POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Gestionale

Tesi di Laurea Magistrale

L'approccio Agile nello sviluppo di prodotti hardware e i vincoli di fisicità

Relatore
Prof. Marco Cantamessa

Candidato
Sara Tosco

Correlatore
Prof. Francesca Montagna

Anno Accademico 2020/2021

Ai miei genitori e a mio fratello

Sommario

Indice delle figure

Indice delle tabelle

INTRODUZIONE

Il contesto in cui si muovono le aziende è in continua evoluzione e la disruption arriva da più fronti. Le organizzazioni devono affrontare i progetti di sviluppo in condizioni sempre più incerte, volatili, complesse e ambigue (VUCA) cercando di mantenere la competitività e cogliere i cambiamenti del mercato.

Con il processo di sviluppo prodotto le aziende trasformano le opportunità di mercato in prodotti realizzabili e conformi alle aspettative dei clienti, tuttavia, per inglobare i continui cambiamenti devono essere celeri, flessibili e altamente proattive. Soprattutto in ambito di ricerca, l'incertezza condiziona fortemente i metodi di lavoro dei team di sviluppo che si trovano incapaci di prevedere i risultati e gli eventi futuri a causa della differenza tra la quantità di informazioni richieste e quelle effettivamente disponibili.

Alla luce di tutto ciò, i classici approcci di sviluppo plan-driven, concepiti secondo un criterio di linearità, risultano eccessivamente vincolati da un piano e poco reattivi, in quanto non consentono di accogliere rapidamente le modifiche in corso d'opera. Per contrastare questa mancanza di reattività dei metodi più tradizionali, molte aziende manifatturiere hanno iniziato a ripensare al modo in cui i processi di sviluppo sono gestiti e a comprendere il vero valore dall'Agile, un paradigma ormai mainstream nel mondo software.

Infatti, l'Agile è un approccio di gestione dei progetti che possiede tutte le peculiarità per soddisfare i requisiti di proattività e adattamento dei processi di sviluppo e, nonostante sia nato per essere applicato al contesto software, per le ragioni sopracitate sta poco per volta contaminando anche il contesto dei prodotti tangibili, con tutte le implicazioni che ne derivano. È un paradigma iterativo e incrementale, che enfatizza la condivisione e la generazione del valore, tuttavia porta con sé alcune complicazioni derivanti dal fatto che non tiene conto dei requisiti fisici dei prodotti, motivo per cui le pratiche non possono semplicemente essere trasferite dal mondo software a quello hardware, ma necessitano di una serie di adattamenti.

Il seguente lavoro di tesi nasce con l'obiettivo di analizzare il grado di agilità che le aziende riescono a implementare nei loro processi di sviluppo e quali adattamenti sono necessari ai diversi contesti. Nei capitoli seguenti, lo studio è strutturato partendo da una prima analisi della letteratura in modo da esaminare i contributi della ricerca passata e delle conoscenze esistenti sull'applicabilità dell'Agile al di là dei domini software. Questo primo punto di

partenza è una base solida su cui costruire la ricerca, esaminando cosa è stato già definito e cosa deve ancora essere indagato.

Successivamente, il lavoro procede con una ricerca qualitativa che analizza una raccolta di interviste semi-strutturate svolte con progettisti e responsabili di team di progetto, in modo da esaminare diverse esperienze di aziende presenti sul territorio italiano e che applicano le pratiche agili. Questi casi di studio consentono di identificare diverse configurazioni di sviluppo intermedie, in cui l'approccio Agile è integrato in modo più o meno spinto nelle organizzazioni, talvolta ibridato ai metodi più tradizionali. Si individuano di conseguenza i principali adattamenti che sono stati messi in piedi per conformare le pratiche del manifesto ad artefatti "non software".

In particolare, lo studio si focalizza sull'analisi dei processi di product development delle aziende in relazione al grado di digitalizzazione dei prodotti e alla loro architettura. Le ragioni di queste due scelte di analisi si fondano sulle caratteristiche fisiche dei prodotti, per comprendere quanto queste possano costituire un vero vincolo all'applicazione delle pratiche agili. Infatti, da un lato si vuole analizzare quanto la vicinanza al mondo software faciliti una conversione delle pratiche e un cambiamento all'approccio di sviluppo per diventare un'azienda agile a tutti gli effetti. Dall'altro, si cerca di comprendere se la modularità sia un requisito essenziale e come può essere applicato uno sviluppo incrementale a prodotti integrali, in cui l'architettura non consente di scomporre facilmente l'oggetto di sviluppo in parti.

1. IL PROCESSO DI SVILUPPO PRODOTTO

Il processo di sviluppo prodotto indica un insieme di attività volte a sviluppare prodotti, processi, sistemi, servizi, che saranno utili per tutto il ciclo di vita del prodotto. È un processo consolidato all'interno delle organizzazioni, che si occupa di tradurre i requisiti in soluzioni o servizi tecnici e commerciali e che spesso coinvolge più aree funzionali.

Ogni processo è unico, ma tutti i processi hanno elementi simili e ricorrenti tra loro e sulla base di questa comprensione, sono stati generati diversi modelli per supportare gli sviluppatori nella trasformazione dei requisiti in soluzioni (Wynn & Clarkson, 2018).
In generale, il processo di sviluppo prodotto può essere inteso come l'interazione continua tra tre sistemi: il sistema degli obiettivi, degli oggetti e il sistema operativo (Figura 1).

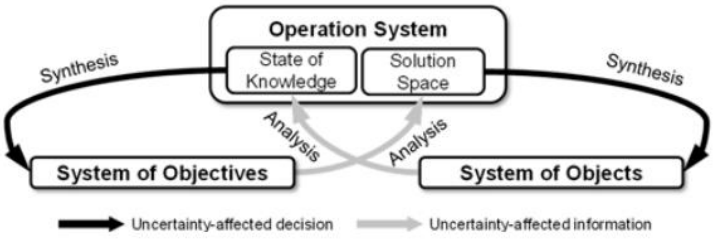

Figura 1: Il Sistema Triplo del processo di sviluppo prodotto

Quindi, è come se ci fosse una trasformazione continua di un **sistema di obiettivi** in un **sistema di oggetti**, passando attraverso un **sistema operativo** (Albers, Lohmeyer, & Ebel, 2011). Quest'ultimo è costituito da attività, metodi e processi, sviluppatori e tutte le altre risorse necessarie per lo sviluppo del prodotto. Il sistema di obiettivi include tutti gli obiettivi che devono essere raggiunti, la loro giustificazione e le interazioni, nonché le condizioni al contorno di una soluzione, ma non la soluzione stessa (Albers, Trost, Heimicke, & Spadinger, 2020). Sulla base del sistema di obiettivi è creato lo spazio delle soluzioni, ovvero uno schema mentale di tutte le soluzioni possibili che soddisfano i requisiti. A questo punto il team di sviluppo (come parte del sistema operativo) sintetizza vari oggetti (prototipi, prodotto finale) nel processo di sviluppo prodotto, che vengono combinati all'interno del sistema di oggetti. Un allineamento del sistema di oggetti con il sistema di obiettivi porta a un ampliamento della base di conoscenza e quindi alla capacità di concretizzare continuamente il sistema di obiettivi durante l'intero processo di sviluppo. Il sistema di obiettivi e il sistema di oggetti sono collegati tra loro esclusivamente tramite il sistema operativo.

1.1 Le fasi del processo di sviluppo prodotto tradizionale

In generale, il processo di sviluppo prodotto differisce da un'azienda all'altra in quanto dipende pesantemente dalle caratteristiche dell'industry e dei prodotti sviluppati. Tuttavia, è possibile definire alcune fasi tipiche che compongono il processo di sviluppo (Figura 2) tradizionale in modo da avere un riferimento generale (Cantamessa & Montagna, 2016):

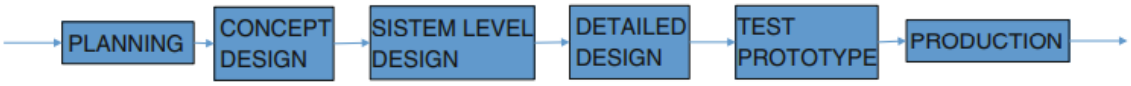

Figura 2: Fasi del processo di sviluppo prodotto

- La **pianificazione** è la fase in cui l'azienda si pone l'obiettivo di definire il nuovo prodotto dal punto di vista tecnologico (quali funzioni dovrebbe avere? Quali performance?) e di mercato (a quali clienti è rivolto? Quali bisogni deve soddisfare?). Si analizzano quindi i bisogni del mercato, traducendoli in requisiti di prodotto, e si definiscono i tempi di realizzazione del progetto, cioè il time-to-market. È una fase altamente interdisciplinare che coinvolge la maggior parte delle funzioni aziendali. Inoltre, prevede la realizzazione di un business case, confrontando l'investimento atteso con le previsioni sui ritorni delle vendite.

- Il **concept design** è la fase preliminare di ogni nuovo progetto ed è anche quella più creativa, in cui prendono forma tutti gli elementi di conoscenza del team di sviluppo. In questa fase si affronta la tecnologia nel dettaglio, esplorando tutte le soluzioni tecniche in grado di soddisfare le esigenze dell'utente precedentemente definite. Da questo punto in poi parte una fase di selezione: nascono tante alternative che sono valutate, selezionate o scartate man mano che il processo avanza.

- Il **system level design** è la vera e propria progettazione a livello di sistema, in cui l'azienda prende importanti decisioni tecniche. Si definisce un'architettura di prodotto e si effettuano scelte sui principali sottosistemi e componenti. Infine, si stabiliscono le politiche di make or buy, dunque le attività svolte internamente e quelle da esternalizzare.

- Nella fase di **detailed design** si concentra la maggior parte del lavoro ingegneristico: si scende nel dettaglio delle soluzioni tecniche, si scelgono materiali e componenti e si studiano le interfacce che li collegano. Grazie a una visione maggiormente dettagliata è

possibile stabilire in modo più preciso i tempi e i costi di realizzazione. Le decisioni prese in questa fase saranno eventualmente riviste.

- La fase di **prototipazione e test** è concettualmente posizionata alla fine del processo, anche se ormai avviene al termine di ciascuna fase di desgin. Comprende la verifica e la convalida delle soluzioni generate nelle fasi precedenti; i test possono essere analitici oppure eseguiti tramite simulazioni su prototipi fisici o virtuali. Spesso i test hanno anche fini normativi per validare il rispetto di determinate specifiche o per ottenere il rilascio di certificazioni da parte delle autorità competenti, senza i quali alcuni prodotti non potrebbero entrare in commercio.
- Il **process design** coinvolge la progettazione del processo di produzione, distribuzione e manutenzione del prodotto. Ciò comporta anche la progettazione delle risorse necessarie come strumenti, stampi e attrezzature per la produzione, manuali di assistenza e utensili per gli ingegneri sul campo e così via.
- Il **lancio e la produzione** chiudono il processo di sviluppo prodotto. L'azienda organizza e blocca le risorse dedicate alla produzione e alla distribuzione dei prodotti. In questa fase si gestisce la transizione verso la produzione a regime, selezionando i fornitori e definendo il piano di approvvigionamento, gli obiettivi e i volumi di produzione, le strategie di marketing e le prime azioni commerciali. Solitamente le aziende assegnano un arco di tempo subito dopo il lancio, durante il quale il prodotto e il processo sono messi a punto per garantire l'aderenza alle specifiche e l'accettazione del mercato, dopodiché il processo di sviluppo è ufficialmente concluso.

È importante sottolineare che una visione orientata al processo di sviluppo prodotto non è sempre valida. In condizioni stabili, in cui il problema da risolvere è ben strutturato, è possibile pensare che esista un processo predefinito per lo sviluppo prodotto e che le attività possano essere definite e organizzate ex ante. Tuttavia, nel caso di problemi incerti e mai esplorati, non è possibile definire ex ante un processo risolutivo, poiché questo verrà "scoperto" volta per volta durante lo sviluppo. Pertanto, le fasi appena descritte sono percorse alla lettera per progetti di sviluppo prodotto con un contenuto innovativo piuttosto moderato, quando le esperienze pregresse possono essere sfruttate come linee guida. Al contrario, questo potrebbe non essere più valido in caso di progetti molto innovativi, per i quali non è stata ancora identificata e formalizzata una chiara strategia di soluzione (Cantamessa & Montagna, 2016).

2. L' APPROCCIO AGILE

L'approccio Agile nacque nel febbraio del 2001, quando 17 sviluppatori software professionisti si riunirono a Snowbird, nello Utah, per cercare una valida alternativa alle tecniche di sviluppo tradizionali (Beck, et al., 2001). Tutti i partecipanti ritenevano che il modello Waterfall non rappresentasse la soluzione più efficace, soprattutto in un'ambiente in cui è importante rispondere ai continui cambiamenti e quindi possedere una certa "agilità" per affrontare uno sviluppo dinamico. L'obiettivo era quello di definire una nuova metodologia in grado di aumentare la qualità, migliorare la flessibilità e accelerare il time to market (Ciric, Lalic, Gracanin, Placic, & Zivlak, 2018).

Il risultato fu l'Agile, un framework di sviluppo che prevede di energizzare, responsabilizzare e consentire ai team di progetto di generare valore in modo rapido e affidabile, coinvolgendo i clienti, apprendendo e adattandosi continuamente alle mutevoli esigenze dell'ambiente circostante. Mediante questo approccio non si pianifica né si progetta un prodotto in anticipo, ma il processo di sviluppo evolve per cicli iterativi. Queste brevi sessioni, chiamate sprint, solitamente hanno una durata di 2 o 3 settimane, in cui ogni membro del team deve completare una serie di compiti assegnati (comunemente chiamati task). Al termine di ogni sprint si analizzano i risultati ottenuti per misurare lo stato di avanzamento dei lavori e la velocità con cui il team di sviluppo brucia le tappe. A questo punto il processo ricomincia da capo. I processi di gestione dei progetti che incorporano questi principi consentono al cliente di ricevere man mano aggiornamenti e segnalare priorità o introdurre cambiamenti quando necessario (Abellàn, 2020).

Dunque, l'Agile consente di rispondere al cambiamento in un business turbolento facendo ricorso a:

- una forte interazione con i clienti e la loro integrazione durante le fasi di sviluppo;
- una riprioritizzazione rapida delle risorse;
- una consegna evolutiva, incrementale e iterativa dei prodotti;
- una metodologia just in time, che generi valore massimizzando il lavoro non svolto.

Rispetto alle metodologie tradizionali che puntano alla massimizzazione del profitto (shareholders based), l'Agile ha l'obiettivo di massimizzare la soddisfazione del cliente, cioè il valore generato (stakeholders based). Inoltre, il processo passa da essere prescrittivo (rule-based) ad adattativo (principle-based).

Le idee alla base del movimento Agile sono state riportate all'interno dell'Agile Manifesto, un documento contenente 12 principi e 4 valori che esplicano le riflessioni dei fondatori (Beck, et al., 2001). Di seguito sono elencati i principi:

(1) La nostra massima priorità è soddisfare il cliente rilasciando software di valore, fin da subito e in maniera continua.
(2) Accogliamo i cambiamenti nei requisiti, anche a stadi avanzati dello sviluppo. I processi agili sfruttano il cambiamento a favore del vantaggio competitivo del cliente.
(3) Consegniamo frequentemente software funzionante, con cadenza variabile da un paio di settimane a un paio di mesi, preferendo i periodi brevi.
(4) Committenti e sviluppatori devono lavorare insieme quotidianamente per tutta la durata del progetto.
(5) Fondiamo i progetti su individui motivati. Diamo loro l'ambiente e il supporto di cui hanno bisogno e confidiamo nella loro capacità di portare il lavoro a termine.
(6) Una conversazione faccia a faccia è il modo più efficiente e più efficace per comunicare con il team e all'interno del team.
(7) Il software funzionante è il principale metro di misura di progresso.
(8) I processi agili promuovono uno sviluppo sostenibile. Gli sponsor, gli sviluppatori e gli utenti dovrebbero essere in grado di mantenere un ritmo costante.
(9) La continua attenzione all'eccellenza tecnica e alla buona progettazione esalta l'agilità.
(10) La semplicità - l'arte di massimizzare la quantità di lavoro non svolto - è essenziale.
(11) Le architetture, i requisiti e la progettazione migliori emergono da team che si auto-organizzano.
(12) A intervalli regolari il team riflette su come diventare più efficace, dopodiché regola e adatta il proprio comportamento di conseguenza.

I valori agili invece sono (Beck, et al., 2001) (Ingrosso, 2020):

(1) *Gli individui e le interazioni più che i processi e gli strumenti.*

Lo scopo di questo valore è proprio quello di sottolineare l'importanza di avere un bilanciamento tra gli individui e i processi. Al giorno d'oggi molti progetti sono caratterizzati da un'elevata incertezza, con una forte spinta all'innovazione e alla creatività. In contesti come questo, seguire un processo predefinito e standardizzato

non è mai la soluzione ottimale per portare a termine con successo un progetto. Di fronte all'incertezza e all'esplorazione di nuove soluzioni, i membri del team dovrebbero avere la libertà di sperimentare e di essere sempre aperti all'apprendimento; ognuno dovrebbe avere la possibilità di creare i suoi processi, personalizzandoli in base alle proprie esigenze e agli obiettivi da perseguire. Tuttavia, i team di sviluppo sono soliti ricevere processi e strumenti predefiniti da parte dell'azienda, rimanendo così bloccati dalle abitudini all'interno di una cultura predittiva e questo chiude l'apertura verso l'innovazione, la creatività e l'esplorazione di nuove soluzioni.

(2) *Il software funzionante più che la documentazione esaustiva.*

Con "software funzionante" si intende un software completo, testato, integrato, che risponda ai requisiti di qualità richiesti e che generi valore per il cliente.

La qualità è soggettiva e può cambiare nel corso degli anni, per questo motivo occorre sempre specificare le richieste in maniera chiara e misurabile, indicando qual è il requisito qualitativo da rispettare.

Per quanto riguarda la documentazione esaustiva invece, si intende tutto quel lavoro che non è consegnato al cliente perché privo di valore.

(3) *La collaborazione con il cliente più che la negoziazione dei contratti.*

Questo valore si basa sulla necessità di mantenere sempre una relazione di collaborazione con i principali stakeholder del progetto, fortemente fondata sul concetto di trasparenza e fiducia. In tal senso, è necessario formulare dei contratti adeguati alla situazione, senza puntare a una continua negoziazione, bensì a una collaborazione costante.

(4) *Rispondere al cambiamento più che seguire un piano.*

La metodologia Agile non prevede l'utilizzo di un Gantt, ma ciò non significa che la pianificazione sia completamente assente. Piuttosto, si procede passo per passo con una pianificazione di tipo adattativo, che accetta cambiamenti al progetto ed è estremamente flessibile. Questo tipo di pianificazione non è supportata da un percorso predefinito così da orientare il team ad abbracciare il cambiamento.

Inoltre, l'Agile istituisce una serie di pratiche di gestione basate su cicli iterativi e di sviluppo incrementale, in cui i requisiti e le soluzioni evolvono dando la priorità ai clienti, grazie al

lavoro di team collaborativi, interfunzionali e auto-organizzati (Beck, et al., 2001). Con il termine “pratica” si intende un tipo specifico di "azione gestionale" che contribuisce all'esecuzione di un processo e che può impiegare una o più tecniche e strumenti (Conforto, Salum, Amaral, Silva, & Almeida, 2014). In totale si possono identificare 6 pratiche di gestione che differenziano chiaramente l'approccio Agile dagli altri:

- L'uso del concetto di "product vision", che prevede l'impiego di strumenti visivi come lavagne, figure o disegni per fornire una semplice descrizione del progetto generale. Aiuta a identificare quali aspetti sono maggiormente apprezzati dai clienti o dal mercato e non a capire "come" sviluppare il prodotto. Utilizza metafore, figure e prototipi, differendo quindi dal modo tradizionale di presentare il design del prodotto (solitamente è sotto forma di WBS o di distinta base, con una descrizione prevalentemente testuale e dettagliata dell’attività da svolgere e dei parametri delle prestazioni del prodotto).

- L'uso di una comunicazione stretta, utilizzando semplici strumenti e processi di comunicazione del piano di progetto. Si fa riferimento alla comunicazione interpersonale, fra tutti gli stakeholder di progetto (cliente compreso) e serve ad avere una buona analisi dei requisiti e una proficua collaborazione tra gli sviluppatori.

- L’uso di una pianificazione iterativa: non è sviluppato un unico piano, ma l’intero progetto è spaccato su brevi iterazioni, all’interno delle quali c’è una pianificazione di dettaglio. Questa pratica consiste nella consegna rapida e continua di porzioni di prodotto, ottenendo costantemente un feedback da parte del cliente in modo tale da rispondere ai cambiamenti nel modo più rapido possibile.

- L'uso di team autogestiti e auto-diretti per lo sviluppo delle attività, che consente di garantire il pieno impegno e coinvolgimento di ogni membro del team.

- L’impiego di team autogestiti e auto-diretti nelle attività di monitoraggio e aggiornamento del piano di progetto, che aiuta a monitorare costantemente i progressi. Il coinvolgimento attivo del team, sia nella pianificazione che nel controllo del progetto, contribuisce a rendere più efficace l'interazione e la comunicazione tra

gli individui, nonché lo sviluppo di figure professionali in grado di imparare, operare e adattarsi di fronte ad ambienti complessi.

- L'aggiornamento frequente delle attività del piano di progetto, soprattutto alla fine di ogni iterazione, è essenziale, a differenza della teoria tradizionale che prevede che il progetto debba essere rivisto dopo un'importante milestone o in concomitanza della conclusione di una fase.

2.1 Scrum: un framework Agile

La filosofia Agile può essere implementata tramite diversi framework di gestione che riflettono i suoi principi. Tra questi, il più diffuso è Scrum, ovvero un metodo di gestione dei progetti che mira a rafforzare i valori di trasparenza, ispezione e adattamento durante tutto il processo di sviluppo.

Scrum è una metodologia di lavoro interattivo e di sviluppo incrementale del prodotto, che utilizza il time box come elemento di delivery e cerca, attraverso una strutturazione del flusso di lavoro, di caratterizzare i 12 principi agile. Il termine Scrum deriva dal mondo del rugby e tradotto dall'inglese vuol dire mischia. Infatti, i progettisti si muovono insieme passandosi velocemente la palla proprio come dei giocatori durante una partita. Il framework si basa sull'intelligenza collettiva di coloro che lo utilizzano: piuttosto che fornire alle persone delle istruzioni dettagliate, le regole di Scrum guidano le relazioni e le interazioni durante tutto il corso del progetto (Schwaber & Sutherland, 2020).

Scrum si fonda sull'empirismo e su un pensiero Lean (snello): l'empirismo afferma che la conoscenza proviene dall'esperienza e dal prendere decisioni basate su ciò che è osservato, mentre il pensiero Lean punta a ridurre gli sprechi, concentrandosi sull'essenziale. Scrum utilizza un approccio iterativo e incrementale per ottimizzare la predittività e controllare il rischio. Coinvolge gruppi di persone che nel complesso hanno tutte le capacità e le competenze per svolgere il lavoro e condividere o acquisire sul campo le skills necessarie.

Scrum non prevede delle riunioni, bensì delle cerimonie con particolari protocolli per essere gestite, e non produce dei documenti, bensì degli artefatti. In particolare, combina quattro eventi formali per l'ispezione e l'adattamento del lavoro, tutto racchiuso all'interno di un

unico sprint. Gli eventi sono funzionali perché consentono di implementare i pilastri empirici del framework Scrum, ovvero (Schwaber & Sutherland, 2020):

- Trasparenza: il processo deve essere visibile sia a coloro che svolgono il lavoro sia a coloro che ne beneficiano. Artefatti con un basso livello di trasparenza possono portare a decisioni che riducono il valore, aumentano i rischi. La trasparenza permette l'ispezione.
- Ispezione: gli artefatti e l'avanzamento verso gli obiettivi concordati devono essere controllati frequentemente e diligentemente, in modo da rilevare deviazioni o problemi indesiderati. Per agevolare l'ispezione, Scrum fa ricorso agli eventi all'interno di ogni sprint. L'ispezione a sua volta consente l'adattamento.
- Adattamento: il processo e le risorse impiegate devono essere adattati in caso di deviazioni al di fuori dei limiti accettabili. L'adattamento deve essere effettuato il più presto possibile per ridurre al minimo ulteriori deviazioni e diventa più difficile quando le persone coinvolte non possiedono l'autorità necessaria o quando non sono autogestite. Infatti, ci si aspetta che uno Scrum Team si adatti nel momento in cui apprende qualcosa di nuovo attraverso l'ispezione.

2.1.1 Il funzionamento del framework Scrum

Il seguente paragrafo spiega il funzionamento di Scrum, il cui flusso è riportato in Figura 3. Lo schema, infatti, rappresenta tutte le attività da seguire per applicare correttamente il framework di sviluppo.

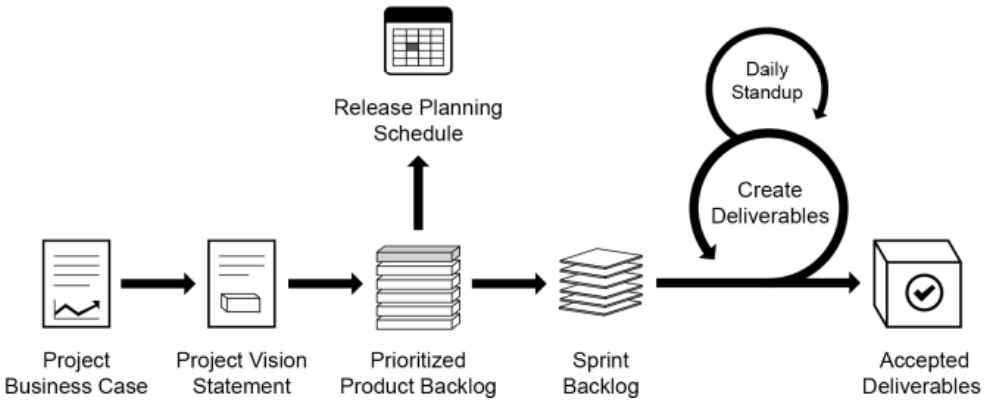

Figura 3: Il funzionamento di Scrum

Il **Project Business Case** è il vero e proprio modello di business che serve in azienda per valutare l'avvio del nuovo progetto. In questa fase è analizzato il business case per creare un documento ufficiale che dichiari quale sia la vision seguita durante tutto il ciclo di vita del progetto (Lavecchia, s.d.). Si produce il documento che si identifica con la fase di avvio per ottenere l'opportuna approvazione da parte degli enti preposti e riassume i ricavi e i costi attesi.

In fase di planning il Project Business Case si traduce nel **Project Vision Statement** (visione del prodotto) in cui si identificano gli stakeholders e i loro fabbisogni, in modo da definire il prodotto da realizzare nella sua totalità. I fabbisogni sono tradotti in requirements che devono essere rilasciati attraverso tasks.

Una volta approvato il progetto, si avvia il Planning generale attraverso la definizione di un **Prioritized Product Backlog**: tutti gli item (elementi del prodotto) sono identificati e classificati in un elenco di componenti prioritizzati da realizzare. Dunque, il backlog contiene tutti i requisiti tecnici e funzionali che il prodotto dovrà soddisfare, classificati in ordine di importanza. La classificazione dei requisiti avviene seguendo la regola *MoSCoW*:

- Must: requisiti essenziali, che devono essere contenuti nel progetto affinché questo sia di successo.
- Should: elementi importanti, ma non critici per il successo del progetto.
- Could: requisiti di scarsa importanza per il successo del progetto, ma se soddisfatti aumentano il gradimento dell'utente.
- Won't: requisiti che non influiscono sul successo del progetto (potrebbero diventare importanti nel futuro) dunque sono postponibili.

Il prodotto è descritto in base alle prestazioni e alle funzionalità che dovrà avere. A ogni fabbisogno corrispondono uno o più requisiti e ogni requisito è scritto in forma di *User Story*. Molteplici User Story insieme fanno parte di un'*epica* (o epic).

Nel product backlog gli elementi funzionali sono raggruppati in release, le quali sono rilasciate durante lo svolgimento del progetto. A questo punto si va a definire un **Release Planning Schedule**, cioè un piano a lungo termine delle release. Dunque, nel framework Scrum la pianificazione non è eliminata completamente, ma è una programmazione ad alto livello di rilascio delle release.

Il backlog dei requisiti si modifica nel tempo a ogni *Sprint*, una time box che solitamente copre da 2 a 4 settimane. Si passa allora a una programmazione di breve periodo, in cui si programma uno sprint alla volta partendo dai requisiti prioritari in cima alla lista del product backlog. Questo avviene attraverso l'attività di sprint planning, che richiede a sua volta l'esigenza di creare un backlog di attività da svolgere di breve periodo: lo **Sprint Backlog**.

Quest'ultima serie di attività è svolta n-volte, in base al numero di sprint da realizzare per completare il progetto. Al termine di ogni sprint si delibera una porzione di prodotto potenzialmente consegnabile al cliente che può essere accettata o meno dalla figura preposta.

2.1.2 I ruoli

Scrum è messo in azione dalla presenza di tre componenti fondamentali all'interno del team di sviluppo, che insieme formano lo Scrum Team (Schwaber & Sutherland, 2020):

- **Team di sviluppatori**

L'unità fondamentale di Scrum è un piccolo gruppo di persone, un Team in cui non ci sono sottogruppi o gerarchie, ma è composto da un'unità coesa di professionisti concentrati su un unico obiettivo alla volta: il Product Goal. È responsabile di tutte le attività correlate al prodotto, in particolare di realizzare incrementi utili e di valore a ogni Sprint.
Il Team è un gruppo multifunzionale, costruito in maniera tale che le competenze siano eterogenee, affinché siano presenti tutte le persone necessarie per svolgere tutte le attività di progetto. È autogestito, cioè i membri decidono come distribuire i compiti al loro interno, stabilendo anche quando e come. Il team è auto-organizzato, ovvero gli individui devono essere auto-motivati e devono accettare responsabilità più grandi per poter realizzare maggior valore. Infatti, l'auto-organizzazione porta a una migliore adesione e a una responsabilità condivisa da parte del team, con un conseguente aumento della motivazione e un miglioramento del livello delle prestazioni. Inoltre, il team deve essere collaborativo, dunque deve cooperare per trarre vantaggio dai contributi reciproci, con lo scopo di produrre qualcosa di più grande. Preferibilmente deve essere co-ubicato, ovvero i membri dovrebbero essere collocati tutti nello stesso ufficio o addirittura allo stesso tavolo di valoro, per facilitare la comunicazione faccia a faccia, sfruttare i vantaggi di un migliore coordinamento per risolvere i problemi e condividere la conoscenza.

Il Team deve essere abbastanza piccolo in modo da consentire l'agilità, ma anche abbastanza grande per portare a termine un lavoro significativo all'interno dello Sprint; solitamente è composto da 10 persone o meno.

- **Product Owner**

Il Product Owner deve massimizzare il valore del prodotto risultante dal lavoro svolto dallo Scrum Team. È l'anello di congiunzione tra gli stakeholders esterni e il team di progetto ed è responsabile per la gestione del Product Backlog, in quanto si occupa di tradurre le richieste del cliente in requisiti funzionali. Fornisce i dettagli e definisce la roadmap di prodotto (o release plan) aggiornando ogni settimana il product backlog. In ultimo, definisce i criteri di accettazione delle attività del teamwork attraverso l'ispezione degli incrementi in sede di Sprint Review.

- **Scrum Master**

È un vero e proprio leader al servizio dello Scrum Team e dell'organizzazione in generale. Lo Scrum Master è responsabile dell'efficacia del lavoro dello Scrum Team, occupandosi di allenare i membri spingendoli all'autogestione e aiutandoli a concentrarsi eliminando gli impedimenti che questi potrebbero riscontrare durante il lavoro. La sua funzione non è predittiva, bensì adattativa. Inoltre, deve battere il tempo dei lavori: esiste un passo di marcia costante che deve essere mantenuto per evitare sovrallocazioni o sottoallocazioni. Si deve assicurare che tutti gli eventi Scrum siano svolti in maniera corretta e produttiva e che siano mantenuti entro i limiti temporali previsti. Lo Scrum Master è responsabile del mantenimento della documentazione e affianca il Product Owner in diversi modi, in particolare nella gestione del Product Backlog, nella collaborazione con gli stakeholder e nell'assicurarsi che il team comprenda al meglio gli obiettivi e il dominio di progetto. In ultimo, aiuta l'organizzazione guidandola nell'adozione di Scrum, pianificando la sua implementazione e aiutando i dipendenti e gli stakeholder a comprendere e attuare questa metodologia di lavoro.

