaggi:

1. Si ottiene un programmazione impersonale (*egoless programming*) per cui nessuna parte del codice è comprensibile da una sola persona. In questo modo si favorisce il lavoro di gruppo e la commistione di idee diverse che portano alla realizzazione di soluzioni migliori anche ai piccoli problemi che si incontrano quotidianamente in fase di codifica.

2. Si ottiene una revisione formale del codice appena viene prodotto: ciò minimizza i costi per individuare i bug in fase di verifica.

3. Si riducono le false partenze in quanto la coppia discute prima di iniziare a sviluppare.

Il *pair programming* permette quindi di gestire meglio il rischio insito nel turnover del team: anche se qualcuno non partecipa più al progetto, la parte di lavoro da lui svolta è comunque conosciuta anche dagli altri colleghi.

Continous Integration Il codice viene integrato con le nuove versioni ogni giorno, anche più volte al giorno. Un modo semplice per fare ciò è lasciare una macchina adibita esclusivamente a questo compito. La coppia di programmatori che devono effettuare l'integrazione possono lavorare su di essa integrando il codice e risolvendo i conflitti affinché vengano completamente passati i test. In questo modo se c'è qualcosa che non funziona correttamente sarà compito di quella coppia risolvere i problemi in quanto è stata proprio lei ad introdurrli poiché l'integrazione precedente aveva lasciato il sistema perfettamente verificato (100I vantaggi che derivanod dall'utilizzo di questa tenica sono:

- assicurare che in ogni momento il prodotto sia in uno stato consistente e perfettamente utilizzabile limitatamente alle funzionalità realizzate;
- cospiquo incremento dell'attività di verifica;
- facile indivudazione degli errori;
- prodotto che viene facilmente reso operativo.

Adottare delle soluzioni hardware e software permette di facilitare l'adozione di questa pratica. È comodo avere un appostio server dedicato alla CI. È possibile utilizzare software apposito come *crouise control* (open source) che facilita il processo di build. Altro suggerimento è utilizzare un repository per poter sincronizzare velocemente tutte le copie. Tipicamente in esso devono essere inserite tutte le cose che permettono di effettuare una compilazione del progetto. Solamente ciò che è pesante e lungo da installare non deve appartenere ad esso, come il sistema operativo, il DBMS o Java.

Fonti: 9, 10, 17, 19.

4.2 TSP

TSP ovvero *Team Software Process* è una metodologia per gestire dei team di sviluppo con un numero non troppo elevato di componenti: da 2 a 20 persone sono tipicamente i limiti che vengono consigliati nella letteratura specialistica.
Lo scopo di questo processo è quello di organizzare al meglio il lavoro all'interno del team, distribuendo i ruoli in base all'esperienza che i singoli hanno maturato nel corso della loro carriera. Ciò avviene solo dopo aver individuato i goal che il progetto si prefigge di raggiungere. Tra i ruoli che partecipano al TSP, oltre ai classici presenti in ogni progetto IT che si rispetti (programmatori, progettisti, analisti, verificatori, ecc...) possiamo individuare anche i seguenti:

- team leader: si occupa di gestire l'andamento del progetto e del prodotto. Risponde direttamente ai vertici dell'azienda.
- coach: gestisce principalmente il team e i singoli individui che lo compongono. Risponde solitamente al team leader.

Una delle caratteristiche che contraddistinguono il TSP è il suo approccio ingegneristico, per cui ogni passo del ciclo di vita del software viene descritto rigorosamente. Il TSP è quindi ben strutturato e richiede la produzione di molta documentazione. Essendo così preciso, è necessario avere nel team delle persone che abbiano esperienza e formazione del campo TSP, viceversa prima di iniziare lo sviluppo sarà necessario un periodo di training. Quest'ultimo è molto importante e deve essere ben strutturato. Non è possibile iniziare ad imparare il TSP dal nulla. Bisogna prima far pratica con il PSP (*Personal Software Process*), solo in questo modo si possono assolvere le propedeuticità necessarie.

4.2.1 PSP

Il PSP è un processo che aiuta gli ingegneri del software a migliorare le loro prestazioni. In particolare si vogliono applicare le regole del CMMI al singolo individuo. Vengono proposte delle linee guida ognuna delle quali richiede degli input, suggerisce dei compiti da svolgere e vuole degli output per essere completata con successo.