2.1.3 Le cerimonie

Nel framework Scrum, le riunioni prendono il nome di Cerimonie (Figura 4) e sono adattate da ogni azienda in base al proprio modo di lavorare (Schwaber & Sutherland, 2020).

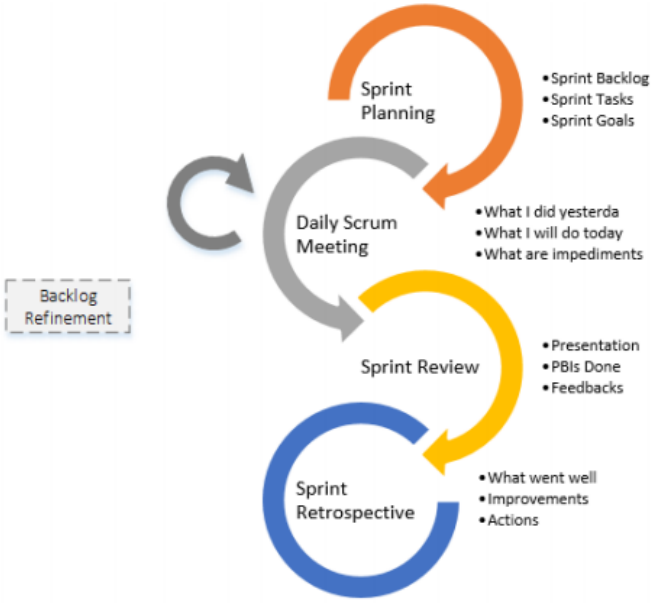

Figura 4: Cerimonie

- **Sprint Planning**

Lo Sprint Planning dà il via allo sprint, stabilendo il lavoro che deve essere svolto durante lo stesso. Il piano che ne risulta è frutto del lavoro collaborativo di tutto lo Scrum Team, del Product Owner e dello Scrum Master. Durante questa cerimonia possono essere invitate a partecipare anche altre persone esterne al team di sviluppo, in modo da ricevere consigli utili.

Durante lo Sprint Planning si stabilisce come il prodotto può aumentare il suo valore e la sua utilità. A questo punto, l'intero Scrum Team collabora per definire uno Sprint Goal in grado di comunicare il valore fornito agli stakeholder. Inoltre, durante lo Spint planning i membri del team selezionano gli elementi del Product Backlog da realizzare e stabiliscono come si svolgerà il lavoro: per ogni elemento selezionato, il team pianifica il lavoro necessario, stimando l'effort per sviluppare le attività identificate e creare un incremento che soddisfi la Definition of Done (definizione di fatto). Questo è spesso fatto decomponendo gli elementi del Product Backlog in elementi di lavoro più piccoli, della durata di un giorno circa.

- **Daily Scrum Meeting**

Il Daily Scrum Meeting (detto anche Daily Standup Meeting) è una riunione quotidiana che ha lo scopo di ispezionare lo stato di avanzamento dello sprint rispetto allo Sprint Goal e di adattare lo Sprint Backlog secondo le esigenze. L'evento solitamente ha una durata di circa 15 minuti e coinvolge lo Scrum Master e lo Scrum Team. Il meeting si tiene sempre nello stesso luogo e alla stessa ora (solitamente a inizio giornata), in ogni giorno lavorativo dello Sprint. Questa cerimonia consente di migliorare le comunicazioni, identificare gli impedimenti, promuovere un rapido processo decisionale e, conseguentemente, eliminare la necessità di ulteriori meeting. Inoltre, consente di implementare il principio dell'accoglienza del cambiamento e dell'adattamento tipico delle metodologie iterative.

- **Sprint Review**

Lo scopo della Sprint Review è quello di ispezionare il risultato dello Sprint e determinare degli adattamenti futuri. È un incontro che avviene al termine dello sprint, durante il quale lo Scrum Team presenta i risultati che possono essere approvati o rifiutati da Product Owner. Durante l'evento si discute dell'avanzamento verso il Product Goal, si raccolgono eventuali feedback e modifiche da apportare insieme agli stakeholder e, sulla base di ciò che è stato portato a termine, si collabora per definire i passi successivi. A tal proposito, il Product Backlog può essere adattato per intercettare nuove opportunità.

- **Sprint Retrospective**

La Sprint Retrospective è la cerimonia che chiude lo sprint con lo scopo di pianificare i metodi per incrementare la qualità e l'efficacia del lavoro. Lo Scrum Team ispeziona l'andamento dello sprint appena concluso relativamente agli individui, alle interazioni, ai processi, agli strumenti e alla Definition of Done. Inoltre, si identificano le cause di eventuali deviazioni. Il team discute di ciò che è andato bene durante lo sprint, di quali problemi ha riscontrato e come questi siano (o non siano) stati risolti. Identifica i cambiamenti più utili per migliorare la propria efficienza e questi possono persino essere aggiunti allo Sprint Backlog così da essere inglobati nello Sprint successivo. Dunque, l'obiettivo della cerimonia è quello di ridurre i rischi e massimizzare i risultati finali.

2.1.4 Gli Artefatti

Gli artefatti di Scrum rappresentano il lavoro svolto e il valore generato. Sono progettati in modo da massimizzare la trasparenza delle informazioni, così che ognuno abbia lo stesso livello di comprensione dell'artefatto.

- **Product Backlog**

Il Product Backlog (Figura 5) è un elenco prioritizzato di requisiti, caratteristiche e funzionalità che il prodotto deve rispettare per essere realizzato. I requisiti sono ordinati e sistemati in modo da formare più release da sviluppare nel corso del processo. Gli elementi possono essere completati (definiti "Done") dallo Scrum Team nell'arco di uno sprint e sono selezionati durante lo Sprint Planning.

Il Product Backlog evolve continuamente insieme al prodotto e all'ambiente: si aggiungono e si modificano dettagli, consentendo un costante adattamento. I progettisti responsabili dello svolgimento del lavoro e della realizzazione degli elementi del Product Backlog sono anche responsabili della stima della dimensione degli elementi e del lavoro necessario per realizzarli.

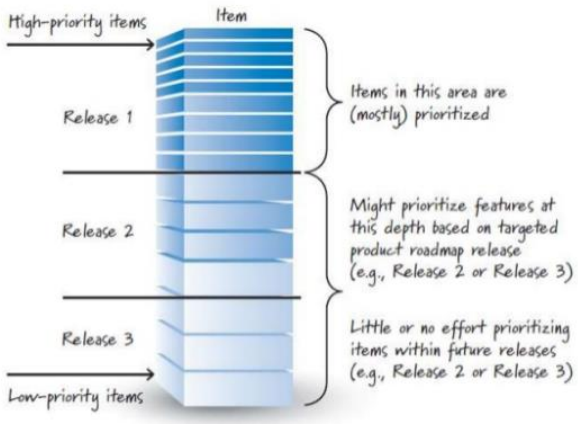

Figura 5: Product backlog

- **Sprint Backlog**

Il Product Backlog è tradotto nello Sprint Backlog (Figura 6), composto dall'insieme degli elementi che il Team di sviluppo deve realizzare per completare lo sprint.

Figura 6: Sprint backlog

Lo Sprint Backlog spesso dà luogo a una board, un tabellone nel quale compaiono i task classificati in "da completare", "in progress" e "completati". Tipicamente questo Sprint Board gestisce in modo semplice i task tramite post-it colorati sistemati all'interno di colonne. In alternativa può essere in formato virtuale, gestito tramite software condivisi in rete, consentendo un rapido accesso da parte di tutti gli interessati.

È un vero e proprio piano elaborato dagli sviluppatori, una fotografia di dettaglio visibile in tempo reale del lavoro realizzato durante lo sprint per raggiungere il Goal. La lista del lavoro da svolgere è selezionata a partire dagli elementi in cima al Product Backlog e contiene tutti i requisiti realizzabili all'interno di uno sprint. Ogni requisito richiede più o meno attività da svolgere, racchiuse all'interno di un kanban. Alla fine dello sprint tutte le attività devono essere terminate (sistemate in "Done"), altrimenti non è possibile deliberare.

- **Burndown Chart**

Il burndown chart (Figura 7) è un particolare tipo di grafico che da visibilità della velocità di avanzamento dei lavori. È aggiornato ogni giorno ed è uno strumento utile per monitorare il progresso degli sprint, mostrando la quantità di lavoro residuo nello sprint in corso.

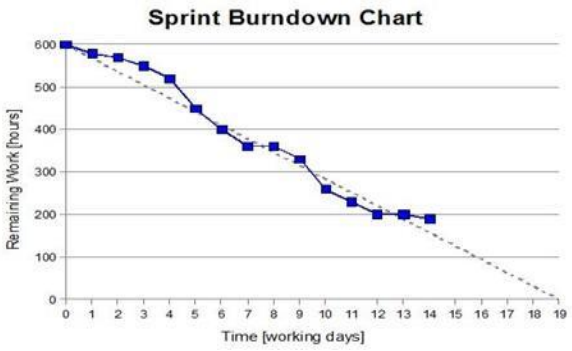

Figura 7: Burndown chart

Inizialmente sul grafico è tracciata la curva del flusso ideale di lavoro, quantificato tramite gli story points sotto l'ipotesi che sia mantenuto un passo di marcia costante. Gli story points sono dei punteggi, che possono essere pesati sia in termini di tempo che di complessità, e per ogni sprint è stimato il numero di punti storia necessari a completarlo. In corso d'opera si traccia una seconda curva che corrisponde a quanto effettivamente realizzato: in questo modo risulta se lo Scrum Team procede secondo quanto previsto, se ci sono stati degli impedimenti e dunque è in ritardo, oppure se è in anticipo con i lavori. Spesso le attività sono sconosciute, quindi i punti storia possono essere utilizzati come obiettivo a cui tendere.

2.1.5 Monitorare l'avanzamento verso il Goal

L'avanzamento verso il Goal finale è realizzato mediante il completamento di Sprint successivi. In particolare, al termine di ogni Sprint Review, il Product Owner registra il lavoro svolto e monitora eventuali ritardi rispetto alle milestone di progetto. In questo modo tiene traccia anche del lavoro totale rimanente, rendendo le informazioni sempre disponibili a tutti gli stakeholder. Generalmente si utilizzano diversi strumenti di appoggio per monitorare lo stato di avanzamento dei lavori, come il Burndown Chart discusso nel paragrafo precedente, che registra la velocità con cui "si brucia" la tabella di marcia. Questi strumenti sono utili per prendere decisioni future in base a quanto si è verificato, tuttavia non sono in grado di predire il futuro, soprattutto quando si tratta di ambienti molto dinamici e complessi.

Ogni sprint rilascia una porzione di prodotto, uno sviluppo incrementale di ciò che deve essere realizzato. A questo punto è importante analizzare nel dettaglio in cosa consiste un incremento di prodotto e quando uno sprint può essere definito "Fatto", ovvero concluso (Schwaber & Sutherland, 2020).

Un **Incremento** rappresenta una milestone per raggiungere il Product Goal e si aggiunge in maniera cumulativa a tutti gli incrementi rilasciati in precedenza. È sottoposto a una verifica molto accurata, per garantire che tutti gli incrementi insieme funzionino correttamente.

In particolare, l'incremento di prodotto consiste nell'insieme degli elementi del Product Backlog che sono rilasciati nel corso dello sprint e che rispettano la definizione di "fatto" stabilita a monte. Per rispettare questa definizione, l'incremento deve essere utilizzabile e

dimostrabile agli stakeholders e deve fornire un maggior valore al prodotto, consentendo al team di fare un passo avanti verso la consegna del prodotto finale.

L'incremento deve essere consegnato e accettato dal Product Owner: solo in questo caso lo sprint si può considerare concluso ed è possibile procedere con lo sviluppo, passando agli altri elementi del Product Backlog.

Andando ad analizzare la **Definition of Done** (Definizione di fatto – DoD), questa chiarisce quando un determinato lavoro deve intendersi completato. Corrisponde a una descrizione formale dello stato dell'incremento quando questo soddisfa le metriche di qualità richieste per il prodotto. Se un elemento del Product Backlog non soddisfa tale definizione, non può essere rilasciato e nemmeno presentato durante la Sprint Review. È importante che tutti i membri abbiano una visione condivisa di ciò che si intende per "Fatto", la cui definizione può variare significativamente da Team a Team. In questo modo, gli individui sono in grado di comprendere quanti elementi del Product Backlog possono essere selezionati per pianificare uno Sprint. Infatti, lo scopo di ogni Sprint è proprio quello di rilasciare degli incrementi funzionali e testabili, che poco per volta vadano a completare le funzionalità del prodotto finale. Inoltre, la Definition of Done è importante in quanto fornisce uno standard di base che deve essere rispettato da tutto lo Scrum Team.

La DoD non è statica, ma può essere articolata e dettagliata nel corso dello sviluppo includendo dei criteri più rigorosi per ottenere una qualità superiore del progetto e, durante la pianificazione di uno Sprint, può essere aggiornata in modo da tener conto dell'esperienza maturata dal team.

2.2 Confronto tra i metodi di sviluppo tradizionali e l'Agile

I metodi tradizionali, anche definiti Waterfall (a cascata), sono basati su uno sviluppo lineare e differiscono notevolmente dagli approcci agili.

I primi consistono in un sistema completo, che va dall'idea iniziale al lancio, dunque sono un vero e proprio processo di macro-pianificazione. Sono caratterizzati da step sequenziali che seguono un flusso lineare: le fasi di sviluppo procedono una dopo l'altra e non è possibile passare alla fase successiva se prima non è stata conclusa quella precedente. Fin dall'inizio c'è

una pianificazione dettagliata, da seguire alla lettera durante tutto lo sviluppo, con un'attenzione particolare ai vincoli di progetto definiti a monte. La produzione della documentazione e la realizzazione dei test avvengono al termine di ogni fase, favorendo il mantenimento della qualità del progetto. Tuttavia, i difetti di progettazione possono essere identificati molto tardi e, se i requisiti non sono chiari e ben specificati, è difficile portare a completamento il progetto con successo (Lavecchia, s.d.).

Al contrario, l'Agile è un approccio di gestione caratterizzato da uno sviluppo iterativo, che opera attraverso gli sprint, ed è basato sulla micro-pianificazione, ricorrendo a una pianificazione di massima all'inizio del processo e a una pianificazione di dettaglio per ogni sprint. Infatti, volendo massimizzare il lavoro non svolto, secondo la filosofia Agile non è efficiente pianificare anticipatamente ogni dettaglio in quanto in un contesto dinamico tutto potrebbe cambiare rapidamente. Per le stesse ragioni è anche un approccio "leggero" in termini di documentazione. Abbraccia i cambiamenti, volendo porre maggior attenzione alla soddisfazione dei requisiti e alla massimizzazione del valore per i clienti.

Sebbene anche le metodologie tradizionali di project management possano generare prodotti di ottima qualità, talvolta, a causa della loro elevata rigidità, può accadere che il progetto si concluda con successo, ma il cliente non sia pienamente soddisfatto del risultato e che quindi i benefici attesi non siano stati raggiunti. Infatti, le metodologie tradizionali consegnano il valore in maniera simultanea al rilascio del prodotto finale e questo avviene solo al termine delle fasi di analisi, progettazione, implementazione e test (Lavecchia, s.d.). Al contrario, la consegna di valore nei progetti agili avviene poco per volta in ogni sprint, cercando continuamente il feedback da parte del cliente così da correggere il tiro in caso di deviazioni. In questo caso, il valore consegnato assume la forma di una curva a "scalini", dovuto proprio dal fatto che il prodotto è realizzato in maniera incrementale.

Anche la gestione della qualità è differente per i due approcci: nei progetti Waterfall, gli utenti rendono chiare le proprie aspettative e il project manager ha il compito di tradurle in requisiti ben definiti e misurabili. Successivamente, a seguito della definizione di un piano di lungo periodo, il gruppo di lavoro sviluppa il prodotto incorporando i requirements, il tutto all'interno di un periodo di tempo predeterminato piuttosto fisso. Eventuali modifiche dei criteri di qualità possono essere gestite solo in modo formale, attraverso un processo di gestione del cambiamento, che implica la stima degli impatti su quanto già realizzato e che richiede l'approvazione di tutti gli stakeholder di progetto (Lavecchia, s.d.).

Nei progetti agili, i criteri di accettazione degli sprint sono stabiliti dal Product Owner che agisce da intermediario e rappresenta direttamente la Voce del Cliente: uno sprint è da considerarsi concluso se il suo risultato è accettato dal Product Owner. Inoltre, quest'ultimo può apportare dei cambiamenti ai requisiti nel backlog in modo da stare al passo con i bisogni dei clienti. Lo Scrum Team dovrà prenderli in carico, incorporando le modifiche richieste nello sprint in corso oppure in quello successivo.

Un'altra differenza importante è legata alla modalità di lavoro (Figura 8): nell'ambito più tradizionale, definito anche plan-driven, tempi e budget di realizzazione sono determinati in base ai requisiti che il prodotto deve soddisfare. Al contrario, l'approccio Agile è value- driven, in cui il tempo e il budget sono determinati all'interno di ogni sprint e le caratteristiche del progetto da realizzare sono flessibili e si adattano a questi vincoli.

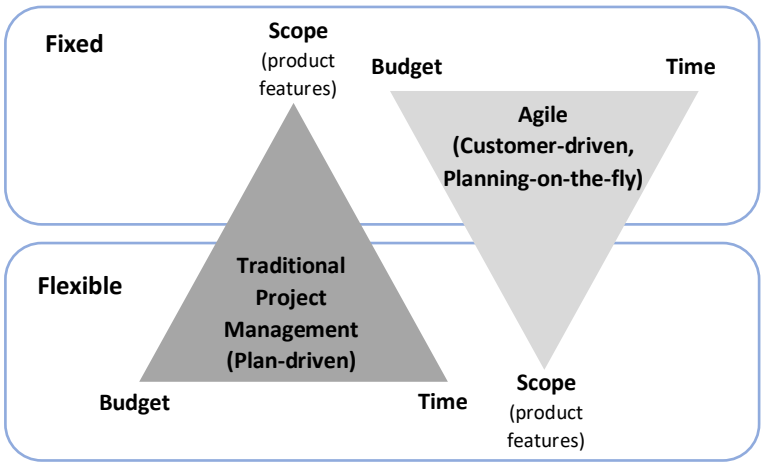

Figura 8: Elementi fissi e flessibili nei due approcci.

In ultimo, il metodo Waterfall è interfunzionale, ovvero coinvolge più funzioni aziendali contemporaneamente, a partire dal marketing, alle vendite, fino al personale tecnico. È anche una guida all'azione in quanto determina le best practice specifiche in ogni fase: dare ascolto alla voice of custromer, costruire un solido business case, progettare un lancio efficace e così via. Al contrario, lo sviluppo Agile è progettato per aiutare gli individui a creare rapidamente un prodotto funzionante con una continua validazione da parte del cliente. Una volta che un progetto di sviluppo è stato approvato e i suoi requisiti iniziali mappati, Agile si concentra sull'esecuzione. I requisiti di prodotto che nelle fasi iniziali non erano del tutto noti sono rivelati e convalidati in corso d'opera attraverso l'iterazione, mentre i requisiti che inizialmente si pensavano importanti, ma non si rivelano tali, possono essere eliminati.

2.3 Agile per lo sviluppo di prodotti fisici

Fino a qualche decennio fa le organizzazioni di successo erano quelle più stabili, che mantenevano dei rapporti consolidati con il proprio mercato di riferimento e i propri partner. Per questo motivo al giorno d'oggi nel campo dello sviluppo hardware sono prevalentemente impegnati gli approcci più tradizionali, sequenziali, che risultano essere altamente efficaci ed efficienti in condizioni stabili. Ciononostante, non sono in grado di far fronte a un ambiente in continua evoluzione, che richiede prontezza e celerità, con requisiti che cambiano frequentemente. Infatti, volgendo lo sguardo al panorama competitivo attuale si può osservare come i modelli tradizionali siano ormai ampiamente superati, fatte salve pochissime eccezioni.

A causa della complessità dei prodotti e delle organizzazioni e della necessità di essere competitivi in condizioni **VUCA** (Volatility, Uncertainty, Complexity, Ambiguity) molte aziende manifatturiere hanno iniziato a comprendere il vero valore dell'Agile, ormai mainstream nel mondo software. Questo approccio è sempre di più implementato all'interno dei processi di sviluppo al fine di garantire un'elevata reattività ai cambiamenti in un contesto dominato dalle incertezze, contrastando la mancanza di reattività dei metodi più tradizionali guidati dal piano (Albers, et al., 2019).

La diffusione continua di questo paradigma ha permesso di contaminare anche il contesto di sviluppo dei prodotti tangibili. Tuttavia, sono sorte una serie di sfide, principalmente a causa delle proprietà fisiche dei sistemi hardware (Schmidt T. , Weiss, Paetzold, Stefan, & Kristin, 2018a). Queste sfide derivano soprattutto dal fatto che gli approcci agili sono stati pensati per il contesto di sviluppo software e non tengono conto dei requisiti fisici dei prodotti.

In ultimo, la maggior parte delle aziende intraprende un percorso di implementazione dell'Agile nello sviluppo hardware attraverso le esperienze pregresse nel mondo software: per queste organizzazioni l'agilità non è un concetto nuovo, ma hanno solamente bisogno di adattarlo ai diversi contesti. Per tale ragione, si può dedurre che le imprese che realizzano prodotti con un elevato livello di componenti software incorporati implementano l'Agile con più facilità rispetto alle aziende che producono prodotti con spiccate funzionalità meccaniche. Dunque, le esperienze nell'industria del software sono spesso il punto di partenza per l'applicazione dello sviluppo Agile per l'hardware (Schmidt, et al., 2019).

2.3.1 Hardware e software a confronto

I contesti di sviluppo hardware e software differiscono notevolmente in quanto si occupano di prodotti ampiamenti differenti, dove i primi sono tangibili, mentre i secondi hanno una natura più astratta. Di seguito sono riportate le differenze chiave che possono essere identificate (Thompson, s.d.):

- Il software si può modificare più facilmente rispetto all'hardware, sia in termini operativi che di costo. Infatti, il costo del cambiamento è molto più alto per l'hardware che per il software.
- I prodotti software evolvono attraverso molteplici rilasci mediante un processo di accrescimento e refactoring (aggiunta di nuove funzionalità e riscrittura della logica esistente per supportarle). I prodotti hardware, costituiti principalmente da componenti fisici, non possono essere sottoposti allo stesso processo e non consentono di apportare nuove funzionalità che richiedano modifiche importanti.
- Il design di un prodotto tangibile è guidato in gran parte dalle scelte architetturali. Poiché il costo del cambiamento è elevato, è necessario che queste scelte siano prese con largo anticipo rispetto alle tempistiche richieste dai prodotti software.
- Il costo di sviluppo per il software è relativamente costante nel tempo, mentre per l'hardware aumenta rapidamente, registrando un picco verso la fine del ciclo di sviluppo.
- A causa dei punti di cui sopra, per il software è possibile apportare cambiamenti importanti, anche a metà del processo di sviluppo, senza incorrere in interruzioni e sprechi massicci. Al contrario, i tentativi di apportare tali modifiche nello sviluppo hardware hanno un costo molto più elevato, coinvolgendo da un lato i costi affondati già sostenuti, dall'altro i piani di approvvigionamento che devono essere rivisti.

D'altra parte, è possibile identificare alcune similitudini tra prodotti hardware e software:

- Producono degli output dopo aver ricevuto degli input;
- Possiedono requisiti funzionali e non funzionali;
- Qualsiasi rappresentazione delle specifiche del prodotto porta a una struttura ad albero, in quanto le caratteristiche principali sono scomposte in caratteristiche via via più fini.

2.3.2 Stato dell'arte

Nel corso degli ultimi dieci anni, numerosi studiosi hanno avviato delle ricerche per comprendere se i principi agili fossero applicabili anche al campo di sviluppo dei prodotti fisici, viste le grandi differenze rispetto ai prodotti puramente software. La principale difficoltà consiste nel fatto che al giorno d'oggi non esistono ancora dei veri e propri modelli da seguire alla lettera, ma le aziende si trovano a dover applicare alcune pratiche in base al contesto in cui operano.

È importante evidenziare che la maggior parte degli studi reperibili in letteratura hanno preso in esame l'evoluzione dello sviluppo Agile per prodotti meccatronici, ovvero prodotti che includono sia una componente hardware che una componente software. Il motivo è che, come già affermato in precedenza, gran parte delle aziende hanno dapprima implementato il paradigma all'interno delle attività di sviluppo software per poi estenderlo anche allo sviluppo di prodotti tangibili. D'altro canto, i prodotti meccatronici più degli altri necessitano costantemente di una forte spinta innovativa.

I ricercatori hanno identificato un crescente interesse per l'applicazione dell'Agile nel contesto dell'innovazione e dello sviluppo di nuovi prodotti in diversi settori industriali. Già a partire dal 2012, Oversen (Oversen & Dowlen, 2012) avviò le prime indagini su come i metodi di sviluppo Agile potessero essere incorporati nella progettazione di prodotti tangibili. Riassunse un'ampia serie di sfide associate all'applicazione di questi framework, in particolare di Scrum, mediante il confronto dei risultati derivanti da uno studio condotto all'interno di 7 aziende danesi e dei progetti degli studenti di MSc Design and Manufacturing Management presso la London South Bank University. Dalle analisi risultarono dei vantaggi significativi derivanti dall'utilizzo di metodi agili, quali l'elevato coinvolgimento dei clienti, la riduzione dei costi di guasto, il feedback tempestivo e la comunicazione efficace. Per quanto riguarda il rispetto delle tempistiche di progetto, Oversen convenne che i processi agili consentissero l'avvio rapido e l'apporto di adattamenti continui, senza che le modifiche dovessero passare attraverso documentazioni standard e rallentare il processo. D'altra parte, rilevò che i vincoli riscontrati per i prodotti fisici limitassero l'applicazione dell'Agile a prodotti semplici, rivolti a un ristretto numero di clienti favorevoli a collaborare con queste nuove pratiche. Inoltre, le sfide inclusero le complicazioni legate all'integrazione dei valori e delle procedure agili in un ambiente di sviluppo tradizionale, come la preoccupazione della

perdita di potere del management e le inerzie interne ai team di sviluppo, e le difficoltà di suddivisione delle attività di sviluppo in task da rilasciare in maniera incrementale.

Un'altra fonte interessante è il documento di ricerca di (Conforto, Salum, Amaral, Silva, & Almeida, 2014) che si basa su un'indagine esplorativa sull'uso di pratiche di gestione dei progetti agili e la presenza di fattori abilitanti (detti abilitatori) in 19 aziende di medie e grandi dimensioni operanti in diversi settori industriali, con riferimento a progetti innovativi. Con "abilitatori" gli autori indicarono una serie di fattori necessari per l'implementazione dell'Agile, che possano influenzare l'uso di pratiche, tecniche e strumenti, secondo l'approccio APM (Agile Project Management). Gli autori riconobbero che alcune delle pratiche agili dipendevano pesantemente dall'ambiente dell'organizzazione e dal contesto del progetto in cui erano utilizzate. I risultati del documento mostrano la presenza di alcuni abilitatori agile all'interno delle aziende e quindi l'opportunità per adattare la teoria a un ambiente diverso da quello puramente software. Inoltre, lo studio ha identificato le potenziali barriere per l'implementazione dell'Agile all'interno delle industrie più tradizionali, quali la difficoltà di dedicare un team a un unico progetto, di co-localizzare tutti i membri del team nella stessa stanza, di creare grandi team multidisciplinari (con tutte le competenze di progetto coinvolte) e di coinvolgere clienti e fornitori con un alto grado di influenza nello sviluppo del progetto. In ultimo, gli autori proposero degli spunti di riflessione e di indagini future, con riferimento alla possibilità di applicare modelli di gestione "ibridi" includendo l'Agile all'interno degli approcci tradizionali, al fine di bilanciare le esigenze di "agilità" con le barriere identificate.

(Stare, 2014) condusse una ricerca quantitativa empirica che coinvolse 5 imprese manifatturiere slovene, indagando su 21 progetti di sviluppo prodotto, al fine di determinare se queste aziende utilizzassero già una qualsiasi delle pratiche agili e quale fosse il loro contributo effettivo sul successo del progetto. La ricerca mostrò che all'interno dei progetti esaminati erano già seguite molte pratiche agili, tuttavia non era ancora possibile parlare di un approccio sistematico vero e proprio per lo sviluppo di nuovi prodotti, bensì di approcci parziali, a cui facevano volontariamente ricorso i singoli team o manager sulla base delle best practice dei progetti passati. Il campione poco numeroso non consentì di applicare in modo efficace un'analisi multivariata sull'influenza delle prestazioni e sul successo del progetto dovute alla presenza di pratiche agili. Questo studio non analizzò l'aspetto della collaborazione con il cliente, ripresa invece da (Lehnen, Schmidt , & Herstatt, 2016) nella loro ricerca sulla gestione Agile dei progetti denominati "lead user", ovvero progetti il cui

cuore risiede nell'integrazione continua degli utenti nel processo di sviluppo prodotto per far fronte alle crescenti necessità di innovare con successo. Infatti, la partecipazione attiva dei clienti è estremamente utile alle aziende per sviluppare nuovi prodotti di successo, maggiormente allineati alle future esigenze del mercato. Gli autori condussero uno studio esplorativo in 249 aziende tedesche e, per rafforzare i risultati, svilupparono 6 interviste approfondite, concludendo che un modello basato su una gestione agile dei progetti fornisce un'implementazione efficiente e flessibile dei progetti lead user nella pratica.

(Cooper & Sommer, 2016) introdussero il concetto di Agile-Stage-Gate, un nuovo metodo di lavoro che prevede l'integrazione dell'Agile nelle fasi di sviluppo, mentre il processo continua ad essere intervallato da Gate intermedi. Gli autori spiegarono che questo nuovo sistema nacque dalla necessità delle imprese manifatturiere di adattarsi ai cambiamenti, ma di non stravolgere completamente il proprio modo di lavorare, anche a causa di vincoli che non possono essere facilmente superati. L'articolo analizza quindi l'uso del modello ibrido e i risultati raggiunti dai primi utenti che hanno implementato questo sistema. Le indagini si orientarono a produttori appartenenti a diverse industrie, dall'alimentare alle attrezzature pesanti. In ultimo, gli autori indagarono sugli adattamenti che l'implementazione dell'Agile-Stage-Gate per prodotti hardware ha dovuto subire rispetto a quanto applicato nel contesto di sviluppo software, tra cui la ridefinizione di uno "sprint fatto" e i metodi per consegnare versioni di prodotto (anche dette "protocept") che consentano di ottenere un feedback continuo dai clienti.

Inoltre, (Schmidt, Weiss, & Paetzold, 2018b) analizzarono l'attuale effetto hype dei benefici derivanti dall'implementazione dell'Agile nei processi di sviluppo dei prodotti fisici, mostrando i progressi sul ciclo Hype di Gartner. Gli autori rilevarono l'esistenza di un divario nella comprensione dello sviluppo iterativo tra il mondo accademico e quello pratico, in quanto l'accademia definisce l'Agile come l'abilità di reagire costantemente e rapidamente ai cambiamenti, mentre le aziende vogliono implementare approcci agili per sviluppare in modo più economico e veloce, cercando di ridurre il time-to-market. Inoltre, il successo riscontrato dalla metodologia nel mondo software può essere fonte di frustrazione per il contesto di sviluppo hardware. Per questi motivi l'introduzione dei metodi agili nel campo di sviluppo dei prodotti tangibili è di circa 10 anni indietro rispetto allo sviluppo software.