Esistono diversi livelli di PSP. Essi sono elencati di seguito:

- PSP0: qui ci sono 3 fasi: pianificazione, sviluppo e post mortem. In particolare lo sviluppo si suddivide in progettazione, codifica, test. La fase di post mortem è la fine del primo livello. In essa si vogliono raccogliere ed analizzare tutti i dati raccolti sul progetto.
- PSP0.1: in questo livello si aggiungono al precedente dei standard per la codifica, inoltre si abilita il PIP (*Process Improvement Plan*). In esso il singolo ingegnere registra delle idee che gli permettono di migliorare il suo processo produttivo, in particolare per il settore della programmazione.
- PSP1: qui si impara a capire quanto tempo possa essere necessario per realizzare un progetto.
- PSP1.1: si scende a grana più fine e si cerca di stimare il tempo necessario per ogni singolo task da svolgere.
- PSP2: si introduce lo studio sulla qualità della progettazione e del codice realizzato. Si cerca di valutare quanti bug sono stati inseriti in fase di programmazione e quanto tempo sia necessario per correggerli.
- PSP2.1: vengono utilizzate delle tecniche di analisi per verificare i risultati ottenuti. Dopo questo punto si può arrivare ad applicare il TSP.

Gli scopi principali di questo processo sono riferiti al singolo ingegnere del software e sono:

- ridurre il numero di bug presenti nel suo lavoro;
- gestire la qualità dei progetti a cui egli partecipa;
- prendere degli impegni che sia veramente in grado di assolvere.
- incrementare la sua capacità di stimare i tempi necessari per produrre qualcosa.

4.3 Confronto Agile-TSP

Risulta difficile fare un confronto tra questi due modi di fare software in quanto, sebbene molto diversi, condividono comunque molti concetti tra i quali:

- avere poca burocrazia all'interno dell'azienda;
- la qualità è importante: meglio consegnare poco, ma fatto bene;
- darsi dei goal realistici: non si promettono mari e monti ai clienti;
- comunicazione frequente tra gli stakeholders, meglio se di persona;

Esistono però molte differenze.
Un vantaggio del TSP sull'agile è la sua rigidità: i ruoli all'interno del progetto rimangono fissi, anche all'inizio di un nuovo progetto. In questo modo si forma un team variegato, ma costante con al suo interno degli esperti ben formati. Ciò risulta difficile da raggiungere nel modello agile in quanto in esso i ruoli non sono ben definiti e soprattutto sono soggetti a grandi modifiche nel corso del tempo. L'Agile non mira ad avere dei singoli esperti per ogni settore, ma preferisce avere delle persone in grado di adattarsi a tutte le necessità che si possono manifestare durante lo sviluppo del progetto: se ci si trova indietro nella fase di codifica è comodo che gran parte del team si occupi della programmazione.
Altro aspetto di divergenza è il rigore. TSP è inoltre più preciso dell'Agile in quanto vengono conteggiate con precisione le ore spese, i bug incontrati, quanto si guadagna ecc... offrendo così un approccio ingegneristico quantificabile in ogni fronte del progetto. Al contrario l'Agile tende ad essere più sommario in questi settori, riducendo così al minimo il tempo non investito nella realizzazione, ma ottenendo maggiori difficoltà in particolare nella stima di quanto tempo possa essere necessario per completare un task. Le previsioni e le stime sono invece cruciali nel TSP dove ogni ingegnere deve saper organizzarsi al meglio: ciò è quanto viene richiesto dal PSP. Le consegne con TSP risultano quindi maggiormente rispettate.
Nel campo dei requisiti l'agile ha invece una marcia che pochi altri cicli di vita possono vantare: l'intensissima interazione con il cliente. Per realizzare un progetto si interloquisce con quest'ultimo, quindi ad intervalli regolari il team mostra al cliente i risultati ottenuti e aggiusta il tiro in base alle sue dritte. Il risultato è un continuo dibattito tra fornitore e acquirente che produce un ottimo risultato finale. Ciò ha però lo svantaggio che non tutti i clienti sono disponibili ad affrontare l'onere di un così stretto rapporto con l'azienda.
Il rigore del TSP porta con sé un grosso vantaggio: l'utilizzo di una terminologia precisa. Ciò permette di passare tra diversi team che utilizzano tale processo senza la necessità di dover apprendere nuove cose o altro, riducendo al minimo l'overhead causato dal turnover. Viceversa l'agile non è così ben definito, ciò rende abbastanza difficoltoso cambiare i membri del team.