(Atzberger & Paetzold, 2019) ripresero ed esaminarono criticamente i contributi delle ricerche passate e delle conoscenze preesistenti sull'applicabilità dell'Agile al di là dell'industria del software. In particolare, aggiornarono i risultati della ricerca di Oversen, integrando le nuove sfide emerse negli ultimi anni ed evidenziando i progressi rispetto alla situazione di 7 anni prima. Conclusero che, rispetto al Manifesto originale, è necessario un allineamento dei metodi attuali e un adattamento ai diversi contesti di sviluppo hardware.

Qui sono riportati i principali articoli trovati in letteratura e analizzati per fornire una base solida al lavoro di tesi. I termini di "sviluppo hardware" e "sviluppo di prodotti fisici" sono utilizzati in modo analogo e fanno riferimento a prodotti che di base hanno una natura fisica, ma possono incorporare anche elementi software all'interno del loro dominio.

A seguito del lavoro di revisione degli articoli è emerso che nel corso degli anni il numero di sfide rilevate per l'implementazione dell'Agile al contesto di sviluppo hardware sono aumentate anziché diminuite; questo è dovuto alla mancanza di maturità del tema sul campo e a una crescente applicazione delle pratiche agili, accompagnata da continue ricerche via via più approfondite.

2.3.3 Implementazione dello sviluppo Agile

L'implementazione dell'agilità in un'azienda è un processo lungo, che può essere spiegato sia in termini di diffusione organizzativa che operativa. La Figura 9 mira a visualizzare le fasi principali di questo processo (Schmidt, et al., 2019): nello schema sono individuati diversi livelli di transizione, indice dello stato di avanzamento dell'implementazione dell'agile nello sviluppo prodotto. Questi livelli assumono un valore da 1 a 5 man mano che l'applicazione dell'approccio progredisce:

- Le aziende di livello 1 hanno implementato lo sviluppo agile dell'hardware in un primo progetto pilota (nell'ambito dello sviluppo di un singolo prodotto);
- Le aziende di livello 2 hanno esteso la simulazione a più progetti, sempre nell'ambito di sviluppo di un singolo prodotto;
- Le aziende di livello 3 hanno implementato l'Agile nello sviluppo completo di un singolo prodotto;

- Le aziende di livello 4 l'hanno esteso allo sviluppo di più prodotti;
- Le aziende di livello 5 non si sono limitate all' R&D, ma hanno esteso il concetto di agilità anche ad altri reparti, come per esempio la produzione e il marketing.

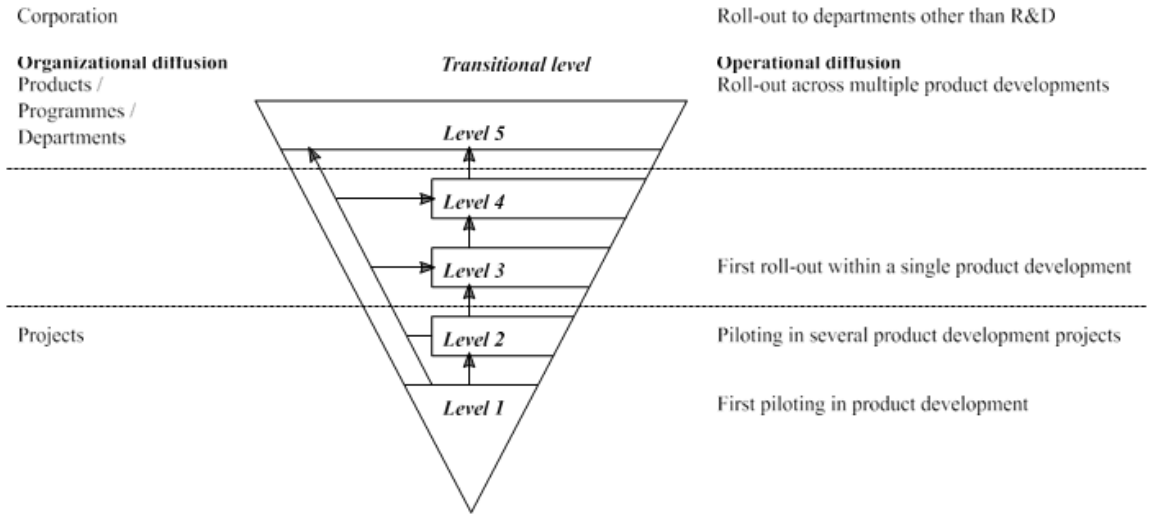

Figura 9: Livelli di transizione per l'implementazione dell'Agile nello sviluppo hardware

In termini di diffusione operativa, un'azienda alle prime armi con l'Agile inizia con la sperimentazione in un progetto di sviluppo prodotto pilota (Livello 1), per poi impegnarsi nella sperimentazione in due o più progetti di sviluppo (Livello 2). Successivamente, avviene il passaggio ad altri reparti all'interno dell'R&D, con il lancio dell'Agile in un primo progetto di sviluppo prodotto completo (Livello 3) e successivamente in più progetti, implementando così l'agilità in tutto il reparto R&D (Livello 4). L'ultimo passo è anche quello più lungo e determina una transizione completa, ovvero il roll-out dell'agilità in altri reparti aziendali oltre a quello di ricerca e sviluppo.

In termini di diffusione organizzativa, il concetto di transizione verso l'Agile si caratterizza in 3 fasi principali: i livelli 1 e 2 riguardano lo sviluppo hardware agile a livello di progetto, mentre i livelli 3 e 4 riguardano il roll-out tra diversi reparti o prodotti. Il passaggio a una società completamente agile si verifica nella fase finale, ovvero al livello 5.
È importante ricordare che un'azienda non deve necessariamente iniziare dal livello 1 e seguire lo schema in modo graduale, ma può iniziare dai livelli più alti e saltare determinati passaggi. Inoltre, la maggior parte delle aziende si impegna a diffondere lo sviluppo agile su

più livelli contemporaneamente: mentre completano il livello inferiore, iniziano con l'implementazione al livello successivo.

Uno studio (Schmidt, et al., 2019) ha monitorato l'interdipendenza tra le dimensioni dell'azienda e il livello di transizione in cui essa si trova. I risultati sono rappresentati in Figura 10:

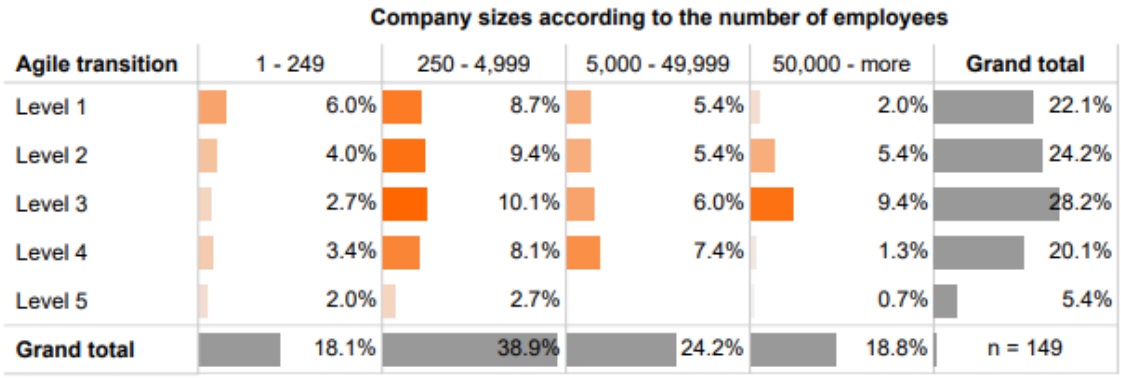

Figura 10: Dimensioni dell'azienda vs. livello di transizione Agile

Dai risultati è possibile comprendere che:

- La maggior parte delle aziende si collocano al livello 3 e le società di livello 5 sono ancora rare.
- Le aziende al livello 5 sono tutte PMI.
- Il 50% delle grandi aziende che hanno partecipato all'indagine si collocano al Livello 3.

Di conseguenza si può osservare che le PMI sono più avanti nella transizione agile e questo è sicuramente dovuto dalla minore presenza di ostacoli dimensionali, che invece si trovano ad affrontare le aziende più grandi. Infatti, queste ultime devono far fronte a scogli notevolmente maggiori e quindi non sono così "avanti" come le PMI.

L'implementazione dello sviluppo Agile non è rapida, ma richiede diversi anni. Secondo i dati, un'azienda probabilmente ha bisogno di più di 5 anni per raggiungere il Livello 5, ovvero il roll-out in tutta l'organizzazione.

Agile è progettato per gestire progetti di sviluppo più dinamici, che devono affrontare mercati fluidi, con le esigenze dei clienti che cambiano velocemente. L'incertezza e l'ambiguità delle informazioni sono ben gestite in questo nuovo sistema, al contrario dei

metodi più tradizionali. Il sistema è adatto a tutti i progetti di sviluppo, tuttavia, date le difficoltà di implementazione della metodologia (analizzate nel dettaglio nel paragrafo seguente) potrebbe essere controproducente applicarla a tutti i progetti all'interno dell'azienda. Infatti, laddove i mercati e le esigenze dei clienti sono noti e stabili, la struttura lineare dei metodi Waterfall non sono è svantaggio, dunque possono essere più efficienti. Agile richiede risorse significative e un team completamente dedicato al progetto (o quasi), che potrebbero non essere disponibili per semplici progetti incrementali. Riservare questa metodologia per i progetti che ne hanno veramente bisogno può aiutare a garantire la disponibilità delle risorse. Pertanto, la maggior parte delle aziende limita l'applicazione di Agile ai suoi progetti di sviluppo più importanti, più grandi, ma soprattutto più innovativi o più rischiosi (Cooper & Sommer, 2016).

2.3.4 Sfide nell'applicazione

L'implementazione degli approcci agili nello sviluppo prodotto vede numerose sfide, in virtù del fatto che le pratiche devono essere trasferite a partire dal contesto di sviluppo software. Con il passare degli anni il numero di sfide non è destinato a diminuire proprio a causa di studi sempre più specifici, della mancanza di un modello ben definito da seguire e della scalabilità dell'approccio Agile in tutti i reparti dell'azienda.

Di seguito sono riportate le principali sfide associate allo sviluppo agile per l'hardware, elencate a partire dai risultati di una ricerca (Atzberger & Paetzold, 2019) basata su una profonda revisione della letteratura. Nel complesso, i risultati sono stati raggruppati dagli autori in 4 categorie principali: vincoli di fisicità (CoP), mindset, scaling e distribuzione del team. Gli aspetti chiave relativi a ciascuna categoria sono riassunti in Tabella 1.

Tabella 1: Sfide attuali associate allo sviluppo Agile dell'hardware

NO.	CHALLENGE	CATEGORY
1	Realization of potentially shippable increments	**CoP**
2	Technical feasibility to produce prototypes	
3	Inability to break down the product into modules	
4	External dependencies	
5	Production of tools	
6	Documentation and certifications	
7	Specialisation of the individual	
8	Synchronisation of the domains	
9	Frequent stakeholder feedback	
10	Establishing an agile mindset	
11	Proper education and training	**Mindset**
12	Incorporating an agile team into a classical company structure	
13	"Prince problem"	
14	Commitment of the top management	
15	Commitment of the middle management	
16	Multi-project management	
17	Internal process models	
18	Transfer of (methodological) knowledge	**Scaling**
19	Structure of the company	
20	Silo mentality	
21	Mindset change of the organization	
22	Adaptation to the company-specific values	
23	Communication of distributed teams	**Team Distribution**
24	Usage of communication tools	
25	Ethical and cultural differences	

- **Vincoli di fisicità (Constraints of Physicality - CoP)**

Le sfide legate ai vincoli di fisicità sono sicuramente le più rilevanti. Data la natura iterativa e incrementale dello sviluppo Agile, il più grande ostacolo è (1) **realizzare incrementi potenzialmente spendibili in un'unica iterazione,** poiché i prodotti di natura fisica richiedono più tempo rispetto al software per essere realizzati. Inoltre, anche la creazione di un prototipo per testare le incertezze è difficilmente realizzabile, soprattutto in alcuni rami industriali, dunque (2) **la fattibilità tecnica di produrre prototipi** in un breve lasso di tempo è una delle preoccupazioni principali. Un ulteriore aspetto che accompagna questo problema

è (3) **l'incapacità di scomporre il prodotto in moduli da testare**.
Data la complessità di molti prodotti, soprattutto dei sistemi meccatronici in cui è necessario prendere in considerazione l'interazione tra hardware, software ed elettronica, entrano in gioco (4) **le dipendenze esterne** dai fornitori o da altri enti che hanno un ruolo nello sviluppo. Infatti, solitamente molti componenti non sono prodotti internamente e questa è una delle principali cause dell'incapacità di realizzare dei prototipi. Inoltre, è possibile che si verifichino dei ritardi nelle forniture degli ordini (come chip, circuiti stampati e tutti gli altri componenti che non sono realizzati internamente) e dopo la consegna, i pezzi devono ancora essere implementati e testati. Diversi componenti, anche se solo ai fini di realizzare un prototipo, non possono essere realizzati con materiale alternativo e questo porta a dipendenze per quanto riguarda (5) **la produzione di strumenti**, questione che può richiedere molto tempo.
I test su componenti non possono essere sempre completati in brevi tempi ciclo, soprattutto nei settori in cui l'interazione del prodotto con gli esseri umani può generare un problema potenzialmente letale in caso di guasto o, in generale, nei settori altamente regolamentati (come può essere quello medicale). A questo punto entrano in gioco (6) **la documentazione e le certificazioni** che richiedono tempo per essere ottenute e devono essere rilasciate da organizzazioni esterne all'azienda, generando ulteriori dipendenze esterne.
(7) **La specializzazione degli individui**, quindi la possibilità di avere diversi team di sviluppo dedicati ai singoli progetti è impegnativa, soprattutto quando si tratta di sviluppare sistemi altamente complessi, come possono essere i prodotti meccatronici che richiedono l'incorporazione di tre domini differenti in un unico sistema. Infatti, secondo i principi del Manifesto, il core team deve rimanere intatto dall'inizio alla fine del progetto e quindi gli individui rimangono bloccati "nel team" durante tutto il progetto, fino alla revisione post-lancio. I progettisti devono avere un profilo desiderato e non è semplice avere la disponibilità di un gran numero di risorse esperte.
Data la complessità dei prodotti hardware, (8) la **sincronizzazione dei domini e il coordinamento delle interfacce** sono provocatori. Un'ulteriore difficoltà risiede nell'ottenere un (9) **feedback frequente da parte degli stakeholder**, poiché gli individui che abitualmente lavorano secondo modelli di sviluppo classici non vedono un reale vantaggio nel ricevere feedback su prototipi che ancora non sono funzionali. Secondo loro, un prototipo è qualcosa di "vicino all'essere pronto", che soddisfa la maggior parte dei requisiti funzionali come un MVP.

Le sfide più grandi risiedono nei lunghi processi di produzione e quindi nella realizzazione di incrementi consegnabili all'interno di un'iterazione, così come la suddivisione del prodotto in incrementi significativi (Schmidt, et al., 2019). Alla luce di tutto ciò, i vincoli di fisicità ostacolano la mera traduzione dei principi Agile dal dominio software a quello hardware.

- **Mindset**

Un altro grande ostacolo risiede nella mentalità, termine superordinato per indicare la comprensione dell'agilità e l'integrazione della sua cultura sia per il singolo individuo sia per l'azienda (intesa come un insieme di individui).

A partire dagli individui, una delle sfide principali è (10) **stabilire una mentalità agile** nelle loro teste. Infatti, le persone tendono a fraintendere cosa sia effettivamente l'agilità, soprattutto quando si introduce il concetto a un team già esistente, che ha sempre lavorato seguendo un piano. In questa situazione, è impegnativo ripensare a come svolgere lo stesso compito di prima, ma con un approccio diverso. Deve essere chiaro che lavorare in modo agile è un cambiamento in termini di collaborazione, garantendo più libertà all'individuo, ma anche più responsabilità a ogni membro del team. (11) **Una corretta istruzione e formazione** e la disponibilità ad accettare questo nuovo stile di lavoro sono aspetti essenziali per essere compresi da tutti all'interno del team. L'uso di buone pratiche, tra cui le cerimonie Scrum come il daily stand-up meetings e la sprint retrospective, hanno lo scopo di migliorare la trasparenza ed evidenziare i problemi, in modo che possano essere discussi e superati all'interno del team. Inoltre, è importante cercare di mantenere viva una cultura "open-error ", in cui gli errori non sono puniti, ma vengono usati come fonte di insegnamento per le situazioni future. Tuttavia, se questa mentalità non è comunicata e messa in pratica nel modo corretto, gli individui possono rifiutarsi di parlare degli errori per paura di essere ammoniti. In tal caso, lavorare in modo agile è destinato a fallire.

Per quanto riguarda la cultura aziendale, è importante superare il pensiero tradizionale. Quando si introduce l'agilità e la si mette in pratica in un progetto pilota, è necessario considerare diversi aspetti che possono portare a eventuali sfide. Innanzi tutto, l'introduzione dell'agilità in un'azienda deve essere intesa come un vero e proprio processo, soprattutto quando si tratta di un'azienda matura che ha alle spalle una storia, in cui predominano tradizioni e routine. (12) **Incorporare un team agile in una struttura aziendale classica** spesso porta a problemi di interfaccia con i dipartimenti circostanti,

dunque le condizioni al contorno devono essere adattate affinché il team possa lavorare in modo agile. Soprattutto nei settori industriali più tradizionali, il successo dell'implementazione dipende fortemente dall'impegno del top management e dalla sua volontà di allineare di conseguenza le routine organizzative e le procedure alla nuova modalità di lavoro.

Le strutture gerarchiche devono essere adattate, diventando più piatte e generando una perdita di potere in alcuni livelli dirigenziali, soprattutto del middle management. Questo problema è denominato (13) "**prince problem**" e rappresenta proprio il fenomeno con cui il middle management perde responsabilità e quindi prestigio. È un problema noto, che ha un impatto non irrilevante, in quanto l'Agile per essere applicato con pieno successo deve trovare consenso da tutti gli attori in gioco.

Nelle strutture classiche, il project manager è responsabile del processo decisionale e impartisce gli ordini ai membri del team. Al contrario, in un team agile il ruolo del team leader, che è vestito dallo Scrum Master, è più di un abilitatore che supporta e responsabilizza le persone a fare tutto ciò che è necessario per raggiungere gli obiettivi, garantendo libertà e potere agli individui. Un tale cambiamento di ruolo deve essere accettato sia dai membri del team che dal team leader stesso. Facendo un ulteriore passo avanti, l'applicazione di metodi agili nella struttura aziendale nel suo complesso deve essere approvata, con conseguente (14) **impegno del top management** e (15) **del middle management**. Se manca questo impegno, lavorare in modo agile non può avere successo, poiché il team deve essere in grado di prendere decisioni e avere la libertà di agire per poter procedere con il proprio lavoro. Dunque, l'introduzione bottom-up dell'agilità all'interno di un'azienda può avere successo solo se il top management supporta questo approccio.

Inoltre, si dovrebbe evitare (16) **la gestione di più progetti** in parallelo da parte di un team, in modo tale che quest'ultimo possa concentrarsi solo sul progetto in corso e (17) anche i **modelli di processo interni** devono essere allineati di conseguenza. Se la cultura tradizionale dell'azienda non si adatta al cambiamento e l'agilità non è supportata dal top management, "la cultura mangia la strategia", lasciando terra bruciata (Atzberger & Paetzold, 2019). In generale, nelle organizzazioni deve verificarsi un adattamento complessivo all'approccio, così che si generi armonia interna.

- **Scaling**

Come già detto nella sezione precedente, dopo aver completato con successo l'implementazione dell'agilità in un progetto pilota, il passaggio successivo alla trasformazione agile è denominato scaling e consiste nel lancio dell'agilità in diversi progetti. A tal proposito, la prima sfida è (18) **trasferire la conoscenza** raccolta durante l'applicazione del primo progetto agile agli altri team di sviluppo. Infatti, i membri del team pilota si sono sottoposti a un processo di apprendimento, imparando a superare le sfide e sperimentando sulla propria pelle i vantaggi di una comunicazione frequente. Insieme a ciò, come già affermato per la cultura aziendale, (19) **la struttura della società deve essere allineata** e quindi devono seguire dei cambiamenti nell'organizzazione in modo da garantire la gestione di team multidisciplinari. Questo è necessario per aggirare (20) **il problema del "silo mentality "**, dove ogni dipartimento si concentra sui compiti di propria competenza piuttosto che sul progetto nella sua totalità.
La gestione di un progetto è resa più complessa anche a causa delle interfacce con gli altri reparti aziendali (come il marketing e gli acquisti), generando ulteriori sfide legate alle dipendenze. Inoltre, (21) **il cambiamento di mentalità all'interno dell'organizzazione nel suo complesso** e l'accettazione dell'introduzione dell'approccio agile è attualmente una delle maggiori sfide che vincolano lo scaling. In ultimo, deve essere intrapreso (22) un **adattamento ai valori specifici dell'azienda**, ma in molti casi le pratiche agili non si adattano sufficientemente alla struttura già esistente della società.

- **Distribuzione del team**

Al giorno d'oggi molte aziende sono suddivise in più filiali sparse in diversi paesi e quindi il tema della collaborazione tra team non co-localizzati ha acquistato sempre più importanza. In particolare, la sfida di come implementare un'efficace (23) **comunicazione tra team distribuiti** è una questione tuttora irrisolta, soprattutto per individui appartenenti a domini differenti e divisi per area geografica. A tal proposito, quando soggetti con diversi fusi orari devono collaborare (per esempio quando i membri del team americano devono collaborare con i membri del team europeo o asiatico) non hanno ben chiaro come interagire per eseguire gli stand-up quotidiani. In aggiunta, entrano in gioco le questioni su quali siano gli (24) **strumenti di comunicazione** da utilizzare per trasferire le informazioni in modo efficiente: sono scelte che oggigiorno non possono essere sottovalutate, quando sempre più persone di

paesi diversi devono cooperare e le barriere linguistiche potrebbero ostacolare la collaborazione. Anche (25) le **differenze etiche e culturali** possono costituire una sfida, poiché la critica aperta e una cultura open-error non sono tollerate in egual misura da tutte le nazionalità.

Tuttavia, considerando quanto affermato da (Carbonell & Rodriguez, 2006), non è tanto importante come i team siano distribuiti (co-localizzati oppure no), bensì quanto siano efficienti i canali di comunicazione e di accesso alle informazioni utilizzati e le modalità d'interazione tra gli individui. Rimane comunque aperta la questione se l'uso di team virtuali sia in grado di fornire lo stesso risultato dei team co-localizzati nella gestione dei progetti agili.

Facendo sempre riferimento allo studio di (Schmidt, et al., 2019), le sfide sono state classificate in base all'impatto sulla trasferibilità degli approcci agili dallo sviluppo software a quello hardware, assegnando un punteggio maggiore a quelle che più ostacolano questa traduzione. I risultati sono riportati in Figura 11:

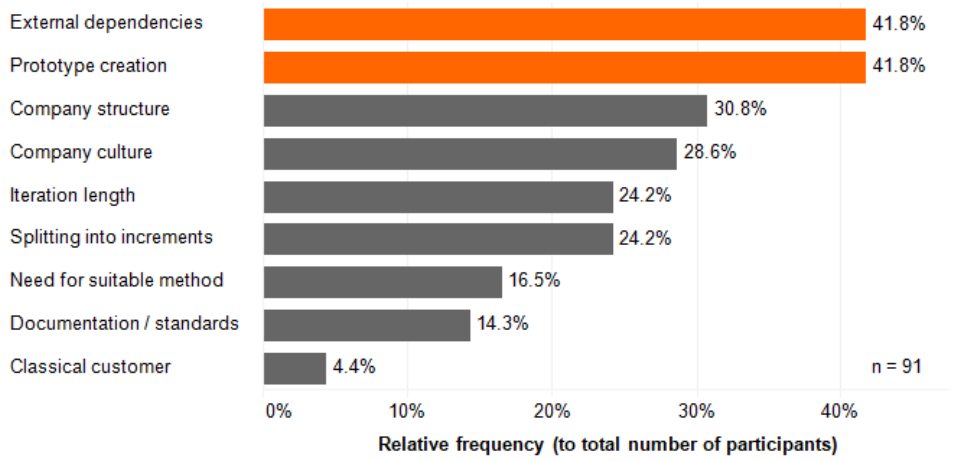

Figura 11: Fattori che ostacolano il trasferimento dei principi Agile dallo sviluppo software allo sviluppo hardware

I risultati sono stati sistemati in 9 gruppi, a loro volta collassati in 2 macro-gruppi definiti come ostacoli organizzativi e ostacoli tecnici. In particolare, risulta che:

- Le dipendenze esterne per lo sviluppo dei prodotti sono state classificate come l'ostacolo più alto da superare, insieme alla realizzazione di prototipi.
- La struttura aziendale, che comprende la disponibilità di risorse esperte e di team interdisciplinari, e la cultura aziendale, che riguarda l'aspetto della mentalità e le

interfacce con le strutture classiche già consolidate, sono il gruppo di ostacoli che si posizionano al secondo posto.

- Il 24% dei partecipanti allo studio ha sostenuto di non essere in grado di generare prototipi potenzialmente consegnabili all'interno di un'iterazione, il che significa che per il contesto hardware, il concetto di sviluppo iterativo a breve termine deve essere completamente riconsiderato.

Dunque, **gli aspetti che maggiormente ostacolano il trasferimento dei principi agili dallo sviluppo software a quello hardware sono correlati alle caratteristiche fisiche dei prodotti,** rendendo necessario un adattamento delle pratiche. Tuttavia, è importante sottolineare che l'Agile si basa su determinati principi e la loro natura non può essere rivoluzionata: prendere delle scorciatoie, attuare degli adattamenti estremi non può che ostacolare ulteriormente la sua implementazione, impedendo di ottenere i benefici sperati. Dunque, per quanto gli adattamenti siano necessari, non devono essere estremizzati.

Nonostante la trasferibilità del Manifesto al dominio hardware stia acquisendo una più ampia comprensione, i vincoli specifici dei prodotti fisici di cui sopra non sono stati ancora affrontati in un metodo completo e progettato nel dettaglio, motivo per cui è necessario stabilire delle linee guida su come affrontare le sfide associate. Inoltre, la maggior parte degli studi si è focalizzata sull'analisi di prodotti meccatronici, la cui complessità di sviluppo è incrementata dalla coesistenza di più contesti in parallelo, cioè di elementi software, elettronici e hardware. La Figura 12 è un tentativo di spiegare questa difficoltà, suddividendo l'ambiente di sviluppo lungo tre dimensioni (Atzberger & Paetzold, 2019).

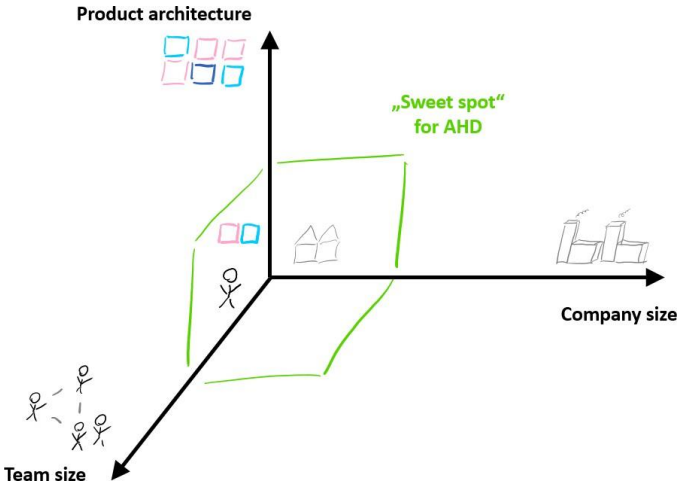

Figura 12: Schema del contesto di sviluppo hardware

In questo grafico, l'ordinata rappresenta la **complessità architetturale del prodotto**, che va da bassa (pochi componenti) a molto alta (diversi componenti con un alto livello di specializzazione). L'ascissa rappresenta invece la **dimensione dell'azienda**, che va da piccola impresa a grande impresa (multicorporate con diverse filiali in diversi paesi). In ultimo, sull'asse z è riportata la **dimensione del team** di progetto, che va da team di piccole dimensioni a più team / team di grandi dimensioni.

Esaminando più da vicino la prima dimensione, è necessario interpretare l'architettura di prodotto come l'interazione dei diversi domini (elettronica, meccanica e software) che consentono di ottenere un prodotto integrato altamente competitivo. A seconda dell'estensione del prodotto da progettare, la sua composizione può variare e arrivare anche alla coesistenza di centinaia di componenti (è il caso di prodotti altamente complessi, come automobili o aeroplani).

Le dimensioni dell'azienda invece caratterizzano l'ambiente in cui il prodotto è progettato: lavorare in una piccola azienda che sviluppa solo pochi articoli e mantiene un'atmosfera da "start-up" con molta libertà decisionale del team differisce in modo significativo da una grande impresa, che incorpora più filiali, con strutture aziendali rigide, vincoli operativi e quindi meno libertà.

Anche la dimensione del team può differire molto, a partire da una singola squadra co-localizzata fino a più squadre sparse in diversi paesi. Inoltre, la composizione del team è variabile e questo ha un impatto significativo sul progetto di sviluppo.

I quattro cluster CoP, mentalità, scaling e distribuzione del team possono essere sistemati all'interno del grafico in Figura 12. In particolare, i vincoli di fisicità fanno riferimento all'architettura di prodotto, mentre il mindset rientra sia nella dimensione dell'azienda che in quella del team. Lo scaling è correlato con le dimensioni dell'azienda, mentre la distribuzione del team è direttamente misurabile sull'asse delle ascisse. È facile intuire che in linea generale i "punti più favorevoli" per implementare uno sviluppo Agile rientrano all'interno dell'area verde. Tuttavia, i prodotti altamente sofisticati e complessi e che spesso richiedono una maggiore reattività nel corso della progettazione, sono prevalentemente sviluppati da aziende di grandi dimensioni che, a differenza di quelle più piccole, sono in grado di incorporare diversi domini. Le grandi aziende tendono a utilizzare team di grandi dimensioni con tutte le sfide che ne derivano, andando contro a ciò che propone il Manifesto, ovvero l'impiego di team piccoli e

co-localizzati. Le deviazioni dal Manifesto originale talvolta sono necessarie per garantire un allineamento ai metodi di sviluppo da sempre utilizzati e inoltre, la natura delle organizzazioni non può essere completamente stravolta.

2.3.5 Affrontare le sfide: gli adattamenti delle pratiche Agile

Nel corso del tempo le aziende hanno tentato di adattare i principi del Manifesto Agile al loro contesto di sviluppo prettamente hardware. Di seguito sono riportati alcuni esempi (Cooper & Sommer, 2018):

- **Trovare risorse**

La sfida principale per molte aziende è trovare le risorse umane necessarie a supportare un approccio Agile, in particolare delle risorse dedicate da inserire in team di progetto, in modo da attuare una completa implementazione del sistema. Sprint brevi, daily stand-up meetings e risultati rapidi sono molto più difficili da raggiungere quando i membri del team sono distribuiti contemporaneamente su diversi progetti o non sono sempre in contatto diretto tra di loro.

A partire da risultati di diversi casi studio di (Cooper & Sommer, 2018), la maggior parte delle aziende hanno trovato insostenibile l'assegnazione di risorse a un solo progetto per volta, per questo motivo hanno sperimentato vari compromessi, per esempio dedicare gli individui a un massimo di due progetti per volta, oppure limitare i carichi massimi di lavoro. Un'azienda in particolare ha descritto come gestisse la sfida del team dedicato, creando un unico gruppo di lavoro assegnato a una coorte di progetti contemporaneamente, tutti progetti simili e che richiedono simili abilità. Questo ha consentito ai membri del team di rimanere sempre in contatto, impedendo loro di essere deviati da altre attività ed eventualmente evitare duplicazioni degli sforzi.

- **Definire i deliverable degli Sprint**

In campo software, un team di sviluppo di solito può produrre qualcosa di funzionante entro la fine di ogni sprint, per esempio un codice eseguibile. Al contrario, la nozione di "sprint

fatto" non si applica così facilmente ai prodotti fisici: lo sviluppo di un nuovo motore, di un dispositivo medico o di una macchina non possono essere facilmente tradotti in piccoli incrementi. Anche quando è possibile creare un prototipo concreto, il tempo richiesto spesso è più lungo di un tipico sprint.

A questo punto è necessario un pesante adattamento: da un lato è possibile allungare la durata degli sprint rispetto alle tipiche 2 settimane impegnate nello sviluppo software. Tuttavia, quando si tratta di prodotti fisici, la defintion of done non deve necessariamente prendere forma in un prodotto funzionante, ma può consistere in un risultato tangibile del lavoro completato durante lo sprint. Per esempio, nelle fasi iniziali di ideazione, la definizione di "fatto" può essere un caso aziendale o il risultato di un esperimento preliminare o uno studio sulla Voice of the Customer. A partire dalla fase di sviluppo, invece, solitamente si produce qualcosa di fisico a cui il cliente può rispondere: disegni 3D generati al computer, prototipi virtuali, modelli grezzi, prototipi realizzati con stampati 3D e modelli funzionanti. Dunque, in questo contesto il risultato di uno sprint potrebbe essere qualcosa di fisico a cui il cliente può rispondere e che il management può vedere e toccare con mano (Cooper & Sommer, 2016). Rimane sempre valida la regola secondo cui il risultato di uno sprint deve incrementare il valore del progetto a ogni rilascio.

Facendo sempre riferimento allo studio di (Cooper & Sommer, 2016) sono riportati alcuni esempi della Definition of Done adottata da alcune aziende manifatturiere:

- Motori industriali Siemens: uno sprint è “fatto” quando il team di sviluppo decide che è terminato. Tuttavia, i risultati devono essere approvati dal Product Owner, il quale determina l'effettiva conclusione dello sprint.
- Produttore svedese di attrezzature edili: uno "sprint fatto" è il risultato del lavoro svolto, documentato come un rapporto su modello A4 (12 × 17 pollici), facile da leggere e facile da pubblicare, rivisto da un collega esperto e archiviato nel repository dei documenti.
- LEGO Education: uno sprint è “fatto" quando una lista di dieci elementi è completata da tutti i membri del team, dal controllo qualità e validata dal Product Owner, che dichiara che tutti i criteri di accettazione del prodotto sono rispettati e che tutti i task aperti sono stati conclusi.

- **Feedback da parte dei clienti**

Anche ottenere un rapido feedback dai clienti alla fine dello sprint può essere una sfida. Per il software, questo implica semplicemente l'invio della nuova funzionalità, di solito in

modalità elettronica, e la richiesta di feedback tramite lo stesso canale. Ma i prodotti hardware non possono essere inviati tramite e-mail e alcuni clienti non sono disposti a spostarsi così prontamente o non possono eseguire test così rapidi. In generale, le aziende hanno affrontato questa sfida in diversi modi (Cooper & Sommer, 2016): alcune hanno dedicato interi sprint alla ricerca della convalida con il cliente, altre hanno chiesto agli stakeholders di impegnarsi a prendere parte al processo di sviluppo prima ancora che il progetto fosse avviato, stipulando con loro degli accordi strategici. Il vantaggio per il cliente nell'investire il proprio tempo e i propri sforzi nel dare dei feedback risiede nel fatto che il prodotto finale è adattato esattamente alle sue esigenze ed è subito implementabile nei suoi sistemi.

2.3.6 Benefici attesi e reali: l'hype effect dell'Agile

Esiste un divario nella comprensione dello sviluppo Agile tra il mondo accademico e quello pratico (Schmidt, Weiss, & Paetzold, 2018b): l'accademia definisce lo sviluppo agile come un'abilità del team di progetto di reagire costantemente e rapidamente ai cambiamenti (previsti e imprevisti) all'interno di un ambiente dinamico e di sfruttarli come un vantaggio (Böhmer, Beckmann, & Lindemann, 2015). Dunque, l'interpretazione accademica non menziona tempi di esecuzione più brevi, né costi di sviluppo ridotti, mentre le aziende vogliono implementare approcci agili per sviluppare in modo più economico e veloce, cercando di ridurre il time-to-market. A tal proposito è da considerare che il successo riscontrato dalla filosofia nel mondo software potrebbe essere fonte di frustrazione per il contesto di sviluppo fisico.

Uno studio interessante (Schmidt T. , Weiss, Paetzold, Stefan, & Kristin, 2018a) ha portato alla luce tale discrepanza, indagando e confrontando gli effetti attesi (motivi dell'implementazione) con quelli effettivi (risultati reali) dell'adozione dell'Agile nei processi di sviluppo aziendali, in particolare nel campo dei prodotti fisici. Infatti, mentre gli effetti attesi fanno riferimento alle motivazioni per cui le organizzazioni vogliono diventare più agili nello sviluppo dei loro prodotti, gli effetti reali riflettono i vantaggi che realmente riescono a ottenere a seguito dell'implementazione. Il delta tra effetti attesi e reali aiuta a spiegare il cosiddetto **Hype Cycle di Gartner** mostrato in Figura 13.

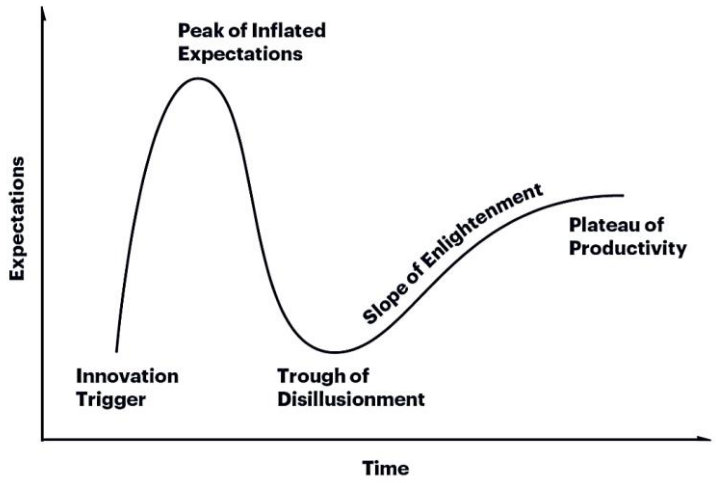

Figura 13: Hype Cycle di Gartner

La tecnologia ha delle aspettative sul mercato che si modificano pesantemente nel corso del tempo. In generale, l'Hype Cycle di Gartner spiega le cinque fasi fondamentali che ciclo di vita di una tecnologia attraversa (Gartner, s.d.):

- **Technology trigger** (innesco dell'innovazione): si avvia una nuova tecnologia potenzialmente dirompente. Lo sviluppo dei primi Proof of concept e l'attenzione dei media scatenano un grande effetto pubblicitario, nonostante non sussista ancora una prova della validità commerciale della tecnologia.
- **Peak of inflated expectations** (picco delle aspettative): l'effetto marketing iniziale nasce da una serie di prime storie di successo, generando così un largo consenso, spesso però accompagnato da molteplici casi di fallimento. Alcune imprese agiscono affacciandosi alla nuova tecnologia, ma molte non lo fanno.
- **Trough of disillusionment** (fossa della disillusione): l'interesse nella tecnologia svanisce quando la sperimentazione e l'implementazione non producono i risultati sperati. Le aspettative vengono disilluse e la maggior parte dei produttori entrano in crisi o falliscono. Gli investimenti non si interrompono solo per le imprese sopravvissute, che continuano a migliorare il prodotto per rialzare le aspettative e soddisfare le esigenze degli early adopters.
- **Slope of enlightenment** (salita dell'illuminazione): inizia a diffondersi la conoscenza dei benefici che la tecnologia può portare. Gli sviluppatori della tecnologia creano prodotti di seconda e terza generazione, andando così incontro al mercato e alle sue esigenze. Un numero crescente di imprese finanzia progetti pilota, mentre quelle conservatrici restano caute.
- **Plateau of productivity** (altopiano della produttività): l'adozione della tecnologia

inizia a prendere sempre più piede fino a diventare mainstream. A questo punto sono stabiliti in modo più dettagliato i criteri di valutazione dell'affidabilità del prodotto.

Dunque, riesaminando il processo seguito dalla tecnologia, all'inizio la nuova idea è divulgata tramite pubblicità e questo accresce le aspettative. Tuttavia, con il passare del tempo queste aspettative gonfiate si rivelano in parte non realistiche, cosicché sempre più persone abbandonano l'uso della tecnologia. La visibilità e le aspettative si riducono al minimo, arrivando al punto più basso della disillusione. Alcune persone continuano a credere nella potenziale tecnologico, quindi lo sviluppo non si interrompe e la tecnologia è continuamente migliorata e messa sul mercato da poche aziende, fino a quando il concept non diventa maturo, affidabile e ricco di dettagli. La visibilità aumenta fino a quando non si arriva al plateau della produttività.

Questo è il ciclo di vita di una nuova tecnologia quando viene sviluppata e lanciata sul mercato, ma può anche fare riferimento al ciclo di vita di un nuovo paradigma di sviluppo, in questo caso l'Agile.

Da un'analisi di Google Trends, l'Agile per lo sviluppo software è stato nel pieno della disillusione nel 2012: "I metodi agili hanno superato il picco delle aspettative, l'hype è finito" dicevano (Janes & Succi, 2012). Successivamente è diventato un approccio standard per i progetti di sviluppo software.

Per ciò che riguarda lo sviluppo Agile dei prodotti fisici, (Schmidt, Weiss, & Paetzold, 2018b) hanno condotto uno studio che indaga scientificamente sui benefici reali e attesi, i cui risultati sono riportati in Figura 14. Da un confronto tra i punti individuati sulle due curve, risulta che aspetti quali la trasparenza, l'apprendimento, la creazione di conoscenza e l'allineamento del prodotto alle strategie aziendali sono sottovalutati dalle aziende, nonostante lo sviluppo agile abbia un effetto maggiore su queste dimensioni. Al contrario, gli effetti sulla riduzione dei tempi di esecuzione dei progetti, sul rispetto delle scadenze, sulla riduzione dei costi di sviluppo e su una maggiore produttività sono tipicamente sovrastimati. Infatti, l'implementazione di uno sviluppo Agile fornisce un vantaggio inferiore per queste dimensioni rispetto a quanto previsto, soprattutto nei primi periodi, quando l'approccio non è ancora maturato e ben instaurato all'interno dell'azienda.

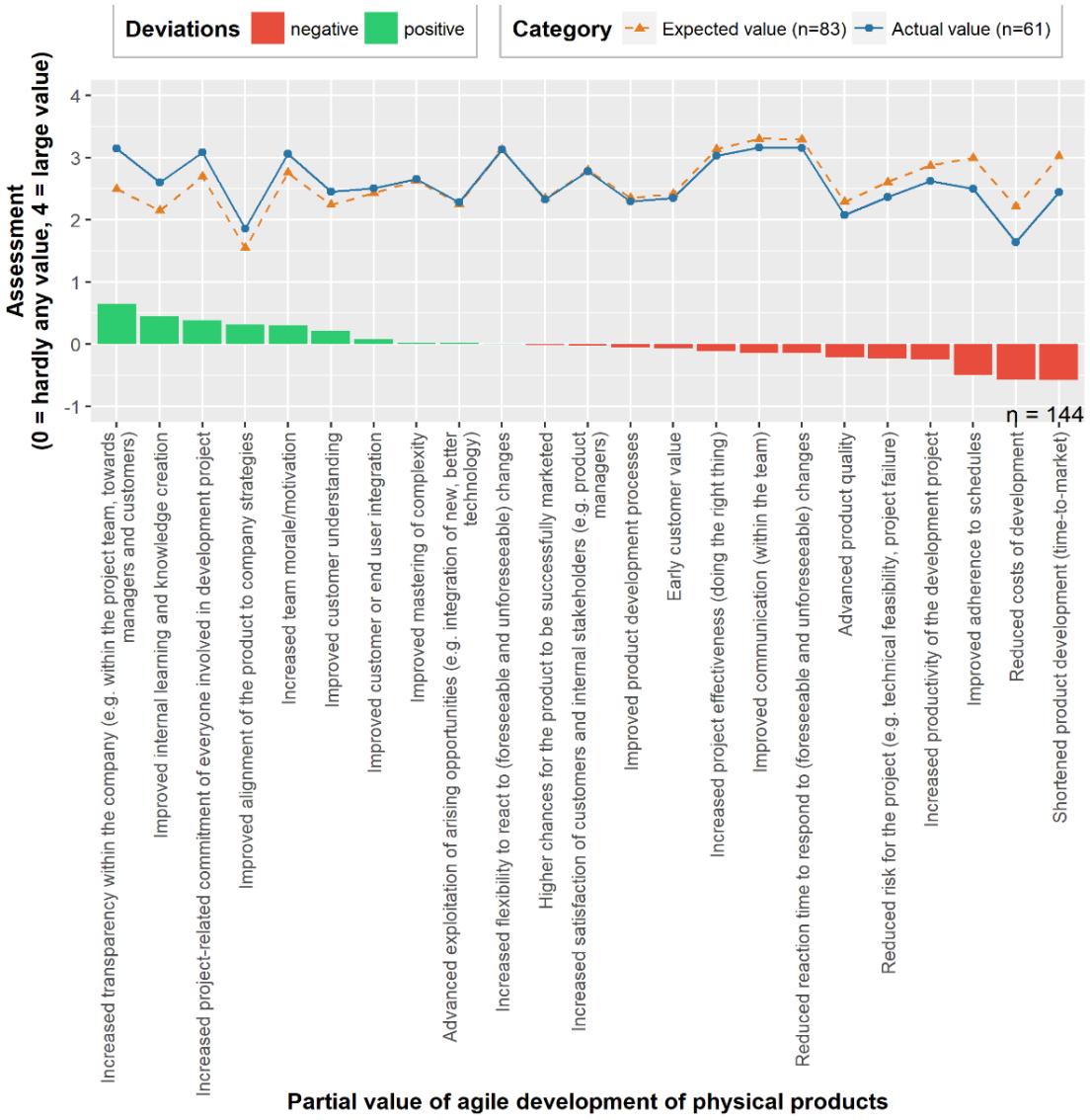

Figura 14: Confronto tra benefici reali e attesi nell'applicazione dell'Agile per lo sviluppo hardware

Analizzando invece i vantaggi che si possono raggiungere mediante l'applicazione dell'Agile, si registra una miglior comunicazione sia tra i membri del team sia con gli stakeholders, con conseguenze positive sull'impegno nel progetto e sul morale della quadra di lavoro. A questi benefici si aggiungono una maggiore reattività ai cambiamenti, una migliore trasparenza e una maggiore flessibilità. Sono aspetti soft, difficili da misurare e solitamente non monitorati nelle aziende.

Al contrario, il valore più basso si registra per la riduzione dei costi di sviluppo, un migliore allineamento del prodotto alle strategie aziendali e una migliore qualità di prodotto. Questi ultimi sono tra i più importanti indicatori chiave di performance (KPI) monitorati dalla maggior parte delle aziende e sono relativamente ben misurabili.

In sostanza, risulta che lo sviluppo Agile ha degli effetti positivi immediati e rilevanti sui fattori soft come la trasparenza, la comunicazione e l'impegno di tutte le parti interessate, mentre ha degli effetti decisamente meno rilevanti sul controllo degli hard KPI come tempo, costi e qualità. Quanto appena visto vale nel breve periodo, in cui si considerano gli effetti che saltano immediatamente all'occhio delle organizzazioni. Tuttavia, con il passare del tempo il miglioramento dei fattori soft si riflette su un controllo più solido degli hard KPI; quindi è possibile affermare che seguire il Manifesto Agile è un mezzo per migliorare direttamente gli aspetti soft che a loro volta contribuiscono al controllo dei KPI core aziendali (per esempio, ottenere una maggiore trasparenza interna all'organizzazione è un mezzo per raggiungere un miglioramento dei processi interni di apprendimento e di creazione di conoscenza, che a sua volta può essere un mezzo per raggiungere un miglioramento della soddisfazione degli stakeholder).

Dunque, l'applicazione dell'Agile porta diversi effetti positivi, alcuni riscontrabili fin da subito, altri verificabili con il passare del tempo. Poiché i principali benefici attesi riguardano il miglioramento dei KPI di controllo, questa evidenza è allarmante ed esiste una minaccia reale che le aziende abbiano delle aspettative troppo alte in merito ai risultati derivanti dall'implementazione dello sviluppo Agile (Schmidt, et al., 2019). Infatti, volendo far riferimento all'Hype Cycle di Gartner (Figura 15), lo sviluppo Agile dei prodotti fisici sembra essere vicino al picco delle aspettative e non ha ancora raggiunto la fase della disillusione. D'altro canto, per quanto riguarda lo sviluppo software, l'Agile ha già superato la disillusione e attualmente si trova sulla slope of enlightenment. Le previsioni affermano che lo sviluppo dei prodotti fisici potrebbe superare più velocemente l'intero ciclo hype rispetto a quanto si è registrato per lo sviluppo software poiché è possibile apprendere dalle esperienze passate (Schmidt, Weiss, & Paetzold, 2018b). Tuttavia, queste previsioni sono messe in discussione dalle sfide specifiche che il contesto tangibile deve affrontare, a cui ancora non è stata trovata una soluzione e il cui numero è destinato a crescere.

Pertanto, facendo riferimento alla curva dell'Hype Cycle di Gartner e l'attuale posizione dello sviluppo Agile dei prodotti fisici, la frustrazione è inevitabile, soprattutto per quanto riguarda gli aspetti di costi e tempi di realizzazione dei progetti.

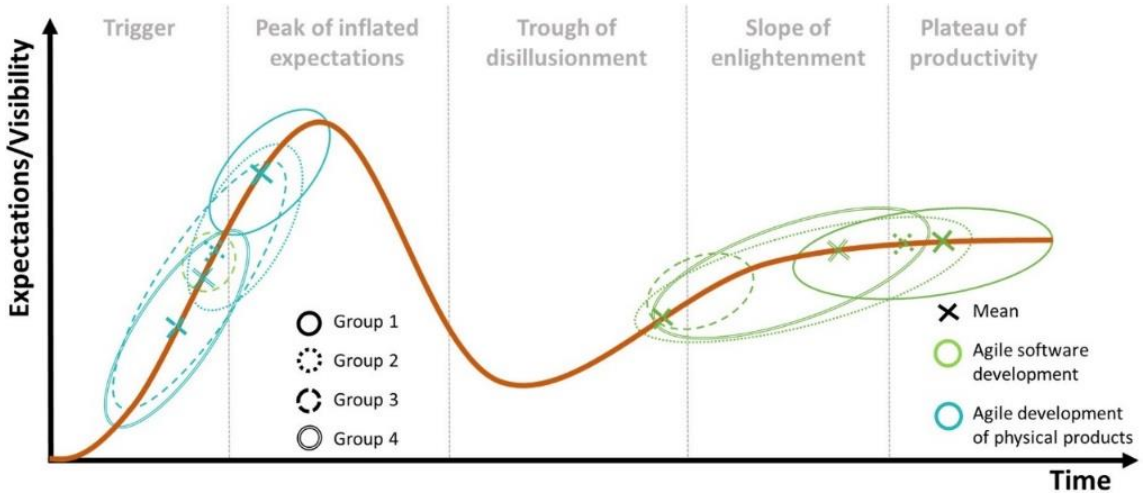

Figura 15: Gartner Hype Cycle per lo sviluppo di prodotti software e hardware

Per riassumere, se implementato correttamente, lo sviluppo Agile migliora fattori soft come la comunicazione all'interno del team e con gli stakeholders, il tempo di reazione ai cambiamenti, la flessibilità di sviluppo, la trasparenza e l'impegno del team. Questi aspetti sono un mezzo che contribuiscono a gestire i KPI di controllo, ovvero tempi, costi e qualità. Di conseguenza, è sconsigliato implementare l'Agile nei processi di sviluppo di prodotti fisici aziendali se l'accelerazione del time-to-market e la riduzione dei costi di progetto sono l'unico miglioramento che si intende raggiungere. L'Agile non rende di per sé i progetti di sviluppo più veloci, né più economici, ma li rende trasparenti e altamente reattivi ai cambiamenti.

3. AGILE - STAGE-GATE

Negli ultimi anni, l'approccio Agile ha contaminato molti settori, spingendosi fino alle industrie in cui i progetti di sviluppo prodotto sono da sempre gestiti con metodi più tradizionali. Tra queste imprese si possono trovare per esempio quella aeronautica e manifatturiera. Tuttavia, applicare le nuove pratiche in contesti così radicati nelle tradizioni può rivelarsi molto complesso e controproducente. Infatti, le grandi industrie con le loro organizzazioni stabili e ben strutturare, differiscono molto dalle imprese di sviluppo software, che sono più piccole e flessibili. La loro natura stabile e multi-progetto ha reso molto complicato abbracciare completamente il nuovo framework e applicare un approccio Agile puro al processo di sviluppo, il che ha portato alla costruzione di soluzioni intermedie.

A tal proposito è stato definito un modello ibrido, comunemente chiamato Agile-Stage-Gate, che prevede l'integrazione dei nuovi approcci agili all'interno dei tradizionali sistemi di gating (Cooper & Sommer, 2016). In particolare, Agile-Stage-Gate si basa sull'applicazione di una logica tradizionale a livello strategico e l'integrazione dello sviluppo iterativo all'interno delle fasi specifiche. Dunque, il progetto mantiene un approccio tradizionale a livello macro, ma è gestito in maniera più flessibile nel dettaglio, sfruttando alcuni vantaggi dello sviluppo agile, senza però abbandonare la stabilità che caratterizza i metodi tradizionali (Ciric, Lalic, Gracanin, Placic, & Zivlak, 2018). In questo modo l'Agile del mondo IT può essere integrato direttamente in Stage-Gate per lo sviluppo di prodotti fisici, senza rivoluzionare completamente il modus operandi aziendale.

La coesistenza tra i due approcci non sempre si è rivelata semplice, perché l'Agile si porta dietro le principali problematiche discusse nel capitolo precedente (*2.3.4 Sfide nell'applicazione*). Tuttavia, l'ibridazione del sistema iterativo in quello tradizionale consente di mantenere in parallelo il focus su due orizzonti temporali distinti. Infatti, l'Agile potrebbe risultare troppo focalizzato sui risultati a breve termine: gli sprint sono eccellenti per mappare ciò che il team di sviluppo dovrebbe concludere ogni settimana, ma possono rendere complicata la visione sulla strategia di lungo periodo. In più, l'Agile lavora con un'idea di continuità, senza interrogarsi sull'opzione di proseguire con il progetto oppure interromperlo. Al contrario, il metodo Stage-Gate consente di mantenere una visione di lungo periodo e fornisce un orientamento strategico al processo di sviluppo prodotto, in quanto i gate lasciano spazio alla direzione per prendere le decisioni più importanti sulla

continuità dei progetti in corso (Cooper & Sommer, 2018). Dunque, si può affermare che un modello ibrido che integra elementi dell'Agile insieme a elementi tipici del modello di gating può aiutare le aziende a capitalizzare i punti di forza di entrambi.

3.1 Il modello Stage-Gate

Il modello Stage-Gate (Cooper R. , 1990) è tra le metodologie più diffuse nell'ambito dello sviluppo di prodotti tangibili e accompagna un nuovo prodotto dalle fasi iniziali di ideazione fino al lancio, consentendo inoltre di gestire l'intero portafoglio di progetti in maniera strategica e coerente tra di loro.

I progetti sono divisi in una serie predefinita di fasi, detti Stage, che sono alternate a momenti di controllo, ovvero i Gate (Figura 16). I Gate sono dei veri e propri istanti di decisione Go/Kill del progetto, in cui si determina se quest'ultimo passerà alla fase successiva oppure se verrà interrotto. In caso di continuazione, verrà assegnato un budget in modo che il team possa lavorare alla fase seguente, ma non oltre, poiché al gate successivo verrà presa un'ulteriore decisione sulla base delle nuove informazioni.

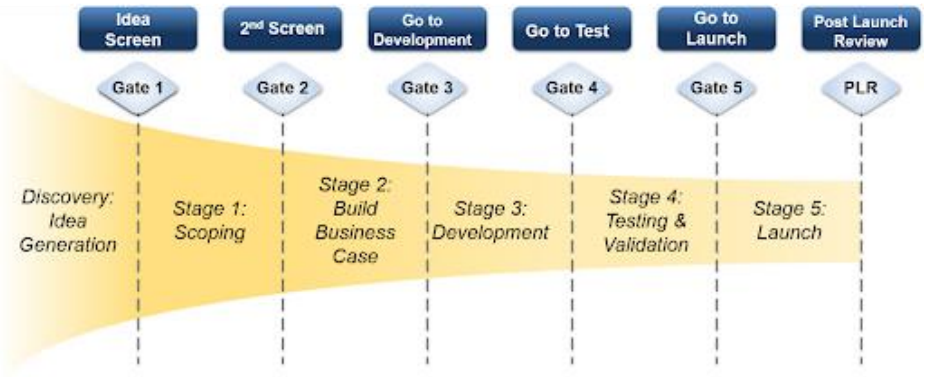

Figura 16: Fasi del modello Stage-Gate

All'interno di ogni fase devono essere chiari gli obiettivi da perseguire, le attività da svolgere e il risultato che deve essere raggiunto. Invece i Gate funzionano da filtro per i progetti più promettenti, consentendo di distribuire le risorse in modo efficiente. Ogni gate è caratterizzato da una serie di deliverable, una serie di criteri di uscita e un output. In particolare, gli input sono i risultati che il responsabile del progetto deve portare al gate, mentre i criteri sono gli elementi su cui verrà valutato il progetto e possono consistere in

requisiti essenziali definiti “must meet”, oppure delle caratteristiche desiderabili definite “should meet”. In ultimo, gli output sono le decisioni che vengono prese al gate e possono esplicarsi in Go, Kill, Hold, Recycle con l'approvazione di un piano d'azione per la fase successiva.

Le decisioni ai gate seguono un modello di scelta di investimento: impegnano le risorse per la fase successiva, in modo che queste siano indirizzate ai progetti migliori man mano che il loro potenziale emerge. Stage-Gate fornisce quindi una guida per quali progetti portare avanti e cosa realizzare all'interno di ogni progetto (Cooper R. , 2016). Seguendo questa logica a ogni gate diminuisce il numero di progetti su cui l’organizzazione sceglie di investire e, esattamente come risulta da Figura 16, l’andamento del processo di sviluppo prodotto assume una forma a “funnel” (a imbuto), tipico dei processi innovativi.

3.2 Agile-Stage-Gate: stato dell’arte

Gli studi empirici sulle prestazioni dei modelli ibridi applicati in domini hardware non sono numerosi, ma esistono. (Sommer, Hedegaard, Dukovska-Popovska, & Steger-Jensen, 2015) hanno condotto alcuni casi studio approfonditi all'interno di sette aziende manifatturiere, coinvolgendo produttori di turbine eoliche, valvole e sensori, insulina, giocattoli di plastica, amplificatori musicali, finestre e cavi di alimentazione, al fine di esplorare quanto l’Agile-Stage-Gate possa migliorare le prestazioni dello sviluppo prodotto. Le società hanno dimostrato che le aziende manifatturiere possono ottenere notevoli vantaggi dall’implementazioni di questi metodi.

Anche il caso studio di (Conforto & Amaral, 2015) riporta un'analisi empirica di un framework ibrido di gestione, che combina l’Agile con il modello Stage-Gate ed è implementato in un progetto guidato dalla tecnologia. I risultati evidenziano un impatto positivo sulle prestazioni di sviluppo e suggeriscono che la combinazione dei due approcci, che bilanciano stabilità e flessibilità, è una soluzione potenzialmente applicabile per la gestione di progetti particolarmente innovativi in aziende altamente tecnologiche.

Secondo (Lehnen, Schmidt , & Herstatt, 2016), adottando Agile-Stage-Gate per i progetti di sviluppo di nuovi prodotti, le aziende sono in grado di superare le sfide tipiche dell’Agile: il modello ibrido infatti fornisce un framework di gestione che da un lato è sufficientemente

rigido da essere riproducibile, dall'altro è abbastanza flessibile da rispondere in modo rapido ed efficace ai cambiamenti e agli imprevisti che sorgono durante il processo di sviluppo.

I due autori (Cooper & Sommer, 2016) con il loro articolo hanno mostrato i risultati raggiunti dai primi utenti che hanno implementato questo nuovo sistema all'interno di diverse industrie, che vanno dall'alimentare fino alle attrezzature pesanti. Due anni dopo, gli stessi autori con il loro studio (Cooper & Sommer, 2018) hanno aggiornato i risultati conducendo delle indagini in sei importanti aziende, segnalando miglioramenti significativi sia nel time to market che nella produttività dello sviluppo implementando l'Agile-Stage-Gate. Per concludere, hanno anche fornito delle raccomandazioni utili per attuare con successo un sistema ibrido di sviluppo prodotto.

3.3 Funzionamento dell'Agile-Stage-Gate nel processo di sviluppo

Un modello ibrido che integra gli elementi di Agile all'interno di Stage-Gate può aiutare le aziende a inglobare i punti di forza di entrambi. Come già detto in precedenza, Agile – Stage-Gate incorpora il modo di lavorare Agile all'interno delle fasi di gating (Figura 17), sostituendo gli strumenti e gli approcci tradizionali di gestione dei progetti (come i diagrammi di Gantt e la pianificazione del percorso critico) con strumenti e processi in linea con la filosofia Agile.

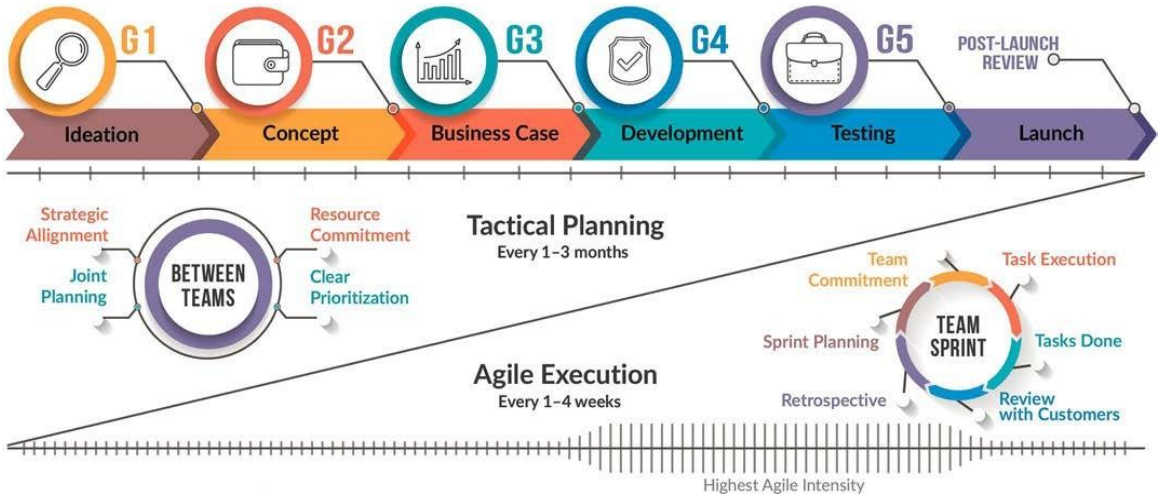

Figura 17: Modello ibrido Agile–Stage-Gate

Il modello è stato descritto nel dettaglio da (Cooper & Sommer, 2018): ogni fase è composta da una serie di sprint, la cui durata può variare da due a quattro settimane (Figura 17, a destra). Ogni sprint è pianificato in tempo reale, al volo, generando un processo altamente reattivo e adattivo. Al termine di ogni sprint il team consegna una release testabile: un risultato, un prototipo o un altro modello fisico, qualunque cosa che possa essere dimostrato alle parti interessate (inclusi i clienti) per la convalida e per identificare eventuali modifiche. In parallelo, le organizzazioni conducono una pianificazione tattica che coinvolge un periodo di tempo più lungo e si aggiorna con una cadenza regolare, che può variare da mensile a trimestrale. Questa macro-pianificazione è realizzata e monitorata in tutti i team di sviluppo insieme al management aziendale, in modo da creare un piano d'azione comune, assegnare priorità alle attività e allocare le risorse limitate per il periodo successivo (Figura 17, a sinistra).