Un buon metro di decisione nel scegliere quale dei due utilizzare è considerare quanto tempo sia disponibile e gli obiettivi che si vogliono raggiungere. Nel caso il progetto sia da completare il prima possibile, uno sviluppo Agile è indubbiamente la scelta più efficace. Viceversa è possibile utilizzare il TSP la cui sovrastruttura dovuta ai documenti da una parte rallenta lo sviluppo, dall'altra garantisce una maggior qualità dei processi applicati nello sviluppo.
Nella tabella 5 raccogliamo un veloce confronto tra questi due modi di sviluppare del software.

	Agile	**TSP**
dimensione progetto	piccoli e medi	piccoli, medi, grandi
documentazione	ridotta al minimo	rigorosa ed ampia
flessibilità	è facile adattarsi a nuovi scenari	molto rigido
interazione con il cliente	molto forte	media
ruoli	variabili	fissi
qualità del progetto	buona	molto elevata
tempi di produzione	bassi	abbastanza elevati
burocrazia	quasi nulla	bassa

Tabella 5: confronto Agile-TSP

fonti 15, 16.

5 Interdipendenza tra rischi

Nel momento in cui si determina il rischio è necessario tenere in considerazione che molti fattori di rischio sono dipendenti gli uni dagli altri. È quindi errato considerare ogni rischio singolarmente in quanto è probabile che si manifesti un effetto valanga generando situazioni difficili da gestire. Una delle migliori soluzioni per capire le relazioni fra diversi fattori è l'utilizzo di appositi strumenti come ISM e AHP.

ISM: Interpretive Structural Modeling ISM fu inventato nel 1973 da J. Warfiled con lo scopo di studiare complessi sistemi socio-economici. Nonostante ciò esso può essere applicato ad ogni settore, anche a quello del risk management. Il metodo principale con cui lavora questo sistema è quello di suddividere un sistema complesso in sotto-elementi individuati da esperti del settore. Ogni elemento verrà quindi messo in relazione con gli altri creando così dei grafi. In questo modo è possibile visualizzare in una mappa le relazioni dirette ed indirette di ogni componente. È quindi possibile capire da cosa è determinato ognuno di essi e cosa esso stesso possa a sua volta determinare.
ISM è un sistema computer-assisted e quindi permette un alto livello di automazione. Possiamo suddividere la realizzazione di questo modello nelle seguenti fasi:

1. Creazione di un team apposito composto da esperti del settore coordinati da qualcuno che abbia anche dei poteri decisionali;
2. Identificare i rischi;
3. Utilizzare matrici di adiacenza grazie alle quali è possibile stabilire quali siano i fattori che concorrono alla manifestazione di un rischio;
4. Utilizzare matrici reachibility che permettono di capire se da un fattore posso arrivare direttamente o indirettamente ad un altro fattore;
5. Decomporre il rischio in diversi livelli per realizzare un modello strutturato. Ciò permette di raggruppare i rischi in base alle relazioni che intercorrono tra di loro. Ad esempio è comodo raggruppare quei rischi che sono una diretta conseguenza di un rischio padre.

AHP Analytical Hierarchy Process Questo processo ha come scopo principale quello di quantificare ogni singola relazione presente all'interno della mappa creata con ISM. In questo modo risulta semplice capire quali siano i fattori che più pesantemente possono determinare un rischio.
fonti: 13,1

6 Permformance Management tool

In questo strumento vengono raccolte un insieme di attività che assicurano il raggiungimento degli obiettivi prefissanti in modo efficiente ed efficace. Ci si può focalizzare sul processo, sul progetto, sul team o addirittura sulla singola persona. Può essere applicato in qualunque contesto dove un insieme di persone cooperano tra di loro. Armstrong and Baron hanno definito il PM (Performance Management) in questi termini: *''un approccio strategico ed integrato per incrementare l'efficacia dell'organizzazione migliorando le performance delle persone che lavorano in essa e sviluppando le capacità dei team dei singoli individui.''*

Delle best practices per questo strumento sono:

- Cercar di far coincidere gli obiettivi dell'azienda con quelli del singolo individuo. In questo modo si stimola l'impegno quotidiano, incrementando la qualità e la quantità di lavoro svolto;
- In alternativa al punto precedente cercar di fare collidere gli interessi del singolo con quelli dell'azienda. Ad esempio si può incentivare lo sviluppo di un progetto inserendovi delle persone che hanno particolare interesse ad approfondire le tecnologie che in esso vengono utilizzate;
- Fare un'analisi degli scopi del progetto. È utile dare una mission statement ovvero una definizione di cosa si voglia ottenere, a chi si deve offrire il prodotto e quali sia il target dei clienti. Lo scopo dell'analisi è capire quali siano gli obiettivi chiave da raggiungere.

I risultati ottenibili con il Permformance Management sono vari e per questo vengono divisi per categorie:

- Ambito finanziario
  - Incremento delle vendite;
  - Diminuzione dei costi dell'azienda;
  - Diminuire gli sforamenti nei tempi di consegna;
  - Collegare gli obiettivi con i budget disponibili: in questo modo è possibile stanziare le giuste risorse finanziare per ogni progetto.
- Team di sviluppo
  - Si stimano le persone in quanto esse capiscono la loro importanza all'interno dell'azienda;
  - Rendere chiari gli scopi da raggiungere favorisce l'impegno costante delle persone;
  - Ottenere feedback dal team;
  - Si sottolinea l'importanza degli obiettivi da raggiungere.
- Sistema di controllo
  - Miglioramento del sistema di comunicazione sia interno che esterno;
  - Si semplifica la comunicazione degli scopi anche perché essi risultano più chiari;
  - Si risponde con maggiore efficacia alle necessità manageriali.

Altro beneficio essenziale ottenuto con questo sistema è allineare l'organizzazione agli scopi che ha il CEO.

6.1 BSC

Le BSC (Balanced ScoreCard) sono una particolare implementazione delle tecniche di Performance Management. Esse permettono di tener sotto controllo l'esecuzione delle attività da parte del team e le conseguenze derivanti da esse. Ciò è possibile mediante delle rilevazioni sia finanziare che non sullo stato del progetto. Gli ambiti che si possono prendere in considerazione sono i seguenti:

1. Prospettiva finanziaria: è necessario chiedersi in che modo l'azienda e il progetto in questione si presentino agli azionisti e se è necessario apportare dei miglioramenti nel caso non vengano incassati molti finanziamenti. Questa è ovviamente l'aspetto elementare di ogni progetto, nonostante ciò non deve essere l'unico ad essere considerato, altre misure sono infatti necessarie per capire come sta procedendo il lavoro e che aspettative può avere per il futuro. Proprio questo è il motivo per qui le BSC vengono definite bilanciate: non si prende in considerazione solo l'aspetto economico;

2. Prospettiva del consumatore: chiedendosi come i prodotti e l'azienda dovrebbero presentarsi ai clienti, si riesce a capire se è necessario modificare la pubblicità attualmente in uso o apportare delle modifiche ai prodotti offerti;

3. Prospettiva interna dell'impresa: la domanda chiave in questa fase è com'è necessario cambiare la struttura interna per adempiere ai primi due punti;

4. Prospettiva di innovazione e apprendimento: qui ci si chiede quali siano gli ambiti in qui è necessario migliorare per adempiere ai 3 punti sopra elencati.

In [13] si può trovare una più ricca dissertazione su questi ambiti.
Una volta individuati dei punti di miglioramento si comunicano al caporeparto del relativo settore. È bene precisare che questi ambiti non sono vincolanti, ma vengono solo suggeriti dagli autori di questo strumento. L'azienda può infatti decidere quali prendere in considerazione in base al progetto in analisi. In particolare è bene scegliere degli ambiti di misurazione che riguardino gli obiettivi prefissati. Essi vengono scritti su una *strategy map*. In essa saranno quindi presenti gli obiettivi e le relazioni di causa-effetto tra di loro. A queste mappe si possono fare dei miglioramenti:

1. Aggiungere agli obiettivi anche le opportunità che possono crearsi in itinere;

2. Creare una destination statement che raccoglie l'impatto degli obiettivi nel futuro. In questo modo si cerca di prevedere lo scenario aziendale dopo la realizzazione di un obiettivo.

Un buon modo per implementare una BSC indipendentemente dall'utilizzo di una strategy map o meno è il seguente:

1. Trasformare la visione aziendale in obiettivi concreti;

2. Comunicare la visione e collegarla agli obiettivi personali che hanno i membri dell'azienda;

3. Effettuare un businness plan;

4. Effettuare dei controlli e correggere la strategia adottata.

6.1.1 Best practices nelle BSC

In questa sezione elenchiamo alcune best practices che riteniamo utili nel momento in cui l'azienda decida di utilizzare le BSC.