La Voice of the Customer è un driver dinamico durante tutta la durata del progetto: il rilascio rapido, iterativo e incrementale di concetti, progetti e prototipi fornisce un feedback continuo da parte dei clienti, che è integrato nello sprint successivo per avvicinare il prodotto ai fabbisogni espressi. I clienti a loro volta partecipano al processo in modo da definire più chiaramente le loro esigenze e avere il beneficio di ottenere un prodotto che rispecchi le loro aspettative.

Il team di sviluppo è idealmente dedicato a un progetto e co-localizzato nella stessa stanza, svolgendo le riunioni periodiche previste da Scrum, come lo Scum Meeting o il Daily Stand-up Meeting, per facilitare la comunicazione e la produttività. In particolare, con la Sprint Retrospective il team è in grado di determinare se le attività assegnate sono state completate, se lo sprint è effettivamente "done" e come può migliorare il proprio metodo di lavoro negli sprint successivi.

Gli Stage e i Gate rimangono una parte importante di questo modello: i gate sono degli instanti fondamentali di decisione Go / Kill dei progetti, in cui sono eliminati quelli più deboli e poco profittevoli, e consentono al top management di rivedere i progetti in corso. Gli Stage forniscono una panoramica di alto livello delle fasi principali e dei risultati attesi al termine di ognuna di esse. Tuttavia, rispetto al modello tradizionale, i risultati finali specifici da presentare a ogni gate sono più snelli, meno granulari, più flessibili e soprattutto sono più tangibili, perché possono comprendere disegni o prototipi invece che report astratti o presentazioni di slide.

Nonostante le prove siano limitate, i primi risultati dal mondo manifatturiero suggeriscono che l'Agile può essere combinato con successo insieme ai tradizionali modelli Stage-Gate per lo sviluppo di prodotti fisici. Alcuni dei vantaggi di questo nuovo modello ibrido includono (Cooper & Sommer, 2016) (Cooper & Sommer, 2018):

- L'integrazione della Voice of the Customer in modo tempestivo, frequente ed economico, così da ottenere un prodotto con il giusto fit di requisiti;
- Una risposta rapida alle mutevoli esigenze dei clienti, aspetto fondamentale quando si affrontano mercati fluidi che cambiano rapidamente;
- La capacità di affrontare l'incertezza e l'ambiguità, spesso caratteristiche di uno sviluppo più innovativo;
- La riduzione del time to market;
- Il miglioramento della comunicazione, della motivazione e del morale all'interno del team di sviluppo, riducendo il cycle time e aumentando la produttività.

Tuttavia, insieme a questi vantaggi sono state sollevate anche alcune sfide nell'implementazione del metodo, che sostanzialmente coincidono con quelle generate dall'approccio Agile. Tra queste rientrano (Sommer, Hedegaard, Dukovska-Popovska, & Steger-Jensen, 2015):

- La difficoltà nell'acquisizione di risorse specializzate che consentano di comporre dei team multidisciplinari;
- Il mantenimento di un collegamento tra i team di progetto e il resto dell'organizzazione;
- L'affronto delle discrepanze tra i requisiti Agile e i sistemi di ricompensa aziendali, che si basano ancora su risultati raggiunti con i metodi più tradizionali di project management;
- Il contenimento di un'eccessiva burocrazia.

3.3.1 Fasi di implementazione dell'Agile-Stage-Gate

In generale, il processo inizia con una piccola task force composta da persone provenienti dalle diverse aree funzionali e coinvolte nel processo di sviluppo (aree tecniche, operative, marketing e vendite), in cui può essere presente anche un esperto esterno o un facilitatore. Mentre la task force si aggiorna rapidamente sulle pratiche Agile, analizza anche il processo

Stage-Gate già esistente in azienda. In questo modo si può valutare lo stato del sistema attuale e spesso il primo compito del team di lavoro è proprio quello di renderlo più snello (Fiore, 2005).

Una volta che il sistema Stage-Gate è stato snellito, la task force mappa il nuovo modello ibrido determinando dove l'Agile potrà funzionare, specificando per quali fasi e quali progetti (Cooper & Sommer, 2018). La task force deve anche determinare come affrontare le sfide che si presenteranno, rimanendo comunque in linea con la filosofia Agile e quindi evitando una pianificazione eccessiva.

Agile-Stage-Gate è un metodo ancora poco maturo: le aziende stanno cercando di capire come applicarlo e non esiste un sistema univoco e corretto per procedere. Questo è un nuovo modo di lavorare che coinvolge sia dirigenti che team di progetto. Durante l'implementazione, i membri del team affrontano una serie di periodi molto diversi tra loro, partendo da un "adattamento" iniziale, per poi passare a un periodo di frustrazione mentre gli individui scalano quella che può essere definita una curva di apprendimento molto ripida.

È importante sottolineare che tutte le aziende che hanno implementato con successo il metodo ibrido avevano alle loro spalle un sistema Stage-Gate molto ben sviluppato ed efficiente. Infatti, l'Agile non è una soluzione per un sistema di gating mal progettato e mal implementato, ma piuttosto supporta un sistema già funzionante, fornendo una risposta più rapida al cambiamento, una maggiore visibilità e una maggiore flessibilità.

Il sistema ibrido, così come l'approccio Agile, è solitamente utilizzato in una piccola percentuale dei progetti di sviluppo, generalmente per quelli più rischiosi, più ambigui e incerti. Infatti, dallo studio di (Cooper & Sommer, 2018) la maggior parte delle aziende ha stimato di utilizzare il sistema ibrido in circa il 20% dei progetti. Tuttavia, questa non è una regola, in quanto il metodo può essere impiegato per lo sviluppo di qualunque tipologia di prodotto, a maggior ragione del fatto che negli ultimi anni le disruption sono sempre più frequenti e arrivano da più fronti.

Sebbene Agile e Stage-Gate possano funzionare bene insieme bilanciando ciascuno i punti di forza e di debolezza dell'altro, sono stati individuati dei punti di scontro. Le aziende devono trovare delle soluzioni a questi conflitti che diventano evidenti quando si tratta da un lato di bilanciare la fluidità a breve termine con la pianificazione a lungo termine, dall'altro di trovare un accordo tra la definizione concreta di prodotto (prevista da Stage-Gate) con quella dinamica e in continua evoluzione (prevista dall'Agile).

3.3.2 La pianificazione nell'Agile-Stage-Gate

Uno dei principi fondamentali del Manifesto Agile è "massimizzare la quantità di lavoro non svolto" (Beck, et al., 2001) ed è rispettato dall'assenza di piani a lungo termine: siccome i piani non saranno validi a lungo, il lavoro per realizzarli è sprecato.

A questo punto sorge una questione: se il piano del progetto è in continua evoluzione e ha un focus sul breve termine, come si possono stimare i tempi, i costi di sviluppo e tutti i dati necessari per supportare un business case e ottenere l'approvazione del progetto? Il dilemma può essere risolto facendo ricorso a un livello più alto della definizione di prodotto, anche se alquanto provvisorio, in modo da fornire una definizione sufficiente per la pianificazione e l'approvazione del progetto. Intatti, il metodo utilizza product backlog che non è bloccato all'inizio del progetto, ma varia in base al feedback dei clienti e ai risultati degli sprint, evolvendo nel tempo man mano che il progetto avanza (esattamente come per l'Agile puro). Pertanto, gli sviluppatori devono accettare l'ambiguità, lavorando inizialmente con una definizione di prodotto provvisoria, la quale diventerà sempre più chiara e definitiva con l'avanzare del processo di sviluppo.

Dunque, la pianificazione è simile a un piano di sviluppo prodotto tradizionale che va dal gate "Go" fino a quello di testing e lancio, ma deve essere necessariamente molto provvisoria e poco dettagliata.

Allo stesso tempo, il management deve imparare a convivere con una certa ambiguità ed essere pronto ad approvare i progetti quando i piani, i costi e i tempi sono solo stime che possono cambiare. In realtà non c'è nulla di nuovo in questo, i tempi e i costi dei progetti approvati non sono mai certi e definitivi. La differenza rispetto ai metodi più tradizionali è che in questo sistema la probabilità di cambiamento è riconosciuta e accettata (Ciric, Lalic, Gracanin, Placic, & Zivlak, 2018).

3.3.3 Il ruolo dei Gate

Con l'Agile c'è flessibilità su ciò che viene consegnato, ma non nella struttura decisionale Go / Kill dei progetti, che invece è inclusa nel sistema ibrido.

La principale differenza tra Agile-Stage-Gate e i sistemi tradizionali consiste nel riconoscere e affrontare la realtà che molti progetti, specialmente quelli più innovativi, all'inizio hanno alti livelli di ambiguità e incertezza (spesso è impossibile sapere tutto in anticipo). Pertanto, le incertezze sui deliverable di prodotto sono gestite accettando e supportando le modifiche, a condizione che queste non influiscano sul piano generale del progetto, sul budget e sull'attrattività finanziaria. Se lo fanno, il progetto può essere contrassegnato per una rivalutazione ed eventualmente "killato", ovvero interrotto.

Nonostante il product backlog sia molto dinamico, i Gate continuano a svolgere un ruolo fondamentale in quanto consentono al management di riesaminare periodicamente i progetti, eliminare quelli deboli, riallocare le risorse alle iniziative migliori e, cosa più importante, garantire che le risorse necessarie siano impegnate in modo che i progetti possano andare avanti. Dunque, i gate non sono solo un punto di controllo della qualità, ma sono anche una decisione di impegno in termini di risorse. In questo modo il team si assicura il personale, il tempo e i finanziamenti necessari per continuare il proprio lavoro (Cooper R. , 2013).
Il Gate consente inoltre al management di tenere traccia dello stato di avanzamento e delle prestazioni dei lavori, monitorando il progetto con un'ottica di lungo termine. Sebbene ogni sprint abbia il suo "piano", il suo periodo di tempo entro il quale deve essere rilasciato e un'elevata flessibilità, la linea temporale a lungo termine rimane relativamente stabile.

I progetti all'interno di questo nuovo sistema sono gestiti con linee temporali o diagrammi di Gantt completi di milestones che definiscono chiaramente l'avanzamento del progetto su un orizzonte temporale predeterminato di lungo termine. Certo, queste tempistiche possiedono un livello di dettaglio piuttosto elevato e molti aspetti possono cambiare nel tempo, ma il piano e la tempistica a lungo termine rimangono intatti.

3.3.4 Gestione dei Gate hardware e software

Un'altra domanda comune riguarda la gestione di gate hardware e software: dovrebbero esserci gate differenti per risultati diversi e quindi si dovrebbero mantenere separati i gate software da quelli hardware? La maggior parte degli utenti ritiene che avere più gate per diversi aspetti dello stesso progetto sia problematico, in quanto aggiungono complessità e ambiguità (Cooper & Sommer, 2016). Questo potrebbe complicare le cose soprattutto

quando, in un prodotto che coinvolge sia l'hardware che il software, gli stakeholder durante la riunione bocciano la release software del progetto, mentre poche settimane prima era stata data una valutazione positiva per la consegna lato hardware.
Questa complicazione che coinvolge in prima linea i prodotti meccatronici in cui le parti del progetto dipendono l'una dall'altra, ha fatto nascere la necessità di una struttura di gate comune. L'unico momento in cui può essere giustificata una struttura di gate separata è quando i due sviluppi non sono totalmente interdipendenti e quindi l'approvazione dei risultati lato hardware esula dai risultati lato software.

3.3.5 Affrontare lo scetticismo del management

Così come per Agile, anche l'implementazione di questo metodo ibrido implica una grande trasformazione culturale, quindi è di fondamentale importanza avere il consenso della leadership aziendale (Ciric, Lalic, Gracanin, Placic, & Zivlak, 2018).

La resistenza del management spesso nasce da idee sbagliate, in quanto molti sono convinti di dover abbandonare del tutto il vecchio sistema di gestione dei progetti, senza comprendere che in realtà implementare Agile non significa abbandonare Stage-Gate, ma i principi del Manifesto possono essere aggiunti al vecchio sistema di gating, dando vita a un modello ibrido che incorpora le caratteristiche di entrambi (Sommer, Hedegaard, Dukovska-Popovska, & Steger-Jensen, 2015).

Molte aziende sono riuscite a superare questa resistenza interna progettando i loro sistemi ibridi intorno ai manager (Cooper & Sommer, 2018): questi ultimi mantengono visione solo sui risultati ai gate point, ma non su tutte le attività di sviluppo. Procedendo in tal senso, l'attenzione dei leader rimane orientata ai risultati e alla misura delle prestazioni, senza essere influenzati dal metodo di gestione utilizzato nel processo in sé. Il monitoraggio di risultati concreti e misurabili può giocare un ruolo rilevante per convincere un gruppo dirigenziale scettico sull'efficacia del nuovo approccio ibrido.

4. REQUISITI FISICI DI PRODOTTO

I vincoli di fisicità analizzati nei capitoli precedenti (*2.3.4 Sfide nell'applicazione*) sono il principale ostacolo al trasferimento dei principi agili dal contesto software a quello hardware. Infatti, la creazione e il test di un deliverable hardware richiede molto più tempo rispetto alla scrittura di un software, così come il costo del cambiamento è differente. Il divario tra i due domini (analizzato nel paragrafo *2.3.1 Hardware e software*) non è trascurabile e, per le aziende che sviluppano prodotti tangibili, questa differenza è responsabile di molteplici sfide nell'implementazione dell'Agile, che riguardano soprattutto **i lunghi tempi necessari alla produzione di una release testabile** e **le dipendenze esterne**.

Sicuramente l'intensità con cui si presentano queste sfide sono ulteriormente condizionate dalla **complessità del prodotto** sviluppato: è possibile applicare le pratiche agili senza incontrare degli ostacoli rilevanti quando si tratta di un prodotto relativamente semplice e omogeneo, realizzato con materiali che non richiedono particolari lavorazioni e che possa essere facilmente replicato. Al contrario, per un prodotto più complesso ed eterogeneo (come possono essere i beni meccatronici), formato da diversi componenti che spesso non sono tutti realizzati dalla medesima azienda, subentrano le problematiche analizzate.

Volendo entrare più nel dettaglio, le limitazioni specifiche dell'hardware sono causate anche dalla **tipologia di connessioni fisiche che esistono tra i vari componenti**, dunque molti impedimenti nell'implementazione dello sviluppo agile sono correlati alla tipologia di architettura del prodotto (Schrof & Paetzold, 2019).

4.1 Architettura di prodotto

Secondo una definizione ampiamente accettata, l'architettura del prodotto è determinata dalle "relazioni tra gli elementi funzionali, la mappatura tra elementi funzionali e componenti fisici e le interfacce tra i componenti fisici" (Ulrich, 1995). Quindi, l'architettura è lo schema attraverso il quale le singole funzioni del prodotto sono allocate ai componenti fisici, descrivendo come questi sono organizzati all'interno del prodotto stesso per garantirne la funzionalità. In parole povere, si tratta dello schema che lega le funzioni del sistema ai componenti che le realizzano.

I collegamenti tra i vari componenti si chiamano **interfacce** e possono essere sia fisici che non fisici (per esempio possono consistere in una comunicazione wireless tra le parti).

Tenendo presente questa definizione, le architetture possono essere classificate in base alla mappatura tra gli elementi funzionali e le componenti fisiche (Cantamessa & Montagna, 2016):

- Le **architetture integrali** sono caratterizzate da interdipendenza funzionale tra i componenti, in cui ciascuna funzione è soddisfatta da più componenti e a loro volta i componenti incorporano più funzioni.
- Le **architetture modulari** sono caratterizzate da componenti funzionalmente indipendenti. Ogni componente ha il compito di implementare una singola funzione e ogni funzione è svolta da un singolo componente. Dunque, tra funzione e componente del sistema sussiste una relazione 1 a 1 e i componenti (o un sottoinsieme di componenti) prendono il nome di moduli. A loro volta, le architetture modulari possono essere:
  - **Slot** se le interconnessioni tra i componenti non sono basate su un'interfaccia standard, ma sono tutte diverse;
  - **Sectional** se l'interfaccia tra ogni coppia di componenti interconnessi è standardizzata e quindi tutti i componenti hanno la stessa interfaccia;
  - **Bus** se i componenti non sono collegati direttamente tra loro, ma attraverso un elemento di interfaccia comune (chiamato bus).

È possibile analizzare la struttura di un rimorchio per ricorrere a un esempio concreto delle due tipologie di architettura. In Figura 18 è rappresentato un rimorchio con un'architettura modulare; se si vuole rimuovere la funzionalità di protezione dalle condizioni atmosferiche (protect cargo from weather), sarà sufficiente esportare la parte superiore (box). Nel rimorchio in Figura 19 caratterizzato da un'architettura integrale, la stessa operazione non è possibile in quanto la funzione specifica è svolta da componenti diversi e la stessa copertura superiore svolge più funzioni.

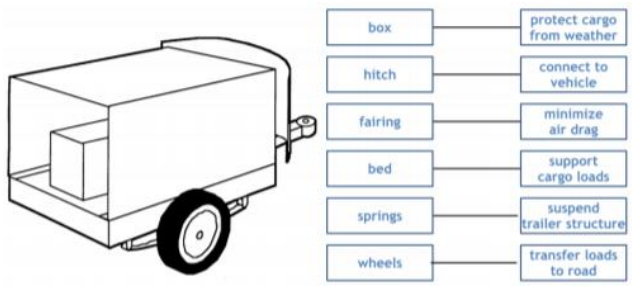

Figura 18: Architettura modulare

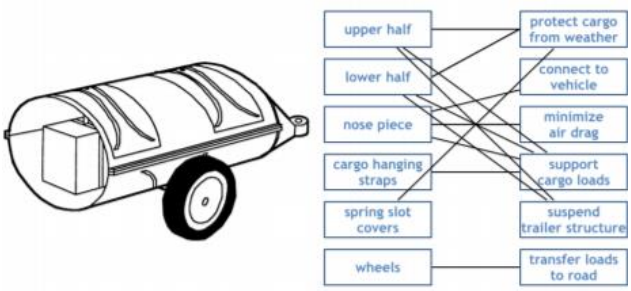

Figura 19: Architettura integrale

Generalmente non esiste un'architettura completamente modulare o integrale, ma le due architetture possono convivere all'interno di uno stesso prodotto, con la prevalenza di una rispetto all'altra. Nel settore automotive è possibile trovare numerosi esempi: per le auto con grandi volumi di produzione prevale l'architettura modulare, mentre per le auto prodotte in poche unità e che necessitano di prestazioni elevate è prevalente un'architettura integrale. Inoltre, la definizione di modularità o di integralità dipende pesantemente dal grado di dettaglio con cui si guarda al prodotto: considerando sempre l'esempio dell'automobile, il prodotto in sé è modulare, ma al suo interno contiene dei componenti altamente integrali, per esempio il motore.

4.1.1 Implicazioni delle scelte architetturali

Le scelte architetturali non impattano solo sul singolo prodotto, ma coinvolgono anche molteplici aspetti quali il dominant design, gli aspetti di integrazione verticale, la possibilità di apportare cambiamenti all'interno del ciclo di vita del prodotto e il suo processo di sviluppo. Di seguito sono riportate più nel dettaglio le principali implicazioni che seguono alla scelta di un'architettura piuttosto che un'altra (Cantamessa & Montagna, 2016):

- **Modifiche del prodotto**

Il prodotto può subire delle modifiche e delle revisioni, sia durante la progettazione che durante il suo ciclo di vita, quando si trova già nelle mani dell'utilizzatore finale. In generale, le modifiche possono essere apportate in maniera più semplice per i prodotti caratterizzati da architetture modulari, poiché i componenti possono essere adattati senza dover riprogettare l'intero prodotto.

Se si guarda al "trasferimento di componenti" tra versioni successive (termine che può essere espresso come la percentuale di componenti che un nuovo prodotto eredita dal suo predecessore), è intuitivo che un prodotto modulare renda relativamente semplice questo passaggio. Infatti, la modularità semplifica la riprogettazione di un prodotto già esistente in quanto consente di sostituire parte dei componenti senza dover partire da zero e stravolgere l'intero sistema. Al contrario, le interfacce che caratterizzano un'architettura integrale sono molto vincolanti e potrebbero richiedere la riprogettazione di tutti o della maggior parte dei componenti.

A livello strategico, l'adozione di architetture modulari consente alle aziende di sviluppare dei prodotti platform-based in cui, a partire da una piattaforma comune, possono essere generati rapidamente derivati di prodotto rispondenti a diverse specifiche di mercato. Grazie alla modularità, i componenti specifici dei derivati sono sviluppati senza dover riprogettare i componenti core e "in-platform". Se un'azienda diventa molto rapida nello sfruttare le architetture modulari e nel modificare la sua offerta di prodotti, può ridurre la sua dipendenza dalle ricerche di mercato e utilizzare l'effettiva accettazione da parte dei clienti come strumento guida. In casi estremi, le aziende possono scegliere di effettuare una "ricerca di mercato in tempo reale", modificando gradualmente i prodotti e i servizi sulla base dei feedback che progressivamente ricevono.

- **Varietà di prodotto**

Le architetture modulari e i prodotti piattaforma non solo consentono un cambio rapido tra versioni successive, ma permettono di offrire in modo simultaneo diverse varianti di prodotto. In questo modo le organizzazioni possono rivolgersi a più segmenti di mercato differenziati sia in senso orizzontale che verticale, con evidenti vantaggi per il loro potere di pricing. Inoltre, le architetture modulari consentono di offrire prodotti caratterizzati da varietà combinate: il prodotto core può essere facilmente personalizzato tramite l'aggiunta o la modifica di un certo numero di componenti.
In generale, le architetture integrali non si prestano bene a supportare la varietà. Tuttavia, potrebbero esserci delle eccezioni per prodotti relativamente semplici e realizzati in volumi elevati, che consentono un'estrema semplificazione della distinta base e quindi facilitando lo sviluppo di più varietà.

- **Standardizzazione**

A causa del disaccoppiamento tra i diversi elementi, le architetture modulari rendono più semplice l'utilizzo di componenti standard. La standardizzazione porta a evidenti vantaggi, tra cui le economie di scala e la riduzione della complessità. Tuttavia, questi vantaggi sono controbilanciati da costi aggiuntivi dovuti a un aumento del numero di interfacce tra i componenti. Al contrario, un'architettura integrale ha il vantaggio di minimizzare il costo standard e di ottimizzare la struttura del prodotto riducendone al minimo i componenti e le connessioni, con ovvi vantaggi sui costi di assemblaggio, sui costi dei materiali e sull'affidabilità.

- **Influenza sull'organizzazione e sulla catena di fornitura**

Le interdipendenze funzionali condizionano pesantemente le scelte strategiche delle aziende in merito ai componenti che intendono produrre internamente e quindi al loro posizionamento lungo la catena di fornitura. Infatti, le organizzazioni si specializzano nello sviluppo e nella produzione di componenti in base alle interdipendenze funzionali che sussistono tra di essi.

In generale, le architetture modulari portano a una riduzione del flusso di informazioni tra i progettisti. Se le relazioni funzionali tra i componenti sono deboli e standard, lo stesso componente può essere facilmente utilizzato in prodotti diversi, portando così a economie di scala. Inoltre, un progettista che sta sviluppando un componente può concentrarsi sulle sue prestazioni piuttosto che dedicare tempo alla sua integrazione nel prodotto, e questo favorisce anche la nascita di economie di apprendimento. Pertanto, le architetture modulari sono spesso associate a un basso grado di integrazione verticale delle organizzazioni, con una conseguente suddivisione tra le aziende produttrici del bene finale e i produttori di moduli e componenti. Ciò porta a una netta separazione dei ruoli e delle competenze sviluppate da ciascuna parte: i produttori del bene finale dovranno sviluppare capacità di assemblaggio e concentrarsi sulle questioni di integrazione, mentre i produttori dei componenti si concentreranno sui requisiti specifici.

Siccome "l'organizzazione rispecchia l'architettura", trattare un'architettura integrale implica essere fortemente integrati verticalmente: più i componenti sono non standard e interconnessi, più sarà difficile trovare dei fornitori disposti a realizzarli. Al contrario,

un'architettura modulare darà origine a scelte di integrazione verticali non spinte, in quanto ci saranno più fornitori disposti a realizzare quella tipologia di componenti meno specifici.

Dal momento che i vantaggi di un'architettura sono gli svantaggi dell'altra, non è possibile stabilire a priori quale sia migliore, ma la valutazione è strettamente legata al contesto. Per i prodotti più complessi, saranno prevalenti le peculiarità dell'architettura dominante.

4.2 Scelte architetturali nel contesto di sviluppo Agile e spunti di analisi

A questo punto è importante analizzare la questione in un'ottica Agile, cercando di capire se la prevalenza di un'architettura piuttosto che un'altra possa facilitare od ostacolare l'implementazione delle pratiche nel contesto di sviluppo.

Analizzando quanto visto finora, l'architettura modulare sembra avere le peculiarità utili a implementare efficacemente l'Agile: consente di suddividere più facilmente il prodotto in incrementi potenzialmente spendibili e permette di effettuare delle modifiche locali, senza compromettere tutti gli altri componenti. Tuttavia, per prodotti estremamente complessi, i tempi di realizzazione possono diventare molto lunghi e le interfacce tra i moduli devono essere altamente standardizzate onde evitare ulteriori complessità e riprogettazioni.

Un'architettura integrale potrebbe costituire un ostacolo in termini di suddivisione del prodotto, tuttavia consente di ottenere dei manufatti meno complessi, costituiti da un minor numero di componenti e con interfacce ridotte (seppur più articolate). Certo è che un prodotto altamente integrato, ma relativamente semplice, non comporterebbe alcun ostacolo per la suddivisione in incrementi potenzialmente consegnabili: sarebbero sufficienti pochi sprint per deliberarlo interamente.

Inoltre, le architetture modulari consentono di offrire dei prodotti facilmente personalizzabili, rispondendo così al principio Agile della massimizzazione della soddisfazione del cliente. Tuttavia, spesso a un'architettura modulare corrisponde un'organizzazione poco integrata verticalmente: a questo punto possono subentrare le problematiche legate alle dipendenze esterne, che vincolano la realizzazione di prototipi da mostrare.

Dunque, per riassumere, l'architettura modulare consente di suddividere il prodotto in moduli da rilasciare in modo iterativo e incrementale, di modificare delle parti senza necessariamente cambiare gli altri componenti, mentre l'architettura integrale consente di abbattere le dipendenze esterne e di ridurre al minimo i componenti e le connessioni che intercorrono tra di essi, rendendo i prodotti meno complessi.

A fronte di queste riflessioni nasce la domanda di ricerca su cui si basa il seguente lavoro di tesi:

Qual è la portabilità dell'Agile in funzione della modularità del prodotto? E quindi, quali sono le sfide per applicare l'approccio in un'architettura integrale?

A cui si aggiunge un'ulteriore analisi, in virtù dei numerosi riferimenti ai continui adattamenti che le aziende hardware sono costrette a mettere in pratica:

La digitalizzazione dei prodotti può facilitare la traduzione del Manifesto Agile dal contesto di sviluppo software a quello hardware?

L'analisi prende in esame diversi prodotti per approfondire le differenze tra le implicazioni delle due architetture nel contesto Agile e stabilire se la modularità è un requisito necessario per un'implementazione efficacie. Sicuramente non si tratta di una variabile booleana Agile/non Agile, ma si vuole comprendere quale grado di agilità consente di raggiungere una determinata architettura di prodotto.

In parallelo, l'analisi considera il grado di digitalizzazione del prodotto sotto l'ipotesi che, diminuendo il contenuto digitale ci si allontana dal contesto di sviluppo software e l'approccio Agile cambia natura, perdendo di flessibilità e lasciando spazio alla prevalenza di pratiche di sviluppo più tradizionali. Al contrario, per un prodotto altamente modulare e con un elevato contenuto digitale (come può essere il modello Tesla) i principi del Manifesto potrebbero essere applicati senza particolari ostacoli.

Il grado di contenuto digitale di un prodotto è un fattore che determina la vicinanza del prodotto stesso al mondo puramente software; insieme alla modularità, può essere una variabile rilevante per determinare la flessibilità dei processi di sviluppo e l'applicazione di un approccio Agile più o meno ibridato ai metodi tradizionali.

5. METODOLOGIA DI RICERCA

Dopo aver esaminato lo stato dell'arte sull'applicazione delle pratiche agili nello sviluppo di prodotti fisici, al fine di rispondere alla domanda di ricerca è stato necessario procedere con delle indagini dirette. Trattandosi di una fase di costruzione delle ipotesi e di approfondimento iniziale dell'argomento, si è scelto di procedere con un approccio di ricerca qualitativo, in quanto cerca di descrivere, decodificare e tradurre concetti e fenomeni piuttosto che registrare la frequenza con cui questi si verificano nella società (Basias & Yannis, 2018). Questa metodologia di ricerca è improntata sull'analisi delle esperienze, dei comportamenti e delle relazioni senza ricorrere all'uso di statistiche e di particolari elaborazioni di dati numerici (Hennink, Hutter, & Bailey, 2011).

Nel caso specifico, le indagini sono state condotte con il supporto di interviste semi-strutturate, in modo da ottenere delle risposte alle domande chiave che esplorano in profondità il tema prestabilito e avere sufficiente flessibilità per cogliere ulteriori aspetti non indagati. Infatti, l'intervista semi-strutturata si basa su una traccia principale in cui sono posti in risalto gli argomenti che dovranno essere toccati. Si tratta quindi di uno strumento a vari livelli di standardizzazione, con alcune domande chiuse e altre aperte. Non è possibile uscire dalla traccia, ma l'intervistatore può scegliere di toccare alcuni argomenti che nascono spontaneamente nel corso del colloquio nel caso in cui questi siano utili alla comprensione del topic indagato.

5.1 Il campione

Per gli studi che si avvalgono di tecniche qualitative è preferibile adottare un piano di campionamento non probabilistico (Sarantakos, 2005). Questo genere di studi (tra cui le interviste) hanno l'obiettivo di indagare sulla natura di determinati fenomeni da cui poter sviluppare nuove teorie, di conseguenza le unità di analisi sono scelte in modo strategico, selezionando chi ha sperimentato il fenomeno in prima persona e ha la conoscenza necessaria per spiegare l'oggetto indagato. Infatti, secondo (Rubin & Rubin, 2005) i soggetti intervistati devono avere l'esperienza (experience) e la conoscenza (knowledge).

Durante la stesura delle domande è iniziata una prima ricerca dei potenziali soggetti intervistabili. A tal proposito, LinedkIn è stato riconosciuto come lo strumento più idoneo per la ricerca di individui che lavorassero o avessero lavorato a contatto con le pratiche agili nell'ambito dello sviluppo prodotto.

Dunque, è iniziata una ricerca per parole chiave, tra cui "*Agile product development*", "*Scrum Master*", "*Product Owner*", "*Agile developer*", "*Agile Hardware*", "*Agile manifacturing*", filtrando prima di tutto per *Persone* e poi per *Settore*, escludendo a priori i settori Informatica, Servizi, Software, Servizi finanziari, Consulenza, internet e tutti quelli che non potevano garantire la produzione di beni fisici. Un terzo filtro è stato applicato alla *Località* selezionando l'Italia come paese di riferimento: le indagini si sono volutamente orientate alle aziende presenti sul territorio italiano per comprendere il livello di conoscenza delle pratiche agili.

Per raccogliere dei contatti in modo più mirato, sono stati cercati dei professionisti all'interno dei gruppi "*Agile Hardware Professionals*", "*New Product Development and Agile Innovation*", "*SCRUMstudy - #1 Group for Scrum and Agile; For Scrum Master, Product Owner and Project Team Members*" presenti sulla stessa piattaforma.

Le ricerche hanno consentito di trovare progettisti, Scrum Master, Product Owner, Head of Projects, responsabili R&D e altre figure professionali che avessero Agile methodologies, Agile Project Management, Scrum, Metodologie Agili, Agile & Waterfall methodologies tra le proprie competenze o qualche certificazione che confermasse le loro abilità in materia. Ogni soggetto è stato contattato personalmente, fissando giorno e orario per l'intervista in caso di riscontro positivo. Allo scopo di allargare ulteriormente la rete di intervistati, al termine di ogni colloquio è stato chiesto consiglio su eventuali colleghi da poter contattare. È stato intervistato anche un Agile Coach, figura professionale a cui molte aziende fanno affidamento per integrare al proprio interno le pratiche agili e adattarsi di conseguenza. Il parere di un esperto nel campo con un'ottica che andasse oltre i confini aziendali è stato d'aiuto per convalidare alcuni degli aspetti raccolti con le interviste e farne emergere di nuovi.