- Per creare le BSC e le *strategic maps* è necessario definire gli obiettivi da raggiungere. È comodo che essi vengano identificati univocamente in modo che ci si possa riferire ad essi con estrema velocità e precisione;
- Gli obiettivi devono essere chiari e precisi. È comodo dare un titolo veloce costituito da un verbo, un soggetto e degli aggettivi chiari come ''Costruire una profonda partnership con i clienti''. Oltre al titolo è utile dare una descrizione precisa di cosa si voglia ottenere;
- Dare delle date di inizio ed di fine per gli obiettivi;
- È meglio concentrarsi su pochi obiettivi alla volta in modo da realizzarli al meglio. Avere troppe iniziative strategiche e obiettivi può portare a molta confusione;
- È necessario portare gli obiettivi e le strategie aziendali anche nei gradi più bassi dell'organizzazione, in modo che tutti i suoi dipendenti siano coscienti di cosa si voglia ottenere quanto meno nel breve termine. Molto comodo è intercalare un obiettivo nei settori che lo interessano maggiormente così che essi possano fare la loro parte per realizzarlo. Ad esempio un obiettivo aziendale potrebbe essere ''migliorare la qualità del sofware prodotto'', tale obiettivo nel reparto dei programmatori potrebbe essere tradotto come sviluppare una migliore campagna di test;
- Le misure che vengono raccolte devono riguardare principalmente gli obiettivi strategici che l'organizzazione si è prefissata. Ad esempio se un obiettivo è quello di diminuire il turnover all'interno di un team e quindi i licenziamenti delle persone, una buona misura è quella di svolgere dei sondaggi in cui si analizza il grado di soddisfazione dei membri del team;
- Valutare con molta attenzione quali siano gli obiettivi e le strategie che devono essere raggiunte. Spesso le organizzazioni si concentrano esclusivamente su obiettivi come incrementare le vendite e quindi i profitti. Ciò sebbene positivo nel breve termine può risultare devastante nel futuro: i dipendenti tendono a guardare al prossimo futuro dimenticando quale sia il vero scopo che governa l'azienda.

  fonti: 14.

7 SAFE

Il SAFE (*Safe Activities For Enhancement*) è un metodologia per la gestione del rischio che nasce nel dominio IT, ma può essere applicato anche in altri settori. Essa offre dei concetti e delle pratiche per il RM. Lo scopo è arrivare ad una conoscenza completa delle significative cause di rischio per un progetto. Ciò si ottiene esaminando apposite checklist e svolgendo delle discussioni di gruppo. Il punto d'inzio è individuare i rischi, quindi si cercano delle tecniche per ridurne la probabilità o il danno che il rischio porta con sé.
SAFE vuole precisare inoltre i seguenti concetti:

- Ogni ambiente di lavoro e progetto portano con sé dei rischi che non sono eliminabili;
- Meno le azioni che fa un'azienda sono ripetitive, più ci sono rischi;
- Gestire casi eccezionali occupa più risorse che andare avanti con la solita routine;
- Non bisogna sottostimare il tempo necessario per capire il rischio che si deve affrontare;
- La valutazione dei rischi va fatta continuamente, non solo all'inizio del progetto.

Viene suggerita questa guida ideale alle spese da intraprendere per il RM:piccoli investimenti per progetti piccoli, grandi per progetti importanti.

La struttura su cui si basa SAFE è visibile in figura 3:

1. Si identifica il rischio e lo si inserisce nella base di dati;
2. Si quantifica il rischio. Essa porta alla realizzazione di un documento (*risk assessment report*) che mostra i rischi a cui un progetto è esposto e la loro natura;
3. Si definiscono quindi degli interventi che mirano a limitare la probabilità che un rischio avvenga e/o il suo impatto sul progetto nel caso esso si manifesti. Ciò si conclude con la definizione del *risk management plan*. Esso contiene varie informazioni su come iniziare il progetto, e delle azioni da intraprendere per prevenire, monitorare e gestire i rischi;
4. Si mette in pratica ciò che è stato stabilito nel *risk management plan*;
5. Alla fine si verifica l'efficacia dell'intervento. In questo modo può esser confermato oppure messo in discussione il *risk management plan*. Lo scopo ultimo di questa fase è comunque quello di migliorare tale documento. Le valutazioni fatte vengono inserite nel *Risk management evaluation report*.

Di seguito esponiamo meglio alcune tappe di questa strategia.

Identificazione
Questa fase ha lo scopo di far comprendere quali rischi possano entrare in gioco durante lo sviluppo del progetto a tutte le persone che vi partecipano. Essa dovrebbe essere realizzata durante lo studio di fattibilità ed ogni qual volta cambiano i fattori di rischio del progetto. Possiamo utilizzare due modi per individuare i rischi:

- *forward chaining*: si pensa alle possibili situazioni nefaste che potrebbero verificarsi cercando quindi le cause che le generano;
- *backward chaining*: si pensa alle conseguenze indesiderate e si cercano le situazioni che le generano.

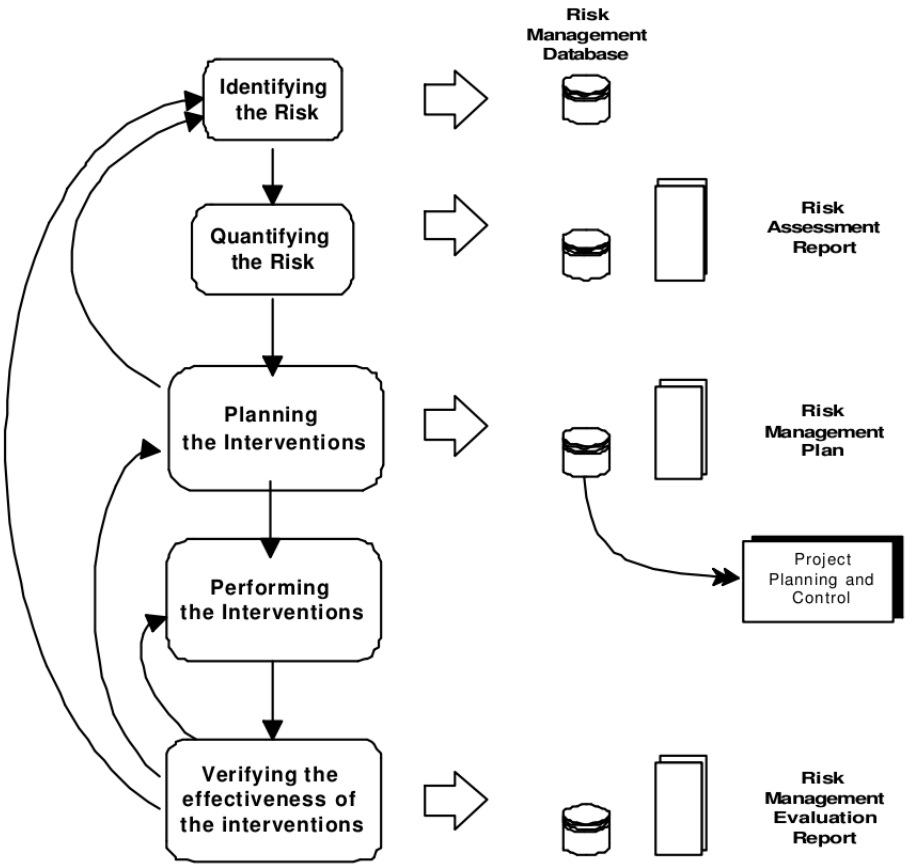

Figura 3: Struttura dell'implementazione della metodologia *SAFE*.

In entrambi i casi si possono utilizzare diversi modi per individuare i rischi. I principali sono:

- Mappa cognitiva del territorio;
- Diagramma di contesto.

Mappa cognitiva del territorio È una mappa che mostra i diversi stakeholders del progetto e le relazioni che intercorrono tra di loro. Le linee che li connettono possono indicare la criticità della relazione e i rischi che essa può portare.

Diagramma di contesto Esso ha lo scopo di mostrare gli elementi di criticità che possono affliggere il progetto in particolare quando esso si relaziona con il contesto nel quale dovrebbe operare. Un esempio è visibile in Figura 4. Si disegna un cerchio che viene diviso in settori concentrici e fette. Le aree definite dalle intersezioni tra queste due divisioni permettono di inserire gli elementi critici per il progetto. Essi sono di due tipi: fattori e soggetti. I primi sono degli eventi, circostanze o altro che non hanno capacità decisionali, i soggetti invece possono intraprendere o meno delle azioni che possono causare dei danni al progetto. Le fette rappresentano delle classi (società, politica, geografia ...). I cerchi concentrici raffigurano invece il modo (la *capacità*) con cui un progetto si relaziona ai fattori o ai soggetti. Si definisco tre diversi livelli:

1. **Control**: in questo caso l'elemento può essere pienamente gestito dall'azienda, riducendo il suo danno.