5.2 Collezione dei dati

Facendo ricorso alle interviste semi-strutturate, è stato possibile procedere con uno schema di domande preimpostato per rilevare le informazioni necessarie a rispondere alla domanda di ricerca, con la possibilità di renderlo flessibile per cogliere ulteriori aspetti, tra cui la componente emotiva dell'intervistato nei confronti delle pratiche agili. È importante sottolineare che la standardizzazione delle domande è dipesa anche dal livello di empatia raggiunto tra intervistatore e intervistato.

La traccia dell'intervista (*rif. Appendice A*) è stata organizzata per blocchi di domande, secondo la struttura seguente:

1) Spiegazione preliminare dell'argomento indagato e di ciò che si vuole ottenere dall'intervista;
2) Domande introduttive sull'azienda e sulla posizione lavorativa dell'intervistato;
3) Domanda principale per comprendere se l'intervistato utilizza/ha utilizzato delle pratiche agili;
4) Domande dirette sulla tipologia di prodotto sviluppato, per comprenderne la complessità, la tipologia di architettura e il grado di digitalizzazione;
5) Domande dirette sulla gestione dei progetti di sviluppo e sugli sprint;
6) Domande dirette sul rapporto con i clienti e i fornitori;
7) Domanda aperta sulle maggiori problematiche riscontrate nell'utilizzo dell'Agile e considerazioni finali.

Le domande non hanno seguito rigidamente questo schema, ma spesso il loro ordine è stato invertito in base alle risposte dell'intervistato. Sono comunque stati toccati tutti i punti sopra elencati e talvolta è stato necessario ripetere le domande in maniera differente per ottenere delle risposte più coincise.

Le interviste sono state condotte per via telefonica e telematica, in modo da avere un'interazione più efficiente. All'inizio di ogni incontro è stato chiesto al candidato il permesso per registrare la conversazione in modo da non interrompere periodicamente il flusso delle domande e poter trascrivere i risultati in un secondo momento, garantendo a ogni professionista il rispetto della privacy. Infatti, le informazioni sono state trattate in modo confidenziale, senza divulgare nomi degli intervistati e delle aziende. Le conversazioni hanno avuto delle durate differenti, a partire da 30 minuti fino a 1 ora di dialogo, concludendosi con saluti e ringraziamenti.

5.3 Metodologia di analisi dei dati

La definizione di un metodo preciso rappresenta un punto fondamentale per mettere in pratica una buona ricerca qualitativa; è importante stabilire quale approccio si intende seguire in relazione all'obiettivo della ricerca e alla tipologia di analisi che questa necessita.

A tal proposito, il seguente lavoro di tesi si è basato su un processo induttivo in grado di portare "rigore qualitativo" all'analisi mediante lo sviluppo di nuovi concetti. Questo approccio proposto da (Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee, 2013) dipende fortemente da una domanda di ricerca ben specificata, anche se piuttosto generale, e si appoggia a diversi fonti di dati (archivi, osservazione sul campo, documentazione, ecc.), dove il più importante di tutti è proprio l'intervista semi-strutturata, in quanto consente di ottenere resoconti da parte di coloro che sperimentano direttamente il fenomeno di interesse teorico.

Il metodo parte da un'analisi dei testi delle interviste e dall'individuazione di concetti[1] rilevanti ai fini della costruzione della teoria. Questi concetti generano dei codici appartenenti a categorie di 1° ordine, successivamente condensati in codici di categorie di 2° ordine. Si tratta di una vera e propria generazione di etichette a partire dal testo, che vengono raggruppate e collassate in codici con un livello maggiore di astrazione, in modo da avere sempre a portata di mano e in modo organizzato tutti gli aspetti rilevanti emersi durante le indagini (il collegamento tra le categorie avviene in un secondo momento, ovvero con l'analisi; qui ci si concentra sull'individuazione delle informazioni rilevanti e sulla loro classificazione).

Durante la categorizzazione di 2° ordine si è all'interno di un regno puramente teorico, in cui bisogna indagare e chiedersi se i concetti emersi potrebbero descrivere e spiegare i fenomeni, prestando particolare attenzione ai concetti nascenti, che sembrano non avere riferimenti teorici in letteratura. Le interviste dovrebbero essere integrate da più fonti di dati (per esempio archivi, osservazioni sul campo, documentazione sui media) per definire in modo più dettagliato il contesto in cui si inseriscono (Anteby, Lifshitz, & Eisenhardt, 2014).

Una volta ottenuta una serie completa di termini di primo e di secondo ordine, ci sono le basi per costruire una struttura che non solo consente di configurare i dati in un supporto visivo, ma fornisce anche una rappresentazione grafica del passaggio dai dati grezzi a termini:

[1] Concetto: si intende una nozione generale che cattura le qualità che descrivono o spiegano un fenomeno di interesse teorico.

questa è una componente chiave per dimostrare il rigore nella ricerca qualitativa (Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee, 2013).

L'applicazione del metodo si basa su due **ipotesi** di base:

- Le persone appartenenti alle loro realtà organizzative sono "agenti consapevoli", vale a dire che le persone nelle organizzazioni sanno quello che stanno cercando di fare e possono spiegare i loro pensieri, intenzioni e azioni. Quest'ipotesi è fondamentale, perché il ricercatore non deve imporre agli individui dei costrutti e delle teorie già esistenti per spiegare a priori la loro esperienza. Deve esserci lo sforzo di dare la giusta voce agli intervistati, soprattutto nelle prime fasi di raccolta e analisi dei dati, in modo da avere ricche opportunità di scoperta di nuovi concetti piuttosto che l'affermazione di concetti già esistenti (Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee, 2013).
- Una seconda ipotesi coinvolge i ricercatori stessi: anche loro sono persone sufficientemente informate, che conoscono il contesto indagato e sono in grado di individuare dei modelli nei dati, facendo emergere concetti e relazioni.

A fronte della scelta del metodo più idoneo da mettere in pratica, il contenuto delle interviste è stato analizzato seguendo questi passaggi:

1. Caratterizzazione di ogni singola intervista: trascrizione, lettura e comprensione dei contenuti;
2. Comparazione tra interviste, individuando somiglianze e diversità;
3. Individuazione dei concetti ed etichettatura del testo delle interviste;
4. Assegnazione delle etichette generate a categorie di 1° ordine e successiva raccolta in gruppi omogenei per ottenere categorie di 2° ordine;
5. Raggruppamento delle categorie di 2° ordine in dimensioni aggregate.

Entrando nel merito dell'applicazione della metodologia e partendo dalle trascrizioni delle interviste, proprio come suggerito dagli autori (Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee, 2013), i testi sono stati letti in modo intensivo così da generare un primo set di dati, termini e codici, ovvero concetti che si sono tradotti in categorie di 1° ordine: Tipologia di prodotto, Numero di componenti, % divisione del valore, Funzionalità componenti digitali, Architettura modulare, Architettura integrale, Presenza di gate, Agile nelle prime fasi di sviluppo, Agile in tutto il processo di sviluppo, Cerimonie Scrum, Backlog, Burndown chart, dashboard, Gantt, Definition of Done, Durata degli sprint, Prototipazione, Team multidisciplinari, Risorse dedicate, Feedback del cliente, Vincoli

esterni con fornitori, Vincoli esterni con enti certificatori, Vincoli interni, Mindset. Successivamente, le categorie di 1° ordine con un alto livello di dettaglio sono state confrontate e strutturate in categorie di 2° ordine: Caratteristiche del prodotto, Digitalizzazione, Architettura di prodotto, Flessibilità dello sviluppo, Strumenti agili, Sprint, Gestione dei team, Rapporto con i clienti, Difficoltà esterne, Difficoltà interne. In ultimo, a livello aggregato, sono state ottenute le tre dimensioni principali (1) Prodotto sviluppato (2) Pratiche di sviluppo e (3) Difficoltà, che rappresentano la base su cui sono state analizzate le interviste. Tutto il processo di individuazione di concetti ed etichettatura del testo è stato realizzato con il supporto del software ATLAS.ti. Di seguito in Tabella 2 è rappresentata la struttura di dati:

Tabella 2: Struttura dei dati

Concetti di 1° Ordine	Concetti di 2° Ordine	Dimensioni Aggregate
Tipologia di prodotto	Caratteristiche del prodotto	Prodotto sviluppato
Numero di componenti		
% divisione del valore	Digitalizzazione	
Funzionalità componenti digitali		
Architettura Integrale	Architettura di prodotto	
Architettura Modulare		
Presenza di gate	Flessibilità dello sviluppo	Pratiche di sviluppo
Agile nelle prime fasi di sviluppo		
Agile in tutto il processo di sviluppo		
Cerimonie Scrum	Strumenti agili	
Backlog		
Burndown chart, dashboard, Gantt		
Definition of Done	Sprint	
Durata degli sprint		
Prototipazione		
Team multidisciplinari	Gestione dei team	
Risorse dedicate		
Feedback del cliente	Rapporto con i clienti	
Vincoli esterni con fornitori	Difficoltà esterne	Difficoltà
Vincoli esterni con enti certificatori		
Vincoli interni	Difficoltà interne	
Mindset		

6. ANALISI DELLE INTERVISTE

Il presente capitolo ha lo scopo di illustrare i risultati ottenuti dalle indagini e di presentarne un'analisi per rispondere alla domanda di ricerca. I testi delle interviste sono stati rielaborati e strutturati nel primo paragrafo (*6.1 Raccolta delle interviste*), in modo da essere più facilmente comprensibili e confrontabili. Il secondo paragrafo (*6.2 Analisi dei risultati*) è dedicato all'analisi dei risultati, effettuata paragonando i concetti di 1° e di 2° ordine e cercando di indagare sull'esistenza dei collegamenti tra di essi. In ultimo, un'analisi semi-quantitativa prova a spiegare la rilevanza del grado di digitalizzazione del prodotto e del ruolo che gioca nella traduzione dei principi agili.

6.1 Raccolta delle interviste

Di seguito sono riportati tutti i dialoghi in una raccolta riassuntiva, in cui le interviste sono suddivise rispetto alle tre dimensioni aggregate ottenute seguendo il metodo proposto da (Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee, 2013) con lo scopo di evidenziare gli aspetti rilevanti e utili per l'analisi. Al fine di garantire l'anonimato, non sono stati riportati i nomi degli intervistati e delle rispettive aziende in cui questi ultimi lavorano o hanno lavorato.

6.1.1 Intervista 1

- **Azienda e intervistato**

L'azienda 1 si occupa della produzione di sistemi per il processamento delle banconote nel settore bancario. Sono dei dispositivi fisici, delle casseforti automatiche molto simili a un bancomat, destinati a uso interno della banca per le operazioni di back office di gestione del denaro. L'intervistato è stato Product Owner nell'R&D dell'azienda e ha lavorato con le pratiche agili per 3 anni. L'Agile è stato introdotto in azienda mutuandolo dallo sviluppo software.

- **Dettagli sul prodotto**

L'intervistato ha seguito lo sviluppo di riciclatori, ovvero dei dispositivi che consentono il deposito e il prelievo del denaro. Sono dei macchinari con un alto numero di componenti (intorno ai 2000 pezzi per macchina) e presentano un'architettura **modulare**.
La componentistica digitale consente di abilitare il deposito e il prelievo in automatico, effettuando anche un conteggio e una verifica delle banconote inserite. A fine giornata, l'operatore può visionare lo stato cassa a schermo.
Considerando il prodotto nel suo complesso e volendo suddividerlo fisicamente in componenti, il contributo funzionale di ogni macro-componente è: 80% per componenti meccanici ed elettronici, 20% software.

- **Dettagli sul processo di sviluppo**

Durante il processo di sviluppo, il team ha utilizzato il framework Scrum. Si è appoggiato a una dashboard virtuale per tenere traccia dei compiti da svolgere e all'inizio di ogni sprint erano pesati i task da rilasciare, assegnando loro un punteggio mediante la serie di Fibonacci (il sistema era simile a quello del burndown chart).
Il team si teneva una pianificazione parallela un po' tradizionale, un fil rouge che partiva dall'inizio e cercava di arrivare fino in fondo, elencando le tappe principali da cui si doveva arrivare. Dunque, c'era un macroflusso che prendeva mesi e dava un po' di visibilità ai vari gruppi su quello che si sarebbe dovuto fare nei mesi successivi, mentre tra ogni sprint c'era una pianificazione più di dettaglio.
Per lo sviluppo hardware sono stati definiti degli sprint un po' più lunghi, della durata di **30 giorni**. All'inizio di ogni sprint erano definiti gli obiettivi da raggiungere che, a seconda dell'oggetto da realizzare, potevano consistere in gradi diversi di progettazione: disegni del gruppo nel livello base con una prima ipotesi o schizzo, definizione dei particolari a partire dalla prima bozza, oppure, una volta definiti i particolari, la redazione di tutte le tavole e i disegni per procedere all'ordine dei materiali. Il team ha fatto anche uso di prototipazione rapida, tuttavia alcuni componenti non potevano essere prototipati facilmente.
Per quanto possibile il rilascio incrementale hardware era gestito in parallelo con quello software, anche se quest'ultimo utilizzava sprint di 2 settimane, dunque i rilasci non erano sempre sincroni. Se la componentistica hardware non era disponibile, il software veniva

rilasciato a tavolino, utilizzando solo delle componenti elettroniche per simulare la macchina.
Per ciò che riguarda il rapporto con i fornitori esterni, non sono stati riscontrati particolari problemi. Invece, la partecipazione del cliente finale agli sprint non è stata colta: l'ufficio marketing era il responsabile della definizione dei requisiti e delle specifiche, dunque non c'era collaborazione con un utilizzatore finale.

- **Problematiche maggiormente riscontrate**

Il problema maggiormente sentito è stato quello di progettare un oggetto fisico in tempi più stringenti e trovare un accordo sulla Definition of Done di uno sprint. Per questo motivo, spesso l'obiettivo era quello di arrivare alla fine dello sprint con qualcosa di consegnato e non con qualcosa di qualità, esasperando un po' l'aspetto quantitativo a discapito di quello qualitativo. Inoltre, l'Agile è stato introdotto in modo un po' forzato, senza dare il tempo al progetto di maturare internamente. Questa forzatura ha generato avversioni interne alle pratiche, che sicuramente hanno ostacolato una buona riuscita dell'implementazione dell'approccio.

6.1.2 Intervista 2

- **Azienda e intervistato**

Il secondo intervistato lavora in un'azienda di robotica, che si occupa di realizzare mani robotiche antropomorfe (non protesiche) per fare operazioni di pick and place in diversi ambiti. In particolare, l'intervistato copre il ruolo di Head of Project, quindi è responsabile dello sviluppo del prodotto sia della componentistica hardware che software. L'azienda ha da pochi mesi iniziato a lavorare con le pratiche agili.

- **Dettagli sul prodotto**

Il prodotto sviluppato prende forma in delle mani robotiche, applicate nel contesto pharma, nucleare, agroalimentare. È una mano che si presta bene anche per la logistica, ma l'unico campo in cui non è applicata è quello medico-chirurgico a causa delle numerose certificazioni richieste. Per il momento, il processo di sviluppo coinvolge una mano presente

sul mercato da 30 anni, quindi un prodotto già esistente che si cerca di migliorare continuamente. La mano è molto complessa, con un numero di componenti decisamente elevato; questo gli consente di fare dei movimenti molto simili a quelli di un arto umano.
Il prodotto è caratterizzato da un'architettura **modulare** e la componentistica digitale è necessaria per controllare i movimenti e dare i comandi per effettuare tutte le operazioni necessarie. Il contributo funzionale di ogni macro-componente è abbastanza equo, ovvero 40% hardware e 60% software: le capacità che deve avere la mano intrinsecamente devono essere elevate, tuttavia è il software il vero responsabile dei movimenti.

- **Dettagli sul processo di sviluppo**

Durante il processo di sviluppo, il team utilizza il framework Scrum, appoggiandosi su Jira[2] come piattaforma digitale per segnare i task dello sprint. I team partecipano alle cerimonie previste da Scrum, dunque organizzano riunioni giornaliere per monitorare lo stato di avanzamento dei lavori e comprendere quali problematiche si sono manifestate, e infine organizzano la pianificazione dello sprint successivo. In ogni team c'è un team leader che copre il ruolo di Scrum Master e organizza la riunione di retrospettiva alla fine dello sprint della durata di circa un'ora e mezza in cui si presentano le storie completate, mentre il Product Owner ha il compito di accettare la consegna.
Per quanto riguarda la pianificazione, trattandosi di un prodotto di ricerca in costante miglioramento è impossibile avere una delimitazione finale e quindi il team si appoggia a delle linee guida, degli obiettivi, dei requirements, che però cambiano molto nel divenire. Dunque, c'è un piano generale molto dinamico, rivisto ogni due settimane, e un piano di dettaglio per ogni sprint.
I team lavorano con gli story points solo quando la user story è assegnata allo sprint, cioè quando effettuano la pianificazione di dettaglio dello sprint. Gli story points in realtà sono usati principalmente come strumento per capire se gli ingegneri hanno compreso il lavoro previsto dal task e quindi per trovare un allineamento sull'obiettivo da raggiungere.
Per monitorare lo stato di avanzamento dei lavori, i team utilizzano il Gantt su Jira, ovvero un Gantt dinamico in cui inseriscono e tolgono i task all'interno di una timeline in modo da

[2] Jira è un software di gestione dei progetti che supporta qualsiasi strumento agile, come le bacheche Scrum o kanban. Sono presenti anche le board e i report tramite cui pianificare, monitorare e gestire tutti i progetti di sviluppo agile da un unico strumento. È ampiamente diffuso tra le aziende che lavorano in linea all'approccio Agile.

vedere i progressi e avere un'idea della deadline; grazie a questa panoramica generale, è possibile comprendere in linea di massima quando il progetto dovrebbe essere completato.
Gli sprint hanno una durata di **2 settimane** per ogni team di sviluppo. Sul progetto lavorano 3 team software e un team hardware che consegnano a settimane alterne così da non avere un sovraccarico di pianificazione: nella prima settimana iniziano a lavorare due team che finiranno dopo 2 settimane, nella seconda settimana invece iniziano gli altri due team, così da avere la consegna dello sprint a settimane alterne. Dunque, procedendo in questo modo si gestiscono due team alla volta invece che quattro.
Il team stabilisce la proof of acceptance di ogni task (il risultato di ogni sprint), che può essere un disegno, un test, un video, una foto, un documento, qualsiasi cosa che possa essere consegnabile e fornire valore al progetto. Trattandosi di progetti di sviluppo su un prodotto già esistente e con caratteristiche modulari, è semplice definire i confini degli item da rilasciare.
Per quanto riguarda la partecipazione dei clienti, questi sono responsabili della definizione dei requirements solo in caso di progetti commerciali su commessa. In caso contrario, non partecipano alla definizione dei requisiti, né forniscono feedback in fase di review.

- **Problematiche maggiormente riscontrate**

L'intervistato ha affermato che per lo sviluppo hardware non è facile avere sempre degli obiettivi ben dichiarati e soprattutto non è semplice realizzarli in 2 settimane, perché il delivery time di alcuni prodotti richiederebbe più tempo. Inoltre, sempre lato hardware, è difficile definire quando una storia è completata oppure no.

6.1.3 Intervista 3

- **Azienda e intervistato**

L'azienda si occupa della produzione di circuiti integrati, più nello specifico l'unità in cui lavora l'intervistato è specializzata nella realizzazione di sensori di posizione magnetica e ha iniziato a esplorare le pratiche agili da circa due anni. L'intervistato è Team Lead di un'equipe di 6 persone che si occupa della verifica del circuito nel processo di sviluppo prima che il prodotto vada in produzione**.**

- **Dettagli sul prodotto**

L'intervistato segue progetti di sviluppo di sensori di posizione magnetica applicati nelle automobili. Sono dei circuiti completi di diversi formati, ognuno incapsulato dentro a un package che il cliente integra nei suoi moduli. È un classico circuito integrato e il package può essere di diversi formati, cioè a 16 pin, 8 pin oppure 2 pin.
Il prodotto è fortemente **integrale**, su cui è inserito un processore custom sviluppato internamente. Il software aggiunge un'enorme funzionalità a un costo irrisorio e il contributo funzionale di ogni macro-componente può essere così suddiviso: 90% software e 10% hardware.

- **Dettagli sul processo di sviluppo**

Il team di lavoro applica Scrum e utilizza una pianificazione generale molto grossolana, cercando di adattare il concetto degli story points al progetto. Per monitorare lo stato di avanzamento dei lavori utilizza un burndown chart, appoggiandosi sulla piattaforma Jira per tracciare le attività. Partendo da un backlog dei requisiti, il team stima il tempo di rilascio per ogni task.
L'organizzazione non usa team cross-funzionali dedicati ai progetti, quindi i gruppi di lavoro si smontano e si rimontano in base alle esigenze e le risorse lavorano in parallelo su progetti diversi. Dunque, la gestione dei progetti a livello di risorse è tuttora ancorata in una mentalità a WBS.
Il lavoro è organizzato in sprint della durata di **2 settimane**. I prodotti sviluppati partono sempre da versioni già esistenti sul mercato (non partono mai da una tabula rasa). La definizione di "fatto" non è sempre la stessa e può evolvere in base allo stato di avanzamento del prodotto sviluppato, soprattutto nel concetto di hardware: l'hardware passa attraverso diversi gradi di descrizione che partono da un livello molto astratto (bozza, disegno, lavagna, ...) e poi si spingono in profondità a livelli di dettaglio sempre più precisi, fino a prendere forma in veri e propri blocchi per rappresentare le maschere. Quindi si ha questo concetto di approssimazione successiva, in cui il prodotto si può concepire con diversi livelli di "done" affrontati di volra in volta. La stessa funzionalità passa da "done" a livello simulazione, "done" a livello schematico, "done" a livello layout, e così via. L'integrazione con il software avviene in tutto il percorso. Il prototipo fisico è generato nell'ultima fase di verifica

e arriva in laboratorio per un test finale; in tutte le fasi precedenti del progetto si lavora su astrazioni in maniera iterativa.
Il cliente non partecipa attivamente, ma è consultato regolarmente ed è la prima fonte di feedback concreta che il team di sviluppo riceve. Nel caso di più clienti, entra in gioco la business unit preposta che funge da interfaccia.

- **Problematiche maggiormente riscontrate**

L'intervistato ha affermato che non avendo un team cross-funzionale, è difficile coordinarsi internamente. Lavorando ancora con una mentalità di WBS entrano in gioco dei vincoli interni.

6.1.4 Intervista 4

- **Azienda e intervistato**

L'azienda in cui lavora il quarto intervistato ha diverse linee di business ed è conosciuta principalmente per la produzione di compressori industriali. In particolare, l'intervistato lavora nella business unit degli industrial tools (utensili industriali) e si occupa principalmente dello sviluppo di strumentazione per tarare gli avvitatori industriali e controllare la qualità della strumentazione. Questi dispositivi di quality assure sono venduti alle stesse aziende che utilizzano gli avvitatori.
L'intervistato si occupa della gestione dei progetti meccatronici in R&D, lavorando con un approccio ibrido per lo sviluppo della parte meccanica, in cui Agile è integrato in un metodo di lavoro tradizionale.

- **Dettagli sul prodotto**

Nel corso dell'intervista sono state esaminate due tipologie di prodotto: un trasduttore di coppia e angolo, composto da circa 50 parti, e un banco di simulazione per giunti molto complesso, dotato di parecchi freni e un'elettronica con una centralina, che supera il centinaio di parti.
Il banco di simulazione per giunti è un prodotto venduto a clienti automotive per calibrare gli avvitatori senza dover essere messo in linea. Si tratta di un prodotto **modulare:** ci sono

dei moduli che si possono integrare e che sono stati sviluppati a parte, come integrazione del prodotto. I banchi sono tarati dall'elettronica: l'operatore va a posizionare l'avvitatore sul banco e tramite computer seleziona i parametri che si vogliono simulare. A questo punto si fa partire l'avvitatore e il banco permette di leggere la coppia misurata. I banchi hanno una centralina che alimenta tutto il dispositivo, un'elettronica che ha la funzione di leggere i dati ed elaborarli e un pc a bordo o eventualmente un tablet per l'interfaccia con la macchina.
Il trasduttore coppie/angolo è un prodotto molto **integrale**. Il sistema dei componenti è abbastanza limitato e la revisione di una features o l'eliminazione di una specifica comporta la revisione dell'intero prodotto. Il trasduttore ha la funzione di leggere una differenza di potenziale elettrico per valutare la prestazione dell'avvitatore. La misura è passata all'elettronica che salva i dati in una memoria interna.
Entrambi gli strumenti sono collegabili con la piattaforma che l'azienda mette a disposizione dei suoi clienti per storicizzare i dati raccolti in un database e tenere traccia dell'efficienza degli strumenti.
Il contributo funzionale di ogni macro-componente è simile sia per il banco che per il trasduttore, in quanto sono entrambi ad alto contenuto meccanico ed elettronico. Il software mette a disposizione alcune funzionalità che consentono di differenziarsi rispetto alle altre soluzioni esistenti, quindi non è responsabile delle funzionalità core, bensì è un plus per cui i clienti sono disposti a pagare e a scegliere le soluzioni offerte dall'azienda. In particolare, per il banco l'88% delle funzionalità sono concentrate nella componentistica elettronica e meccanica, mentre la restante parte è assegnata al software; per il trasduttore invece è possibile dividere 98% hardware e 2% software.

- **Dettagli sul processo di sviluppo**

L'azienda usa le pratiche previste da Scrum, principalmente per lo sviluppo dei prodotti software e firmware. Per quanto riguarda l'hardware, invece, utilizza una metodologia ibrida: un approccio Waterfall a livello macro, con il processo diviso in gate che va dalla fase di pre-studio fino a quella di vendita. All'interno delle varie fasi, soprattutto quelle di pre-studio in cui non è ancora definito un design vero e proprio, ci sono dei momenti in cui l'azienda attua uno sviluppo incrementale, lavorando con un backlog di features prioritizzate: ci sono meno pressioni esterne e più possibilità di fare dei cicli di prova, per cui si possono anche testare varie configurazioni, in modo da comprendere quali features includere nel prodotto e con quali costi. Man mano che il prodotto evolve e diventa più

maturo è più difficile per la componentistica hardware fare delle modifiche sostanziali. Tuttavia, si possono includere dei dettagli che prima non erano stati approfonditi, sempre che non vadano a stravolgere la struttura.
Gli sprint hanno una durata di **1 mese** (mentre per il software e il firmware sono di 3 settimane) e la Definition of Done varia in base alla consegna dello sprint: si parte da valutazioni teoriche, in cui si deve scegliere quale configurazione hardware è la più idonea sulla base delle specifiche tecniche comunicate dal marketing, a successivi test su schede già sviluppate. Dopo varie iterazioni è possibile arrivare a diversi risultati sperimentali che mostrano qual è la soluzione migliore. Per la restante parte del progetto, il team lavora in maniera Waterfall.
I nuovi prodotti sono sviluppati seguendo le linee guida del marketing che collezionano le richieste dal mercato, dunque i clienti non forniscono feedback in sede di review. Inoltre, l'azienda riscontra alcune problematiche legate ai tempi dei fornitori, per cui non è sempre facile garantire una time box precisa per lo sviluppo.

- **Problematiche maggiormente riscontrate**

Man mano che il prodotto evolve e matura è difficile portare modifiche sulla parte hardware perché hanno un impatto elevato sul progetto. Inoltre, i tempi di sviluppo dipendono molto dai fornitori, per questo motivo non è sempre facile avere una time box ben definita.

6.1.5 Intervista 5

- **Azienda e intervistato**

L'azienda progetta, costruisce vende e installa macchine di stampo metalmeccanico per la lavorazione di diversi tipi di materiale.
L'intervistato, che copre il ruolo di Senior Scrum Master, ha iniziato a lavorare nell'azienda da circa 2 anni per portare internamente l'approccio Agile e dare supporto ai team di sviluppo prodotto che ideano e realizzano i primi prototipi delle nuove macchine. Dunque, nel suo ruolo integra anche il lavoro di un Agile Coach, coprendo lo sviluppo di tutte le linee di business nell'azienda.

L'azienda ha iniziato a integrare le pratiche agili a partire dallo sviluppo dei macchinari, senza prima testarle nello sviluppo software, dunque lavora in modo Agile per lo sviluppo hardware, ma non per lo sviluppo software. Tuttavia, non tutte le linee di business applicano il framework agile in modo altrettanto efficacie, poiché alcune di queste riescono a essere profittevoli anche senza di organizzarsi in questo modo: è il caso di prodotti in cui l'azienda è sostanzialmente monopolista.

- **Dettagli sul prodotto**

I macchinari industriali sono prodotti altamente complessi e possono superare anche le centinaia di componenti. In particolare, volendo concentrare l'attenzione sulle macchine utensili, queste presentano un'architettura **modulare**.
I componenti elettrici hanno il compito di muovere le varie parti della macchina. Ci sono dei controllori CN (controllo numerico), delle interfacce grafiche che permettono di tenere sotto controllo il pezzo in lavorazione e che aiutano gli operatori e i manutentori a fare pulizie. Le macchine iniziano ad integrare anche un po' di intelligenza artificiale. Infine, ci sono dei software che aiutano nel taglio e nella lavorazione, grazie ai quali l'operatore va a caricare in macchina il disegno del pezzo e questo viene eseguito in modo quasi automatico.
L'azienda è di carattere meccanico, quindi il valore è concentrato sulla componentistica meccanica. Tuttavia, sta iniziando a valorizzare anche il software a causa della forte spinta alla digitalizzazione richiesta dal mercato. Dunque, il contributo funzionale di ogni macro-componente può essere così scomposto: 15% per la parte elettronica, 60% per la parte meccanica e 25% per il software, che corrisponde a 75% hardware e 25% software.

- **Dettagli sul processo di sviluppo**

Per lo sviluppo hardware, l'azienda ha integrato l'Agile all'interno delle pratiche tradizionali, lavorando quindi con un framework ibrido e non con un Agile puro. In particolare, i team di sviluppo usano Scrum adattato ai prodotti fisici, ibridato a un modello tradizionale Stage-Gate, in cui ci sono delle fasi che devono precederne altre, intervallate da Gate attraverso cui i progetti devono obbligatoriamente passare. Paradossalmente, i progetti puramente software non usano pratiche agili in senso stretto.
C'è una prima fase di funnel in cui gli stakeholders e il comitato di sviluppo nuovo prodotto decidono se vale la pena esplorare un nuovo progetto. Una volta approvato, viene dato un

orizzonte temporale di massima che è aggiornato in corso d'opera in base alle iterazioni fatte con gli sprint (come prevede Scrum). Una volta al mese è fatto un aggiornamento a livello di portfolio che sistema la parte ibrida verso lo Stage-Gate, in cui si fissano delle milestones generali e non delle informazioni più precise.
Il team lavora sulla base di un backlog e quindi in ogni sprint pianifica quale parte rilasciare. Un primo livello di monitoraggio si verifica a livello di singoli sprint, direttamente dalle storie completate, dando un'idea di quali sono le funzionalità e gli aspetti del prodotto esplorati. Dopo di che c'è un avanzamento temporale globale che è monitorato aggiornando la roadmap complessiva, in cui c'è un Gantt minimale composto di milestone.
A ogni progetto è assegnato un team multifunzionale, composto da 15 – 18 persone che nel complesso possiedono tutte le maestranze necessarie per lo sviluppo; le persone sono dedicate al progetto per tutta la sua durata, dunque non sono coinvolte in più commesse in parallelo.
Gli sprint hanno una durata fissa di **2 settimane**; nelle fasi iniziali del progetto, quando si è nelle macro-fasi di analisi fattibilità e progettazione, i deliverable consistono in disegni, report, analisi, quindi sono dei documenti che possono avere diversa natura. Successivamente, si esplorano di volta in volta dei disegni tridimensionali dei componenti. Tutto lo sviluppo non si ferma finché non è realizzato il primo prototipo fisico, in cui si cominciano a montare dei sottogruppi della macchina che poi andranno assemblati; al prototipo si pensa già in fase intermedia, dovendo ottenere l'approvazione per l'acquisto dei materiali. Inoltre, negli sprint si può avere anche lo studio di una piccola parte della macchina nel caso in cui questa rappresenti un elemento critico che deve essere esplorato separatamente.
Nelle review partecipano anche i responsabili funzionali, che sono interessati allo sviluppo in quanto i loro operai dovranno lavorare a contatto con le macchine; in questo modo possono dare dei feedback o suggerire delle correzioni sui deliverable.
Il rapporto con i fornitori è l'ultimo vincolo che rimane in gioco: il fatto di dover passare da una fase di approvazione per procedere con l'approvvigionamento è il motivo per cui l'azienda non ha abbandonato del tutto la metodologia di sviluppo tradizionale.

- **Problematiche maggiormente riscontrate**

Il rapporto con i fornitori rimane l'unica vera problematica da gestire; è un vincolo in quanto il team di sviluppo non può iniziare a realizzare prototipi finché non sono arrivati i materiali.

6.1.6 Intervista 6

- **Azienda e intervistato**

L'intervistato 6 ha lavorato nella stessa azienda dell'intervistato 1, ma si è occupato dello sviluppo di un prodotto differente, che comunque rientra nei sistemi per il processamento delle banconote nel settore bancario. Anche in questo caso si tratta di un dispositivo fisico, una cassaforte automatica utile per le operazioni di back office di gestione del denaro. L'intervistato è stato Scrum Master nell'R&D dell'azienda e ha iniziato a lavorare con le pratiche agili direttamente dall'hardware.

- **Dettagli sul prodotto**

In particolare, l'intervistato ha seguito lo sviluppo di depositi, ovvero dei dispositivi che consentono il deposito del denaro, ma non il prelievo. Sono dei macchinari con un alto numero di componenti (intorno ai 2000 pezzi per macchina) e un'architettura **modulare**.
La componentistica digitale consente di conteggiare le banconote depositate tramite un lettore che costituisce l'intelligenza della macchina. Grazie allo stesso sistema, a fine giornata l'operatore può visionare lo stato cassa a video. Inoltre, sono presenti i componenti responsabili del movimento, quindi le schede elettriche e il firmware che aiutano a leggere, riconoscere e movimentare le banconote, a sportarle all'interno per arrivare in cassaforte e collocarle nell'apposito spazio in base al taglio.
Il contributo funzionale di ogni macro-componente è: 80% per i componenti meccanici ed elettronici, 20% per il software.

- **Dettagli sul processo di sviluppo**

Durante il processo di sviluppo, il team utilizzava il framework Scrum, con tutte le sue cerimonie per pianificare, aggiornarsi, monitorare lo stato di avanzamento lavori e concludere gli sprint. Appoggiandosi al software Jira, i task da realizzare erano monitorati facendo ricorso a una dashboard digitale ed erano pesati con un punteggio mediante la serie di Fibonacci.
Il team faceva una pianificazione di massima all'inizio e poi dettagliava le funzioni in corso d'opera perché spesso emergevano delle nuove esigenze. Dunque, c'era un macroflusso che

prendeva mesi e dava un po' di visibilità ai vari gruppi su quello che si sarebbe dovuto fare, mentre tra ogni sprint c'era una pianificazione più di dettaglio.
Il team era multifunzionale, composto da circa 15 persone che coprivano diverse competenze: meccanica, elettrica, firmware, software e ingegnerizzazione (uno specialista di prodotto).
Per lo sviluppo hardware erano definiti degli sprint di **21 giorni**. La Definition of Done di ogni sprint poteva consistere in diversi risultati in base alla consegna, come il disegno meccanico di una parte e la sua documentazione, mentre la produzione poteva occupare un intero sprint a causa dei lunghi tempi di approvvigionamento e di montaggio. Verso la fine del progetto poteva esserci prototipazione fisica di certi componenti realizzati con una stampante 3D e talvolta al termine di uno sprint poteva essere presentato un modulo fatto. Tuttavia, la prototipazione era considerata un costo, uno spreco di tempo e risorse, dunque si faceva ricorso solo quando strettamente necessario.
Il rilascio incrementale hardware avveniva in parallelo con quello software, anche se quest'ultimo utilizzava sprint di 2 settimane, dunque i rilasci non erano sempre sincroni.
Per ciò che riguarda il rapporto con i fornitori esterni, non sono stati riscontrati particolari problemi: nei tempi di attesa si spostava l'attenzione ad altri moduli, mettendo in standby quel pezzo specifico finché non sarebbero arrivati i materiali. Invece, la partecipazione del cliente finale agli sprint non è stata colta: il product manager era l'interfaccia diretta con l'esterno e raramente partecipava agli sprint.

- **Problematiche maggiormente riscontrate**

Le difficoltà maggiormente sentite sono state quelle di progettare un oggetto fisico in tempi più stringenti e gestire dei gruppi misti molto numerosi. Altre difficoltà riscontrate, più relative all'azienda in sé che non ai vincoli di fisicità dei prodotti, è stata la mancanza di convinzione da parte di tutto il management. Inoltre, dopo un po' di tempo, l'azienda non ha più investito in risorse riducendo il numero di Scrum master: i pochi rimanenti hanno dovuto gestire in parallelo tanti team numerosi, provocando confusione e inefficienza generale.

Nota: l'intervistato 1 e 6 lavorano nella stessa azienda, ma si sono occupati dello sviluppo di due prodotti differenti. I prodotti sono trattati in modo diverso nelle analisi, tuttavia quando si farà riferimento generale all'azienda, per semplicità si farà riferimento all'azienda 1.

6.1.7 Intervista 7

- **Azienda e intervistato**

L'azienda realizza strumenti di scansione tridimensionale e strumenti di supporto alle attività sui dati scansionati. In particolare, l'intervistato, nel ruolo di sviluppatore hardware, si è occupato dello sviluppo di una linea di scanner odontoiatrici utilizzata per la ricostruzione dei denti a partire da modelli in gesso.

- **Dettagli sul prodotto**

Lo scanner dentale 3D è composto è composto da diversi componenti che erano acquistati esternamente, tra cui telecamere, sistemi di proiezione, motori e schede elettroniche, mentre della BOM sviluppata internamente facevano parte circa 200 pezzi. Il prodotto è monolitico ed è caratterizzato da un'architettura **integrale**; richiede che i componenti siano sincroni tra loro e la criticità del singolo componente rischia di fermare l'intero sistema. Le telecamere, insieme a un sistema di proiezione, hanno la funzione di scansionare l'oggetto. La componente elettronica è necessaria per gestire i motori con cui muovere l'oggetto da scansionare, insieme a qualche led per avvisare che lo scanner è in funzione, mentre la sensoristica ha il compito di rilevare se l'oggetto è in posizione corretta. La parte più funzionale consiste nella proiezione dell'immagine che è rimandata al computer collegato allo strumento. Non c'è un software integrato direttamente nel dispositivo, ma la soluzione venduta comprende un portafoglio di scanner più software dedicato da utilizzare a calcolatore. Dunque, tutta la parte di logica si trova sul computer, mentre il prodotto in sé contiene i singoli componenti e la meccanica.
Nel suo complesso, l'hardware è importante quanto il software: il prodotto non è scindibile dal programma necessario per rilevare e rielaborare le immagini, ma d'altra parte tutte le telecamere devono avere una qualità adeguata a garantire i risultati giusti, mentre i motori e il resto dei componenti meccanici sono necessari per avere un'usabilità ottimale. Focalizzando l'attenzione sulla funzionalità di acquisizione dell'immagine, il contributo dei macro-componenti può essere scomposto in 55% hardware e 45% software.

- **Dettagli sul processo di sviluppo**

La pianificazione dei progetti ha seguito un processo abbastanza rigido, in quanto la progettazione di un nuovo prodotto doveva passare per certi step ben precisi: c'erano alcuni stadi predefiniti da cui non era possibile tornare indietro e questo era dovuto principalmente alla compliance alle normative di prodotto.
Lo sviluppo ha seguito un modello ibrido piuttosto rigido: c'era un macro-piano gestito con un diagramma di Gantt, tramite cui il management si interfacciava con i team leader dei vari team di sviluppo, mentre i gruppi avevano il compito di stimare la durata di ognuna delle fasi. Invece, le singole fasi erano gestite all'interno del team tramite cicli di sviluppo per sprint. Le variazioni in corso d'opera dovevano essere comunicate al management, mentre tutte le funzionalità principali da implementare erano stabilite fin dall'inizio, tramite un backlog prioritizzato, dove ogni requisito aveva assegnato un range di accettabilità.
Il team usava un burndown chart per sprint, appoggiandosi a Jira per gestire le board virtuali. I gruppi erano divisi per competenze e, in particolare, la squadra di lavoro era composta da 3 team: 2 di sviluppo software (uno applicativo e uno algoritmico) e uno di sviluppo hardware che seguiva tutti i prodotti. Il team di sviluppo hardware era composto da 4-5 persone e un team leader di riferimento, e si aggiornava usando le cerimonie Scrum.
I lavori per lo sviluppo hardware erano organizzati in sprint della durata di **2 settimane**, così come per il software, ma i due mondi erano abbastanza indipendenti tra loro. Il risultato di uno sprint lato hardware poteva consistere nella definizione di un nuovo design, nella verifica che questo fosse compatibile con le procedure di produzione, che fosse approvvigionabile in maniera corretta e rientrasse nel budget. Inoltre, poteva esserci un set di specifiche, un campione, un disegno tecnico e così via. All'inizio dello sviluppo era pianificato il rilascio di un certo numero di prototipi da realizzare nel corso del tempo e per ognuno di essi si definiva cosa doveva essere validato. Tuttavia, erano predefiniti alcuni stadi da cui non era più possibile apportare modifiche sostanziali che mettessero in discussione l'intera struttura. Questo era dovuto dalle proprietà integrali dell'architettura, ma soprattutto dalla compliance alle normative di prodotto. Infatti, una volta che il prototipo era certificato, non era più possibile modificare nulla che compromettesse il rispetto della normativa stessa, ma potevano essere apportate solo piccole modifiche che non impattassero sul certificato di test.

La validazione funzionale con il cliente non era fatta al termine degli sprint, ma a delle milestone intermedie in cui erano definite chiaramente le fasi prototipali. Quindi il cliente era coinvolto nel momento in cui veniva rilasciato un prototipo.

- **Principali difficoltà**

In generale, le dipendenze hanno costituito la problematica maggiormente sentita dagli sviluppatori dell'azienda. Da un lato c'erano le dipendenze da vendor esterni, che potevano causare il ritardo nella conclusione degli sprint, dall'altro le dipendenze da fornitori interni (altre funzioni aziendali) a causa di prioritizzazioni diverse tra le attività. Inoltre, trattandosi di un prodotto odontotecnico, il rapporto con gli enti certificatori ha costituito un altro elemento di dipendenza esterna, rendendo difficile l'integrazione dell'attività di test per le certificazioni all'interno degli sprint.

6.1.8 Intervista 8

- **Azienda e intervistato**

L'azienda è una società multinazionale che opera in tutta la catena del valore del gas naturale e nello specifico l'intervistato è R&D manager dell'area elettronica, che si occupa dello sviluppo di contatori del gas volumetrici.
Il gruppo ha iniziato a inserire l'Agile da circa 2 anni, migrando da un approccio Lean e Kaizen, partendo con progetti pilota nel quartier generale applicati a prodotti meccanici e/o alla logistica per poi via via contaminare tutte quante le aree, arrivando anche all'elettronica. Nella specifica divisione dei contatori, l'Agile è stato approcciato in prima battuta allo sviluppo di tipo firmware, che è l'interfaccia tra l'hardware e il software.

- **Dettagli sul prodotto**

Il contatore gas volumetrico è un prodotto meccanico a cui è applicata un'elettronica embedded di conteggio, dentro al quale gira un firmware specifico. Le funzionalità della componente digitale è rilevare e misurare il dato. In generale il contatore ha lo scopo di misurare un parametro (flusso o volume che sta percorrendo il contatore stesso), elaborarlo,

memorizzarlo, dare evidenza del dato con statistiche e valori agli utenti finali tramite il display e poi inviare il dato a un centro di raccolta.
L'azienda sviluppa circa un centinaio di componenti del contatore, prodotto con un'architettura fortemente **integrale**.
Considerando il contributo funzionale dei macro-componenti di un contatore con tecnologia statica, può essere scomposto in 95% hardware e 5% software.

- **Dettagli sul processo di sviluppo**

Il prodotto ha specifiche tecniche stringenti, dunque i requisiti sono noti fin dall'inizio. La baseline è suddivisa in sprint, valutando i change request in corso d'opera. La progettazione si adatta alla contingenza, tuttavia, trattandosi di un prodotto standard, deve passare in momenti ben precisi attraverso certi gate.
Nei progetti di ricerca e sviluppo l'azienda utilizza un approccio estremamente dinamico basato su Scrum, adattando la programmazione dei lavori sui nuovi input che emergono. I task da realizzare sono ordinati nel backlog e misurati in termini di complessità, assegnando loro un punteggio tramite la scala di Fibonacci e monitorati con l'appoggio di virtual planning sul software Jira.
A livello di dettaglio, i lavori sono organizzati in sprint di **1 o 2 settimane** a seconda delle fasi di sviluppo in cui ci si trova, con allineamenti periodici previsti dalle cerimonie Scrum: all'inizio gli sprint sono molto ravvicinati, soprattutto quando i gruppi di lavoro sono nuovi, in seguito diventano quindicinali.
Il risultato di uno sprint può coincidere con la definizione di una o più specifiche tecniche, delle simulazioni a livello hardware oppure virtuali sul funzionamento di una parte, la definizione di una funzione, un risultato tecnico o un prototipo.
Ereditando il metodo di lavoro dall'approccio Lean, il prodotto è sviluppato con cicli di affinamento per fasi successive: si arriva al prodotto per fasi, partendo dall'elemento più esterno e procedendo per zoom via via più dettagliati, fino a quando non si raggiunge il minimo grado di dettaglio del prodotto. Man mano che si procede con lo sviluppo, si realizzano dei prototipi.
Tuttavia, una volta che la piattaforma hardware è stata approvata, questa rimane piuttosto bloccata perché porta a delle scelte di investimento importanti e un suo cambiamento comporterebbe la messa in discussione dell'intero progetto. Questo vuol dire che, una volta che i prototipi realizzati (per esempio con stampanti 3D) sono stati approvati, allora si

effettuano gli investimenti di approvvigionamento dei materiali necessari a realizzare i prototipi finali per ottenere le omologazioni (per queste ultime operazioni è necessario avere il prodotto finito). L'approccio agile può continuare a esistere, ma senza portare modifiche rilevanti.
Dunque, in generale si può affermare che l'azienda utilizza un approccio Agile lato hardware per le prime fasi di sviluppo, ibridato a uno Stage-Gate.
Il lavoro è gestito da più team in parallelo, specifici per la tipologia di prodotto. Le risorse non sono bloccate sul singolo progetto (come invece vorrebbe un approccio puramente Scrum), ma sono dedicate per il massimo del tempo possibile, ovvero per almeno 3 giorni, mentre i restanti 2 giorni lavorano su altri progetti che non seguono un approccio agile. Se questo non è possibile, allora si creano dei gruppi specifici che lavorano su specifici argomenti, organizzati a livello funzionale in base alla loro specialità.
Alle review si cercano di coinvolgere tutti gli attori che possono dare un contributo, in alcune fasi anche il cliente, soprattutto all'inizio in sede di definizione dei requisiti.

- **Principali difficoltà**

La criticità generata dal rapporto con i fornitori è stata risolta stipulando delle partnership strategiche con questi ultimi. Alcune problematiche continuano a sussistere lo stesso poiché certi componenti non possono essere trovati rapidamente. Un ulteriore problematica riguarda la possibilità di apportare delle modifiche sull'hardware: l'agile è applicato in modo efficiente fin quando non sono congelati gli investimenti, dopo di che risulta complesso apportare dei cambiamenti.

6.1.9 Intervista 9

- **Azienda e intervistato**

L'intervistato 9 lavora in un'azienda multinazionale che sviluppa e realizza prodotti telecom, in particolare nella business unit "microwave", dunque si occupa di ponti radio. L'azienda ha introdotto l'Agile nello sviluppo software a partire dal 2006, mutuandolo successivamente anche nell'hardware.

- **Dettagli sul prodotto**

Il prodotto di cui si occupa l'intervistato consiste in una scheda, un PCB (printed circuit board), su cui girano delle logiche riprogrammabili a cui si possono aggiungere delle features in maniera incrementale, dette FPGA. Il prodotto è quindi un cheap vero e proprio ed essendo in un contesto molto simile al software, si presta bene per l'implementazione delle pratiche agili. La parte di sviluppo incrementale coinvolge soprattutto le logiche del FPGA, che sostanzialmente sono logiche completamente riprogrammabili implementate a bordo di un PCB.
Il prodotto ha un'architettura **modulare**, con funzioni attribuibili per l'80% al software e il 20% all'hardware: il valore fornito dalla componentistica software è molto elevato, tuttavia l'hardware ha un ruolo non del tutto trascurabile in quanto deve essere in grado di garantire una certa banda e una certa velocità dell'apparato.

- **Dettagli sul processo di sviluppo**

Nel processo di sviluppo, il team utilizza il framework Scrum di Scrum, ovvero degli sprint che girano all'interno di un Main Sprint. Oltre alle cerimonie previste da Scrum, il team partecipa a dei meeting (detti main stream) ogni 4 giorni circa, in cui c'è un allineamento generale sullo stato di avanzamento del main sprint.
Il team tiene una pianificazione a livello più generale di program management, attribuibile al Main Sprint. In parallelo, per tutti i sotto-sprint sono gestite delle pianificazioni di dettaglio, che si innestano sul path principale per tenere traccia della deadline di progetto. Per monitorare i lavori si utilizzano delle board virtuali, condivise e visibili a tutti, tenendo traccia dello stato di avanzamento tramite burndown chart.
Gli sprint hanno una durata di **4 settimane**. Inoltre, il team ha introdotto il concetto di "sprint 0" che consiste in una riunione di brainstorming in cui ogni membro del team porta dei feedback dai progetti precedenti. L'output dello sprint 0 è tipicamente uno studio, mentre la Definition of Done per gli altri sprint può consistere in simulazioni virtuali con tool specifici e l'hardwarista deve arrivare con delle analisi di integrity in base agli input che riceve. Quando si converge verso una soluzione che sta in piedi, allora si passa alla parte di produzione, cioè si costruisce il PCB. In generale non c'è prototipazione, altrimenti i costi si alzano a dismisura, ma il team ha la possibilità di effettuare due run per PCB: un primo run per verificare i risultati delle simulazioni, un secondo run che comprende delle modifiche

da apportare sul PCB. Questo secondo run è effettuato solo se necessario, in ottica di miglioramento del primo prototipo.
Il team di lavoro è cross funzionale, dunque è composto da un sistemista, il test engineer, il responsabile di validazione, l'hardwarista e tutte le sotto specializzazioni. Può comprendere fino a 20 persone, mentre all'interno del team c'è una suddivisione in sotto-team più atomici, che gestiscono dei rilasci più brevi e che vanno a confluire nel main sprint.
I clienti sono consultati all'inizio per la definizione dei requisiti e poi nelle fasi di check intermedie. Tuttavia, l'azienda non sviluppa prodotti per un cliente specifico, per questo motivo la comunicazione non è diretta, ma passa attraverso le figure aziendali responsabili della mediazione.
I prodotti non sono custom, ma si cerca di bilanciale le caratteristiche volute dall'azienda con quelle richieste esternamente e si tenta di realizzare dei prodotti il più possibile riconfigurabili a seconda delle esigenze del cliente.
I rapporti con i fornitori sono sempre stati molto stretti: quando possibile, l'azienda se li è portati direttamente in casa per avere un aiuto nello sviluppo. Spesso un particolare era sviluppato direttamente dal fornitore, integrandolo nello sviluppo di alcune parti (soprattutto per i FPGA).

- **Principali difficoltà**

L'intervistato non ha riscontrato particolari difficoltà nell'implementazione dell'Agile. I vincoli esterni dovuti ai fornitori ci sono sempre, ma la maggior parte di essi è stata risolta mediante la stipulazione di contratti corporate.

6.1.10 Intervista 10

- **Azienda e intervistato**

L'intervistato 10 è Product Owner in un'azienda che opera nel settore delle micromeccaniche di precisione, realizzando delle macchine che si occupano di lavorare materiale prezioso, principalmente catene per gioielli. In particolare, l'intervistato si occupa dello sviluppo di laser per la saldatura dei singoli anelli delle catene.

L'azienda ha iniziato ad applicare le pratiche agili al processo di sviluppo da circa un anno e mezzo, mutuandole dallo sviluppo software che invece fa uso delle pratiche da 5 anni.

- **Dettagli sul prodotto**

Il prodotto è un laser digitalizzato, con un'architettura fortemente **modulare**, ed è composto da circa 200 componenti. La digitalizzazione del sistema è legata al controllo elettronico dei singoli moduli e del sistema di emissione del laser; i moduli nel loro insieme sono gestiti da un software e un firmware sviluppati internamente. Il prodotto ha una forte presenza di componenti elettroniche e digitali, con funzioni attribuibili al 60% hardware e 40% software.

- **Dettagli sul processo di sviluppo**

Nel processo di sviluppo il team usa il framework Scrum, con tutte le sue cerimonie. La parte di retrospective è stata alleggerita nel corso del tempo per non forzare la consegna di qualcosa da mostrare.
La pianificazione iniziale non è mirata: non sono più dettagliate tutte le specifiche di progetto, ma si lavora con un backlog dinamico. Lo stato di avanzamento del progetto è monitorato con burndown chart e burnup chart, assegnando punteggi alle attività mediante la serie di Fibonacci. I lavori seguono una doppia pianificazione: una a livello macro, monitorata dal direttivo aziendale e dai Product Owner, in cui si tiene traccia dello stato di avanzamento complessivo del progetto, e una di dettaglio che lavora con sprint della durata di **15 giorni**.
Per quanto riguarda gli sprint, la Definition of Done può consistere in specifiche progettuali, layout e disegni. Nel limite del possibile il team cerca di mostrare qualcosa alla consegna dello sprint, tramite moduli o sottoinsiemi di prodotti che stanno nascendo, facendo ricorso a sistemi di prototipazione o altri archetipi. Il team, e in generale l'azienda, sta spingendo fortemente a un approccio orientato ai prototipi, lavorando in un'ottica modulare proprio per avere qualcosa da testare. I componenti sono prototipati dalla divisione interna di stampa 3D, garantendo celerità che non si potrebbe avere se si rivolgessero a fornitori esterni o ad altri centri di lavoro in quanto richiederebbero tempistiche molto più dilatate di uno sprint.
In generale, i team di sviluppo non sono cross-funzionali e questo genera delle dipendenze interne nel corso dello sviluppo. Nel caso specifico, l'intervistato lavora con un team in cui

è presente una risorsa dedicata al software, per cui è più semplice garantire una pianificazione efficiente.
In ultimo, i clienti non sono coinvolti nelle review in quanto l'area commerciale dell'azienda funziona da filtro con il mercato.

- **Problematiche principali**

Le dipendenze da fornitori esterni e interni all'azienda costituiscono un problema proprio a causa delle diverse prioritizzazioni tra gli attori, sia in caso di ritardi da parte del fornitore, sia in caso di ingresso di più ordini in azienda. In quest'ultimo caso, si tende a dare priorità alle linee di produzione piuttosto che ad altri reparti interni, causando dei ritardi nella conclusione degli sprint. Dunque, la problematica nasce proprio dalla mancanza di allineamento delle prioritizzazioni con fornitori interni ed esterni all'azienda. Le dipendenze nascono anche dalla mancanza di team non cross-funzionali; solo nel caso del team dell'intervistato c'è una risorsa dedicata al software, per cui è più facile garantire una pianificazione piuttosto efficiente.
Il risultato degli sprint è un altro punto critico da affrontare poiché talvolta si devono mostrare delle attività non testabili, solo per dare un'idea di avanzamento. Le riunioni di retrospettiva sono state alleggerite proprio per evitare la forzatura di voler mostrare qualcosa di finito, soprattutto per le attività minori.

6.2 Analisi dei risultati

Tutte le interviste si sono volutamente concentrate su chi ha messo in pratica i principi Agile in modo da comprendere quanto le aziende riescano a farli propri, inglobandoli all'interno delle loro routine organizzative. Inoltre, la totalità delle aziende in cui lavorano o hanno lavorato gli intervistati si occupano della realizzazione di prodotti meccatronici.

In primo luogo, è possibile notare la mancanza di uniformità, l'assenza di una formula univoca da rispettare e che possa essere valida per tutti, bensì ognuno cerca di tradurre al meglio i principi del Manifesto e adattarli al proprio contesto operativo. Gli approcci di sviluppo e la pianificazione dei progetti non sono uguali per tutti gli intervistati, in quanto

ognuno ha introdotto l'Agile nel proprio processo in modo più o meno spinto. Questo ha consentito di individuare tre diverse macro-configurazioni:

1. **Un approccio più rigido**, in cui i gate sono il cuore dell'intero processo di sviluppo e ogni stage è vincolante per l'avanzamento del progetto. Nelle fasi tra un gate e l'altro si lavora in modo Agile, soprattutto per le fasi iniziali di pre-studio, quando ancora è stato definito un design preciso.
2. **Un approccio intermedio**, in cui sono presenti degli istanti di verifica sotto forma di gate (presenti, ma flessibili) o di milestone. Questi servono a indirizzare la corretta esecuzione dei lavori, a introdurre momenti di feedback del cliente o a effettuare ulteriori test su quanto sviluppato. Consentono di mantenere una certa flessibilità e di seguire la pianificazione a lungo termine del progetto. Anche in questo caso si lavora per rilasci incrementali in ognuna delle fasi.
3. **Un approccio più flessibile**, in cui si segue uno sviluppo Agile pressoché puro, in cui i gate sono quasi inesistenti e al più ci si attiene a delle linee guida predefinite.

Tutti gli intervistati fanno riferimento a Scrum, un framework che racchiude al meglio l'insieme delle pratiche agili. Ogni team di sviluppo ha messo in pratica in modo più o meno efficiente quelle che sono le cerimonie previste a partire dallo Sprint Planning, al Daily Scrum Meeting e alla Sprint Review, fino alla Sprint Retrospecitive. L'efficacia di queste riunioni, soprattutto della review, è stata messa in discussione dalla maggior parte degli intervistati a causa della difficoltà intrinseca di decidere in cosa possa consistere la release degli sprint:

"...è stato difficile trovare una quadra, mettersi d'accordo su cosa fosse il pacchetto rilasciato alla fine del ciclo..."

e dalla presenza di numerosi membri del team che non parlano la stessa lingua. Quest'ultima difficoltà è stata riscontrata solo da alcuni dei casi intervistati, dove i lavori sono gestiti da team multifunzionali in cui sono presenti tutte le maestranze necessarie a completare il progetto. Di fatto è complicato far comunicare i progettisti hardware con i progettisti software in quanto ognuno lavora con le proprie specifiche e il proprio linguaggio tecnico. Questa difficoltà risulta sicuramente accentuata per quei prodotti altamente meccatronici, in

cui i due ambiti di sviluppo sono di egual importanza per la buona riuscita del progetto e non sono del tutto indipendenti l'uno dall'altro (come per l'azienda 1 e l'azienda 2).

Rimanendo sempre nell'ottica del lavoro in team, non tutti i casi analizzati rispettano la pratica che prevede l'utilizzo di team unici, con risorse completamente dedicate al progetto e co-localizzate nella stessa stanza. C'è chi tenta di rispettare questa pratica con qualche adattamento, dedicando le risorse a progetti gestiti in maniera Agile per la maggior parte del tempo possibile (come nel caso dell'azienda 8) o chi semplicemente continua a utilizzare un approccio più tradizionale in cui le risorse lavorano in parallelo su più progetti (come nel caso delle aziende 3, 7 e 10). Questo aspetto dipende fortemente dalla filosofia aziendale e da come questa intende modellarsi per ridurre i vincoli che possono sorgere in fase di progettazione. Tuttavia, molti hanno dichiarato che non è semplice bloccare le risorse a un solo progetto per volta perché questo richiederebbe una larga disponibilità di progettisti specializzati e quindi maggiori costi per l'azienda.

Anche la gestione della multidisciplinarietà dei team non è sempre rispettata: da un lato per i team multifunzionali, soprattutto se molto numerosi, possono sorgere problemi di comunicazione interna e si possono allungare le tempistiche delle cerimonie Scrum. I due ambiti di sviluppo parlano il proprio linguaggio tecnico e non è sempre possibile comprendersi a vicenda. D'altro canto, questi rallentamenti sono trascurabili se paragonati alle dipendenze interne che potrebbero nascere per team non cross-funzionali, in cui le risorse sono organizzate in base alla propria funzione seguendo una logica a WBS. È il caso dell'azienda 10, per cui l'intervistato ha affermato:

"...Sviluppiamo dei prodotti per cui ci sono team dedicati, un team software e un team hardware. Qui sussistono delle problematiche nella pianificazione dovute alla presenza di Product Owner e backlog diversi con priorità differenti, non è banale rispettare le tempistiche..."

Ovviamente questi problemi vengono meno quando si tratta di prodotti altamente meccanici (come per l'azienda 5 e l'azienda 4), oppure per prodotti molto vicini al mondo puramente software (come per l'azienda 3 e l'azienda 9).

Tutti i team fanno uso di dashboard virtuali, sfruttando il software Jira o altri strumenti di appoggio simili, in modo da avere sempre sotto controllo i task realizzati, quelli in corso e

quelli conclusi. Ogni task include una o più user story pesate in termini di tempo (ore/uomo) o complessità così da quantificare il lavoro. A tal proposito nel corso delle interviste è stata nominata diverse volte la serie di Fibonacci.

Per ciò che concerne la pianificazione, tutti i team fanno uso di due linee parallele più o meno flessibili: una di alto livello che consente di stabilire delle milestone e avere un quadro generale dello stato di avanzamento del progetto da comunicare al top management, e una di dettaglio, ovvero una lente di ingrandimento sugli sprint.

"*...Ci tenevamo una pianificazione parallela un po' tradizionale. C'era un fil rouge che partiva dall'inizio e cercava di arrivare fino in fondo, elencando le tappe principali a cui bisognava arrivare...tra ogni sprint c'era una pianificazione più di dettaglio...*"

All'inizio del progetto sono definiti i requisiti principali e raccolti nel Product Backlog, in una lista più o meno dettagliata in base alle caratteristiche del prodotto, mentre il resto è definito in corso d'opera. Trattandosi di progetti di ricerca e volendo accettare i cambiamenti, non è possibile pensare di includere fin da subito ogni specifica.

I requisiti definiti inizialmente possono essere più o meno dettagliati e rigidi: per prodotti più tradizionali (come per l'azienda 4 e l'azienda 8) o venduti in mercati altamente regolamentati (come per l'azienda 3 e l'azienda 7) sicuramente sono definiti in modo dettagliato fin da subito, con la possibilità di esplorare nuovi requisiti in corso d'opera. Al contrario, per prodotti più innovativi e meno vincolati (come per le aziende 2 e 10), i requisiti sono una "vera scoperta" man mano che si procede con lo sviluppo. In questo senso, l'uso di un backlog risulta essere la base di una gestione agile, un grande punto di svolta rispetto al passato.

"...Solitamente con la metodologia Waterfall succedeva questo: si pianifica in modo maniacale il progetto, si arriva in una fase di verifica o una milestone e si scopriva di essere in ritardo di mesi. Il fatto di aver buttato sul backlog tutte le attività ci permette di avere visibilità, cosa che non avevamo prima..."

Il cliente finale è spesso consultato per la definizione dei requisiti, ma viene raramente coinvolto in prima persona. Solitamente in azienda c'è l'area funzionale preposta che si interfaccia con il cliente in modo diretto oppure in modo indiretto tramite indagini di

mercato. La modalità con cui il suo feedback è inserito nel processo di sviluppo varia da azienda ad azienda, ma solitamente il cliente non partecipa attivamente nelle riunioni di review a fine sprint. Questo risulta da tutte le interviste, eccetto per l'azienda 5, 7 e 8 per cui gli intervistati hanno affermato rispettivamente che:

"... i capigruppo funzionali sono stakeholder d'ufficio, loro partecipano alle riunioni perché i loro uomini lavorano a contatto con le macchine che realizziamo..."