2. **Consideration**: questi elementi non sono sotto il controllo dell'azienda, nonostante ciò essi possono arrecare dei danni ad un progetto. È comunque possibile stabilire dei piani per gestire il problema nel caso si manifesti.

3. **Influence**: non è detto che si possa influenzare l'elemento.

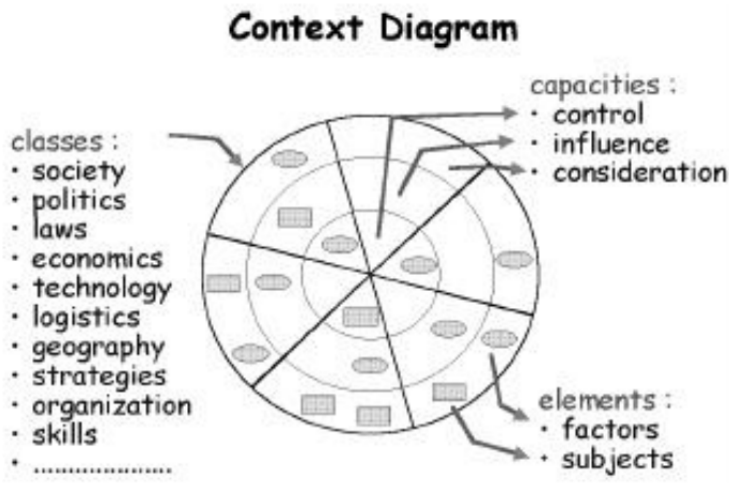

Figura 4: Esempio di *Diagramma di contesto*.

Quantificazione del rischio
Questa attività ha la caratteristica di essere molto legata al periodo in cui viene svolta: la probabilità e il danno legati ad un rischio variano considerevolmente nel corso del progetto. Dopo aver svolto una buona analisi quantitativa, è comodo utilizzare le matrici di rischio visibili in Figura 5. Esse vengono così interpretate nella metodologia safe. Viene preso il diagramma di contesto e lo si trasforma in una matrice. Gli elementi sono rappresentati come righe, le capacità (control, consideration, infulence) sono invece rese attraverso le colonne. Si da un valore per un'intersezione tra una riga e una colonna, tenendo presente che ogni riga può avere al massimo un valore in quanto un elemento può appartenere solo ad una capacità. Tale valore sarà la probabilità del rischio moltiplicata per l'impatto causato al progetto (nella figura tali valori sono espressi genericamente con delle lettere). Sia la probabilità che l'impatto vengono valutati su una scala che va da 0 a 10 (0 = minor probabilità o impatto). A questo punto si calcola il valore totale delle capacità sommando i valori presenti nelle loro rispettive colonne. Si ottengono quindi i valori X,Y,Z della figura. Essi permettono di calcolare il coefficiente *Self Determination Index* (SDI):

$$SDI=100*(X+Y/2)/(X+Y+Z)$$

Y/2 viene aggiunto in quanto rappresenta la capacità *Consideration* può essere o meno influenzata dal progetto e quindi viene considerato che solo una metà degli elementi possa essere gestita dall'azienda. Quando il SDI tende a zero significa che il progetto può gestire pochi rischi e quindi avrà un'importante probabilità di insuccesso. Viceversa un valore prossimo a 100 è indice di un progetto che ha ampie possibilità di concludersi per il meglio non perché su di esso incombano pochi rischi, bensì perché esso può gestire al meglio la maggior parte di essi.

Pianificazione interventi
Grazie a questa attività è possibile avere una lista di possibili interventi da fare per relazionarsi con i rischi. *SAFE* suggerisce di attenersi all'*Euromethod: Strategy*

Risk Matrix

Critical Element	Control	Influence	Consideration
E1	a		
E2	b		
E3		c	
E4			d
E5	e		
E6		f	
E7			g
E8			h
E9			i
E10		l	
E11		m	
E12			n
E13	o		
Totals	X	Y	Z

Figura 5: Esempio di Matrice di Rischio come viene utilizzata nella metodologia *SAFE*.

Model. Esso fornisce delle indicazioni su come affrontare un progetto che abbia dei rischi al suo interno. I rischi secondo questo modello vengono classificati in base alla complessità e all'incertezza. Ai primi appartengono rischi legati al management e alla dimensione del progetto, mentre ai secondi appartengono l'incertezza riguardo ai requisiti e l'innovazione tecnologica.
Si possono così individuare delle azioni da svolgere:

- *Prevenzione*: vuole minimizzare le probabilità che un rischio si manifesti;
- *Monitoraggio*: cerca di individuare quando un rischio si sta manifestando. Servono dei sensori che aiutino questa fase. Essi sono tipicamente socio-tecnici nel senso che utilizzano sia persone che strumenti;
- *Combattimento*: è la miglior soluzione in quanto si prefigge di cancellare gli effetti negativi dovuti alla concretizzazione di un rischio.

Fonti: 18.

8 Bibliografia

[1] *D. Aloini et al*, Risk Management in Enterprise Resource Planning

[2] *Don Shafer*, Software Risk: Why must we keep learning from experience?, 2004 reperibile all'indirizzo http://www.compaid.com/caiInternet/ezine/Software_Risk_Shafer.pdf

[3] *Linda Westfall*, SOFTWARE RISK MANAGEMENT, reperibile all'indirizzo http://www.westfallteam.com/Papers/risk_management_paper.pdf

[4] *Tom Kendrick*, Identifying and Managing Project Risk Essential Tools for Failure-Proofing Your Project

[5] *C. Ravindranath Pandian, Applied Software Risk Management,* A Guide for Software Project Managers, *Auerbach publications.*

[6] ISO/IEC Guide 73:2002 (2002). Risk Management. Vocabulary. Guidelines for use in standard ISBN: 0 580 40178 2, 128.

[7] *Dale F. Cooper, Stephen Grey, Geoffrey Raymond e Phil Walker,* Project Risk Management Guidelines, Managing Risk in Large Projects and Complex Procurements, *John Wiley & Sons Ltd*, 2005

[8] *Daniel D. Galorath e Michael W. Evans Software Sizing*, Estimation, and Risk Management, *Taylor & Francis Group, L*, 2006

[9] Sito Web del manifesto agile: http://agilemanifesto.org/iso/it/

[10] *Ian Sommerville*, Ingegneria del software 8ed, *Pearson Education.*

[11] *Stephen Sims*, http://www.sans.edu/research/leadership-laboratory/article/risk-assessment

[12] *Daniel D. Galorath e Michael W. Evans* Software Sizing, Estimation, and Risk Management, Taylor & Francis Group, L, 2006

[12] *Gorvett Rick, Liu Ningwei*, Using Interpretive Structural Modeling to Identify and Quantify Interactive Risks, reperibile all'indirizzo: http://www.actuaries.org/ASTIN/Colloquia/Orlando/Papers/Gorvett.pdf

[13] *Robert S. Kaplan, David P. Norton*, Balanced scorecard. Tradurre la strategia in azione, ISEDI

[14] *Bill Barberg*, Balanced Scorecard Best Practices: Understanding Leading Measures, reperibile all'indirizzo
http://businessintelligence.com/article/133.

[15] *Capers Jones*, Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies, McGraw-Hill

[16] http://www.agileteams.com/papers/Publications/TUG2003/ComparingAgileAndTSP-v1_61.pdf

[17] *Kent Beck*, Extreme Programming Explained: Embrace Change.

[18] *Meli, R* (1998). SAFE: A Method to Understand, Reduce, andAccept Project Risk. ESCOM-ENCRESS 98, Project Control for 2000 and Beyond; Rome, Italy, May 27,29.

[19] http://www.martinfowler.com/articles/continuousIntegration.html

[20] http://www.agileweboperations.com/scrum-vs-kanban

[21] http://www.scrumalliance.org/learn_about_scrum

[22] *Henrik Kniberg, Mattias Skarin*, Kanban and Scrum - making the most of both.

8.1 Altre fonti

Di seguito si propongono altre fonti che sono state rintracciate dal team nel corso della realizzazione del presente documento, ma che per motivi economici o temporali non sono state utilizzate per la sua produzione.

http://www.icmrindia.org/casestudies/Case_Studies.asp?cat=Enterprise%20Risk%20Management: sono presenti dei casi di studio sul risk management. Essi sono a pagamento e spaziano tra vari settori, alcuni riguardano anche quello IT con particolare rifermento a note azienda quali Microsoft, Cisco e Dell.