"...cerchiamo di coinvolgere tutti gli attori che possono dare un contributo, anche il cliente in alcune fasi..."

"...La validazione funzionale con il cliente non era fatta a sprint, ma a delle milestone intermedie che coprivano più sprint, in cui avevamo definito delle fasi prototipali ben chiare..."

Diverso è quando si tratta di progetti su commessa, dove la partecipazione del cliente sia in fase di definizione dei requisiti che per controlli in step intermedi è una prassi.

Quanto visto finora è un'analisi generale delle metodologie di sviluppo implementate dalle aziende e dell'allineamento ai principi Agile, tuttavia sono aspetti che prescindono dalle caratteristiche fisiche dei prodotti. A questo punto, per rispondere alla domanda di ricerca è necessario fare un focus sugli sprint e su com'è possibile attuare un processo iterativo e incrementale ai prodotti fisici nei singoli casi specifici, considerando anche l'aspetto delle dipendenze che possono emergere durante il processo di sviluppo. Inoltre, una lente d'ingrandimento sul grado di digitalizzazione del prodotto può fare emergere o meno una tendenza che lega la vicinanza dei prodotti al mondo software all'applicazione di un approccio Agile sempre più "puro".

6.2.1 Gli sprint e la Definition of Done

Il cuore dell'analisi risiede negli sprint, ovvero quel periodo di tempo durante il quale il team di sviluppo ha il compito di rilasciare una porzione incrementale di prodotto, testarla e

validarla. Allo sprint è possibile attribuire le principali problematiche e differenze che emergono nello sviluppo Agile dell'hardware rispetto allo sviluppo di prodotti puramente software, per questo motivo le sue caratteristiche hanno dovuto subire un forte adattamento, consentendo la traduzione delle pratiche agili al contesto tangibile e adeguando a loro volta il concetto di sviluppo iterativo. Questo ha comportato delle grandi modifiche, sia per quanto riguarda la tipologia di consegna rilasciata alla fine di ogni sprint, sia per quanto riguarda la durata degli sprint stessi.

Gli sprint per definizione sono l'unità di base dello sviluppo Scrum e hanno una durata fissa, generalmente da 1 a 4 settimane. Dalle interviste risulta che, nella maggior parte dei casi, gli sprint di sviluppo hardware sono più lunghi rispetto a quelli per lo sviluppo software e in particolare sono emerse tre durate: 15 giorni, 21 giorni e 30 giorni. La responsabilità di un allungamento delle tempistiche e del disallineamento tra i due mondi di sviluppo è sicuramente da attribuire alla difficoltà di realizzare in tempi stringenti delle porzioni di prodotto e consegnare delle release testabili. La suddivisione del lavoro in sprint ha generato non poche difficoltà negli intervistati, soprattutto quando si tratta di stabilire la tipologia di deliverable:

"…a *livello hardware non è facile avere degli obiettivi ben dichiarati e non è facile realizzarli in 2 settimane perché a volte il delivery time di alcuni prodotti richiede più tempo. Per la parte hardware è difficile definire quando una storia è completa oppure no…"*

Inoltre, per alcuni intervistati lavorare per time box ben definiti è stata una vera e propria corsa contro il tempo, in quanto l'obiettivo era quello di arrivare alla fine dello sprint con qualcosa di consegnato e non con qualcosa di qualità, percependo una tendenza ad esasperare un po' l'aspetto quantitativo a discapito di quello qualitativo.

In linea generale, al contrario del software, nel contesto hardware è impossibile ottenere in ogni release una porzione di prodotto incrementale e proprio come ha ribadito un intervistato:

"...non sempre si riesce a mostrare qualcosa di funzionale e vendibile agli stakeholders. Si cerca di farlo piuttosto spacchettando qualcosa di più grande e questa è un po' una

forzatura. Al contrario, i colleghi del software hanno sempre una funzionalità da testare…"

Certo è che lo spacchettamento delle funzionalità del prodotto dipende fortemente dalle caratteristiche intrinseche del manufatto da sviluppare.

I risultati degli sprint possono assumere un carattere diverso in base allo stato di avanzamento del progetto: nelle fasi iniziali, le iterazioni sono molto teoriche e possono avere come output dei report, delle analisi, dei set di specifiche, dei disegni, quindi dei documenti di diversa natura. Successivamente ci possono essere disegni tridimensionali di alcuni componenti, dei video, dei test e dove possibile, si possono esplorare piccole parti dell'oggetto di sviluppo. Quindi all'inizio si tratta principalmente di effettuare delle scelte, poi possono esserci documenti, disegni e simulazioni virtuali, fino ad arrivare alla realizzazione di prototipi testabili. In generale, cioè che si vuole ottenere a fine sprint è qualcosa che aggiunga valore al proseguo del progetto.

Tuttavia è bene specificare che, esattamente come emerso dalle interviste, non sempre è possibile realizzare dei prototipi. Ove possibile gli intervistati hanno fatto ricorso a strumenti di stampa 3D, ma alcuni elementi per loro natura non possono essere prototipati. Infatti, un componente realizzato con materiale differente non è sempre funzionale e spesso può essere molto costoso.

Rimanendo sempre nel merito della prototipazione, di seguito sono riportati due estratti esemplari dalle trascrizioni delle interviste:

"…Realizzare un prototipo è un costo, soprattutto quando non è utile ma serve solo per raggiungere il risultato di uno sprint, perché in quel caso diventa uno spreco di soldi e di risorse…"

"…I prototipi sono necessari, anche se determinano una spesa importante, perché se si arriva in fondo al progetto con un errore, il costo di recupero è talmente alto che ci si ripaga 10 volte tutta la prototipazione e tutta la fase di progettazione. Nessuno ha delle competenze estreme e sa che non ci saranno errori, anche i più esperti sbagliano…"

Queste due frasi fanno emergere due punti di vista in netto contrasto tra loro:

1. La prototipazione è un costo che può non essere sostenuto qualora serva solo per avere il risultato di uno sprint;
2. La prototipazione, anche se costosa, è importante perché consente di avere una visione completa del progetto e comprendere se ci sono errori da correggere.

Non esiste una verità assoluta. Per prodotti a basso valore, probabilmente non varrebbe la pena spendere somme significative nella prototipazione, se questa risultasse difficile da realizzare a basso costo. Sicuramente a questo punto entra in gioco un altro aspetto che esula dallo studio del seguente lavoro di tesi, ovvero la redditività del prodotto sviluppato.

Tornando ad analizzare la Definition of Done degli sprint, dalle interviste risulta che i prodotti hardware sono sviluppati seguendo due strade alternative:

- Si può suddividere il progetto in macroaree, quindi spacchettare il prodotto in parti più piccole e concentrarsi su un singolo pezzo per volta;
- Si può arrivare al prodotto per fasi, partendo dall'elemento più esterno e procedendo per zoom via via più dettagliati, fino a quando non si raggiunge il minimo grado di dettaglio del prodotto.

In entrambi i casi, si può passare per diversi livelli di descrizione, seguendo un concetto di approssimazione successiva, per cui il prodotto si può concepire con diversi livelli di "done" affrontati volta per volta. Quindi la stessa funzionalità passa da done a livello schematico, done a livello layout, done a livello di simulazione e così via.

L'architettura del prodotto è la variabile che determina quale tipologia di scomposizione applicare: infatti, un'architettura modulare consente di suddividere il prodotto in diversi componenti indipendenti tra di loro e affrontare lo sviluppo in modo separato, senza compromettere la funzionalità dell'intero progetto. Questo non può essere messo in pratica in un prodotto integrale, dove i componenti non sono scindibili in sottoinsiemi, ma devono essere sempre analizzati nel loro insieme e la progettazione deve riguardare l'intera struttura.

La modularità in ottica Agile risulta essere un requisito preferito, ma non è sempre possibile stravolgere del tutto l'architettura di un prodotto. I risultati delle interviste hanno mostrato che gli adattamenti sono possibili, tuttavia non portano sempre a un risultato ottimale. Analizzando in particolare il caso degli intervistati 4 e 8, responsabili dello sviluppo di prodotti integrali, è emerso che man mano che il prodotto evolve e diventa più maturo è più

difficile per la parte hardware apportare delle modifiche sostanziali. Si possono includere dei dettagli che prima non erano stati approfonditi, sempre che non vadano a stravolgere l'intera struttura, ma il processo di sviluppo prodotto arriva a un certo punto in cui il design rimane "bloccato". Questo momento corrisponde alla fase in cui si effettuano le scelte di investimento e il blocco è dovuto alla conformazione del prodotto, per cui la modifica di una funzionalità andrebbe a compromettere l'intera struttura del progetto.

"...Finché i cambiamenti avvengono all'inizio della fase di progettazione non c'è nessun problema, nel senso che fino a quando non sono congelati degli investimenti, anche per l'hardware adoperiamo l'agile. Nel momento in cui abbiamo fatto degli investimenti poi non cambiamo più..."

"... A livello aziendale, una volta che è stata congelata la soluzione, aggiungere una funzionalità come si fa in un prodotto software o firmware e lavorare per delta di integrazioni successive di solito non lo facciamo."

Questo limite coinvolge anche l'azienda 7 che produce scanner odontotecnici, tuttavia risente maggiormente del vincolo legato alle certificazioni ottenute da enti esterni (approfondite in seguito). Diversamente, il vincolo architetturale non è risultato valido per l'azienda 3, che produce dei sensori integrati: le piccole dimensioni della scheda consentono di lavorare per livelli di astrazione successive, come se fosse un singolo modulo. Dunque, anche se possiede tutte le caratteristiche di un'architettura integrale non risente pienamente dei suoi vincoli.

In generale, si può affermare che è possibile lavorare in ottica incrementale per prodotti caratterizzati da architetture integrali, tuttavia quando il design di prodotto è ormai maturo, è complicato apportare dei cambiamenti sostanziali. Il costo delle modifiche sarebbe troppo elevato, dunque le strutture rimangono piuttosto bloccate a cui è possibile apportare solo dei piccoli cambiamenti che non impattino sull'intera struttura. Questo vincolo coinvolge prodotti altamente complessi, tuttavia non impatta in modo deciso su prodotti più semplici e di piccole dimensioni, che potrebbero essere assimilabili a singoli moduli da sviluppare.

6.2.2 Le dipendenze interne ed esterne

Un ultimo aspetto che influenza le release negli sprint sono le dipendenze: queste possono suddividersi in dipendenze esterne e dipendenze interne. Le prime riguardano i **ritardi o le mancate consegne da parte dei fornitori** che causano problematiche di continuità nello sviluppo. Diversamente da quanto ipotizzato prima dell'analisi delle interviste, queste dipendenze non influenzano solo i processi per prodotti modulari, ma anche quelli integrali. Per alcuni degli intervistati il rapporto con i fornitori non costituisce un problema, ma per la maggior parte di essi si:

"...Non è facile garantire una time box per lo sviluppo perché ci sono tempi che variano molto in relazione alle tempistiche dei tempi dei fornitori..."

In realtà l'attenzione si deve spostare verso chi non riscontra questo vincolo. In tal senso, gli accordi stipulati da alcune aziende con i propri fornitori sono risultati essenziali per vincere il problema. È il caso dell'azienda 8, che ha stipulato delle collaborazioni tali da ridurre le criticità dei rapporti di fornitura, accorciando anche i tempi di approvvigionamento. Il rischio non si è annullato, si è semplicemente ridotto, poiché alcuni componenti (soprattutto quelli altamente specifici) non possono essere trovati rapidamente e necessitano di tempi di produzione ben determinati. L'azienda 9 rappresenta invece un caso estremo: tramite delle partnership strategiche ha integrato i fornitori direttamente all'interno di alcune fasi specifiche di sviluppo, annullando il problema per quei componenti maggiormente a rischio.

"*…Il contatto con i fornitori era molto frequente, anzi spesso ce li siamo portati in casa per aiutarci nello sviluppo. Spesso abbiamo affidato a loro lo sviluppo di una cosa particolare, perché noi non riuscivamo...*"

C'è chi invece è riuscito a gestire questo vincolo adattandosi di conseguenza, spostando la progettazione su altri moduli e mettendo temporaneamente in standby lo sviluppo del pezzo che necessita di materiale specifico in arrivo dall'esterno. È il caso dell'azienda 1:

"...quando ci sono ritardi da parte dei fornitori oppure quando si verificano dei tempi morti il meccanico va avanti a progettare altri moduli, come se mettesse in standby quel pezzo..."

Per chi invece ancora non ha trovato una soluzione questo problema continua a persistere, rendendo complicato il lavoro in sprint.

Tra le dipendenze esterne rientrano anche quelle legate agli **enti certificatori**. Nonostante la problematica sia stata evidenziata solo dall'intervistato 7, il suo impatto non è da sottovalutare. Alcuni prodotti sono venduti in ambienti altamente regolamentati, che richiedono il rispetto di requisiti molto stringenti per entrare in commercio ed è soprattutto il caso dei prodotti appartenenti al settore medicale. I prodotti devono essere conformi alle normative in vigore ed essere rapidi ad adattarsi qualora queste fossero aggiornate. Le aziende devono prevedere delle milestones da dedicare all'ottenimento delle certificazioni da parte di enti terzi. Tuttavia, le certificazioni vincolano la possibilità di modifiche future in quanto non è più possibile apportare cambiamenti che compromettano il rispetto della normativa, ma sono consentite solo piccole modifiche che non impattino sul certificato ottenuto. Infatti, dalle interviste risulta che:

"...Dato che il prodotto doveva rispettare specifiche normative, una volta che il prototipo era certificato non si poteva più modificare nulla che compromettesse il rispetto della normativa stessa, ma potevano essere fatte solo piccole modifiche che non impattassero sul risultato del certificato di test..."

È il caso dell'azienda 7 che, in quanto produttrice di scanner odontoiatrici, è soggetta a questa tipologia di vincoli. Risultano essere l'ostacolo finale a cui per il momento non è stata trovata una soluzione, se non adattare di conseguenza la pianificazione dei progetti di sviluppo e le possibili modifiche.
Le dipendenze interne (già accennate in precedenza) hanno la stessa natura di quelle esterne, ma semplicemente i fornitori corrispondono alle **unità interne all'azienda**. Il problema di fondo risiede nella differenza di prioritizzazione delle attività:

"...Se io sviluppo un prodotto e devo avere dettagli dal marketing, ma il marketing non ha prioritizzato le attività come l'abbiamo fatto noi, diventa difficile..."

Le dipendenze al di fuori del team sono una grande criticità che ostacolano lo sviluppo e rendono difficile stare nei tempi prestabiliti. Questo potrebbe verificarsi anche nello sviluppo di prodotti puramente software (per cui verrebbero meno le dipendenze di fornitura esterna).

Dunque, si può affermare che, rispetto allo sviluppo software, nello sviluppo hardware la problematica legata alle dipendenze risulta gonfiata dalla componente delle forniture esterne.

6.2.3 Relazione tra flessibilità e digitalizzazione

Man mano che le interviste hanno generato i loro risultati, questi ultimi sono stati ripetutamente analizzati cercando dei collegamenti tra i codici appartenenti alle categorie di 2° ordine. In particolare, è emerso un legame tra il concetto "Digitalizzazione" che caratterizza la tipologia di prodotto sviluppato e "Flessibilità dello sviluppo", cioè la flessibilità del processo seguito dai team. Per mettere in luce questa relazione sono state utilizzate le seguenti dimensioni:

- **Flessibilità del processo di sviluppo**

La presenza di gate intermedi influenza la flessibilità del processo di sviluppo e la possibilità di implementare dei cambiamenti in corso d'opera. Una volta individuate le tre macro-configurazioni di sviluppo che si sono presentate nel corso delle indagini, è stata definita una scala ordinale che assegni loro un valore da 1 a 3, in cui valori più alti corrispondono a maggiore flessibilità:

1 I gate sono il cuore dell'intero processo di sviluppo e sono vincolante per stabilire l'avanzamento o meno del progetto. Nelle fasi tra un gate e l'altro si lavora in modo Agile, soprattutto per le fasi iniziali di pre-studio, quando ancora non c'è un design definito.

2 I gate sono presenti, ma flessibili oppure sono delle milestones intermedie da cui passare. Servono a indirizzare la corretta esecuzione dei lavori, a introdurre momenti di verifica intermedia o di feedback del cliente. Consentono di mantenere una certa flessibilità e di seguire la pianificazione a lungo termine del progetto. Si lavora in modo Agile all'interno delle fasi, tra uno step e l'altro.

3 I gate sono quasi inesistenti, al più ci si attiene a delle linee guida predefinite e lo sviluppo segue un Agile pressoché puro.

- **Grado di digitalizzazione dei prodotti**

Il grado di digitalizzazione è un vero e proprio indice che rappresenta quanto il software sia core rispetto alla totalità delle funzionalità di un prodotto. Prende forma a partire dalle risposte a due domande specifiche all'interno dell'intervista, cioè:

(8) All'interno del prodotto sono presenti dei componenti digitali? Quali? Qual è la loro funzione?

(9) Considerando il prodotto nel suo complesso e volendo suddividerlo fisicamente in componenti, qual è il contributo percentuale in termini funzionali fornito rispettivamente dai componenti meccanici, elettronici e software?

Con la prima domanda è stato possibile comprendere il ruolo della componentistica software nel prodotto, per poi eventualmente correggere le valutazioni soggettive in risposta alla seconda domanda. In questo modo il valore del prodotto in termini funzionali è stato suddiviso tra la componentistica software e quella hardware, ottenendo delle percentuali riportati nella Tabella 3 sottostante (con "hardware" si intende la somma del contributo meccanico ed elettronico, dunque i componenti tangibili di un prodotto).

A partire dalle percentuali, i prodotti sono stati ulteriormente catalogati all'interno di 4 categorie di digitalizzazione, attribuibili a una scala ordinale che assegna un valore da 1 a 4 in base alla prevalenza del software rispetto all'hardware:

1 Prodotti meccanici, prevalentemente non software, ma con qualche contenuto di natura digitale;
2 Il software maggiori funzionalità, ma non è fondamentale;
3 Il software è fondamentale, ma non è totalizzante;
4 Il software è il core delle funzionalità del prodotto.

Tabella 3: Categorizzazione dei prodotti in base al grado di digitalizzazione

Prodotto	% HW	% SW	Categoria
Trasduttore	98%	2%	1
Contatore	95%	5%	
Banco	90%	10%	
Riciclatore	80%	20%	2
Deposito	80%	20%	
Macchinari	75%	25%	
Laser	60%	40%	3
Scanner	55%	45%	
Mano Robotica	40%	60%	
PCB	20%	80%	4
Sensori	10%	90%	

- **Frequenza delle iterazioni**

La frequenza delle iterazioni, quindi la velocità con cui sono rilasciati degli item da testare, è una terza dimensione misurata in relazione alla durata degli sprint. È stata volutamente inserita per verificare l'esistenza di un secondo legame, quello tra "Durata degli sprint" e "Grado di digitalizzazione del prodotto" sotto l'idea che prodotti più digitalizzati (quindi più vicini al contesto software) seguano sprint più brevi. La frequenza delle iterazioni è stata dimensionata attraverso le tre durate emerse nel corso delle interviste: 14, 21 e 30 giorni.

Le dimensioni che caratterizzano ogni prodotto sono riassunte nella Tabella 4 seguente:

Tabella 4: Riepilogo dei dati di analisi

Prodotto	% HW	% SW	Categoria	Flessibilità di sviluppo	Durata degli sprint
Trasduttore	98%	2%	1	1	30 giorni
Contatore	95%	5%	1	1	15 giorni
Banco	90%	10%	1	1	30 giorni
Riciclatore	80%	20%	2	2	30 giorni
Deposito	80%	20%	2	2	21 giorni
Macchinari	75%	25%	2	2	15 giorni
Laser	60%	40%	3	3	15 giorni
Scanner	55%	45%	3	2	15 giorni
Mano Robotica	40%	60%	3	3	15 giorni
PCB	20%	80%	4	3	30 giorni
Sensori	10%	90%	4	3	15 giorni

Una volta definite le grandezze di misura con cui dare evidenza del legame tra i concetti, è stato possibile costruire il grafico a bolle in Figura 20. Sugli assi sono riportati la flessibilità del processo di sviluppo e il grado di digitalizzazione dei prodotti, dando evidenza delle categorie di digitalizzazione a cui possono essere assegnati i prodotti con 4 colori differenti (categoria 1: azzurro, categoria 2: viola, categoria 3: verde, categoria 4: arancione), mentre la dimensione delle bolle è proporzionale alla durata degli sprint (bolle più grandi coincidono con durate più estese).

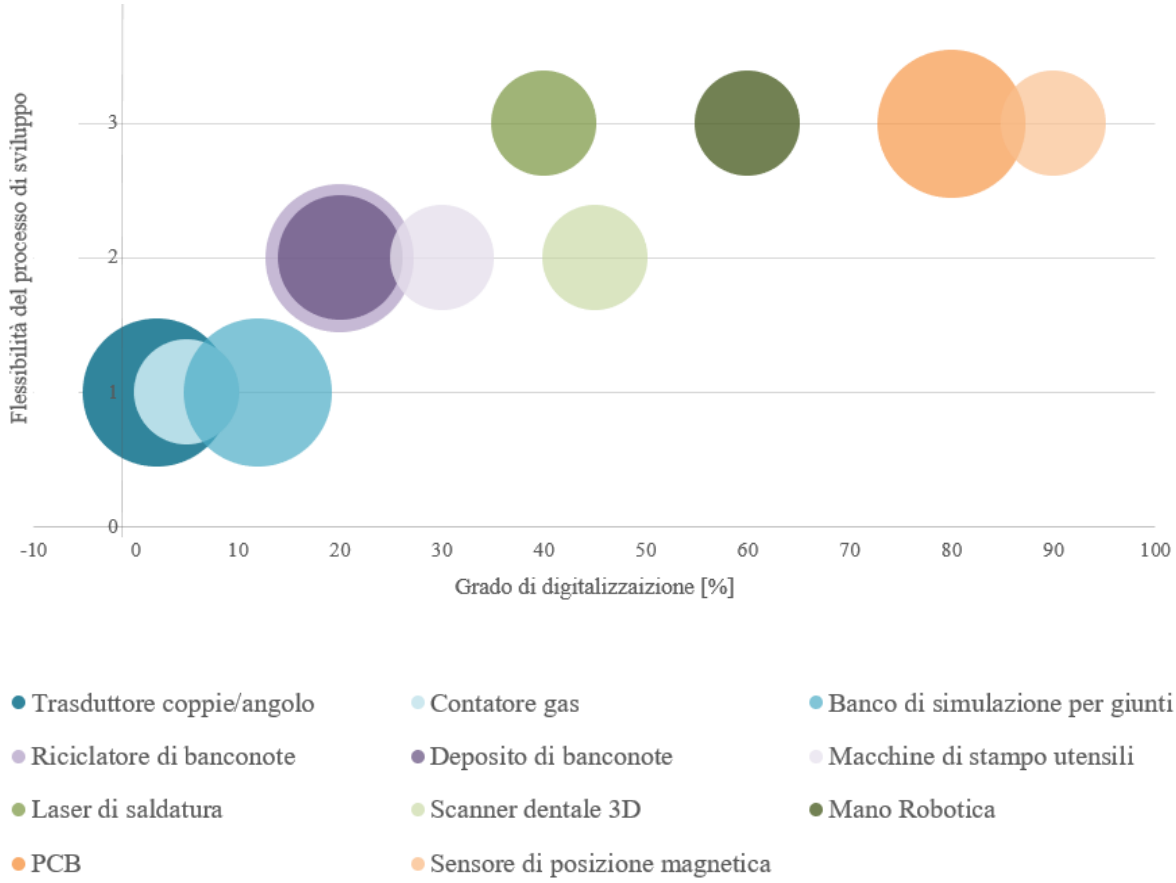

Figura 20: Flessibilità del processo di sviluppo vs. grado di digitalizzazione dei prodotti

Lo scopo del grafico è quello di evidenziare il legame tra il grado di digitalizzazione del prodotto e la flessibilità del processo di sviluppo emerso nel corso dell'analisi delle interviste, verificando anche se il grado di digitalizzazione impatti o meno sulla durata degli sprint.

Dalla concentrazione di bolle più grandi verso sinistra, la durata degli sprint sembrerebbe essere lievemente più dilatata per prodotti che presentano un minor grado di digitalizzazione, anche se questo non è sempre valido: sia per i contatori del gas sia per le macchine utensili, che presentano un alto contenuto fisico meccanico, si lavora per sprint di 15 giorni, mentre per i PCB in cui il software ha un ruolo core, si lavora per sprint di 30 giorni. Dunque, si può identificare una certa tendenza ad allungare le tempistiche rispetto alle due settimane di sprint che si è soliti impiegare nello sviluppo software, soprattutto per prodotti meno digitali e quindi più meccanici, che necessitano di tempi maggiori per rilasciare delle porzioni incrementali di prodotto. Tuttavia, questa tendenza dipende pesantemente da come il team di sviluppo riesce a organizzare i lavori.

Passando ad analizzare il primo legame, i prodotti che presentano un minor grado di digitalizzazione, soprattutto quelli appartenenti alla Categoria 1, risultano ancora altamente vincolati da una metodologia di sviluppo piuttosto tradizionale: l'Agile è introdotto nelle prime fasi di pre-studio, quando ancora non è definito un design, ma man mano che si procede con lo sviluppo prevale un flusso di lavoro Waterfall, intervallato da gate intermedi. Invece, lo sviluppo dei tre prodotti in Categoria 2 e dello scanner in Categoria 3 deve passare anch'esso tramite dei gate che determinano se il progetto può continuare, o semplicemente deve seguire delle linee guida con degli step ben precisi. Tuttavia, si lavora in maniera agile dall'inizio alla fine del progetto, fino alla fase di "go" in produzione. In particolare, per quanto riguarda lo scanner, nonostante abbia un alto contenuto digitale e il software giochi un ruolo importante, è un prodotto che deve ottenere delle certificazioni ben precise ed essere conforme delle normative molto più stringenti per essere commercializzato. Per questo motivo il processo di sviluppo che deve seguire è ancora molto vincolato in alcune fasi.
Infine, ci sono i prodotti in Categoria 3 e 4 che seguono uno sviluppo Agile, senza particolari vincoli. Si tratta comunque di prodotti altamente innovativi e con un forte contenuto digitale, soprattutto per i prodotti in Categoria 4 in cui la componentistica software è prevalente rispetto all'hardware.

Quindi, dal grafico in Figura 20 e rispetto a quanto descritto, risulta evidente un andamento nei dati raccolti, un legame tra il grado di digitalizzazione dei prodotti e la flessibilità nel processo di sviluppo, ovvero il grado con cui le aziende riescono a implementare un Agile più o meno spinto in termini di sviluppo incrementale e iterativo. Questa tendenza può essere spiegata dal fatto che le aziende più manifatturiere, che producono prodotti altamente

meccanici, hanno alcuni vincoli progettuali che di fatto limitano la trasformazione in toto del processo di sviluppo, trovandosi ancora fortemente legate ad alcuni aspetti tradizionali.

Tuttavia, è bene sottolineare che un forte grado di digitalizzazione del prodotto non è un requisito fondamentale per lavorare in modo Agile ed essere estremamente flessibili, poiché questo andamento in realtà è frutto di una serie di fattori che entrano in gioco: in primo luogo, l'Agile è un approccio ancora giovane e poco maturo per quanto riguarda il contesto di sviluppo hardware, soprattutto per il paese italiano, dunque le realtà aziendali devono ancora adattarsi e trovare il giusto fit tra le pratiche di sviluppo. Sicuramente una vicinanza più stretta al mondo software facilita le imprese ad avvicinarsi a uno sviluppo puramente agile (di cui hanno già esperienza) e trasferirlo al contesto hardware.

Per concludere, si vuole analizzare un ultimo aspetto che riguarda la mera traduzione delle pratiche al contesto hardware: la cultura aziendale accoglie le logiche agili poco per volta, partendo da progetti pilota finalizzati a testare e trovare un approccio su misura per l'organizzazione, per poi scalarlo internamente. Un adattamento delle pratiche è obbligatorio e non può avvenire un semplice trasferimento del Manifesto dal contesto software a quello hardware. D'altro canto, l'Agile non può essere applicato con tanti sconti e dunque l'adattamento non deve essere estremizzato, altrimenti si tratterebbe di seguire semplicemente alcuni buoni principi di lavoro e non una filosofia vera e propria.

A tal proposito, indagando ancora più a fondo nelle trascrizioni delle interviste, è emerso un aspetto rilevante che costituisce un vincolo al successo dell'Agile all'interno delle organizzazioni: il **mindset**. Con questo termine si generalizzano i problemi legati alla "mentalità" (di cui già si era discusso nel paragrafo *2.3.4 Sfide nell'applicazione*) e riguarda la difficoltà di stabilire una mentalità Agile sia nei singoli individui che nelle aziende (intese come un insieme di individui) che hanno sempre lavorato seguendo un piano, con logiche ben precise. Infatti, tra le problematiche più impattanti che si sono manifestate nel corso delle interviste rientra proprio la difficoltà di cambiare il metodo di lavoro da parte di colleghi, di responsabili oppure dall'intervistato stesso e le inerzie interne non facilitano l'applicazione di un approccio che responsabilizza, dà più libertà ai membri del team di sviluppo e si fonda sulla collaborazione reciproca.

In tal senso, il caso dell'azienda 5 merita un'analisi più approfondita: è stato proprio l'intervistato a evidenziare l'aspetto del mindset, spingendo a una lettura più profonda tra le righe di tutte le trascrizioni delle altre interviste. L'azienda sviluppa, realizza e installa dei

macchinari industriali, ovvero prodotti altamente meccanici, tuttavia i team di sviluppo riescono ad applicare l'Agile nelle fasi tra un gate e l'altro rispettando tutte le sue pratiche in modo ottimale. Sono ancora costretti a lavorare in maniera ibrida dovendo ricevere l'approvazione per la fase di approvvigionamento dei materiali:

"*… abbiamo cercato di sovrapporre tutte le fasi eccetto quella dell'approvvigionamento: abbiamo il gate che ci dà l'approvazione per procedere con gli investimenti, ma prima di quel gate non siamo autorizzati. Questo è uno dei vincoli per cui c'è ancora del Waterfall nell'Agile. Io riesco a sovrapporre tutte le fasi di progettazione, validazione, messa in servizio, collaudo e industrializzazione, fatta eccezione per quella di analisi fattibilità che ci autorizza a spendere per poi realizzare i prototipi...*"

Il fatto curioso è che l'azienda lavora in maniera Agile lato hardware, ma non lato software. Infatti, l'intervistato ha affermato anche:

"*…Questo è dovuto a qualcosa che nell'agile è cruciale: le persone. Il dirigente che mi ha assunto crede molto in questo approccio e quindi ha spinto a portare l'Agile nell'hardware prima ancora del suo omologo nel software…*"

Il caso particolare dell'azienda 5 ha dato la spinta per analizzare gli altri casi da un punto di vista differente e arrivare al concetto di mindset. L'Agile (puro o ibrido) funziona se c'è un cambiamento di mentalità, se è accettato da tutti e da parte di tutti c'è la volontà di mettersi in gioco per cambiare le regole di lavoro utilizzate fino a quel momento.
In netto contrasto con l'esperienza dell'intervistato 5, l'azienda 1 è l'esempio di un aggiustamento forzato che si è cercato di attuare, causando però il fallimento del successo dell'Agile. L'approccio è stato introdotto molto velocemente e ha riscontrato ulteriori difficoltà quando l'azienda ha scelto di non investire più in risorse, riducendo il numero di Scrum Master. I pochi rimasti hanno dovuto gestire diversi progetti in parallelo con team molto numerosi, a danno di una buona organizzazione dei lavori. Anche la qualità delle cerimonie Scrum è venuta meno.

"*…Dopo un po' l'azienda ha tolto l'Agile sostenendo che non era efficacie, in realtà non era stato implementato in maniera corretta. Non hanno più investito negli Scrum Master,*

quindi ci siamo trovati a gestire in parallelo tanti team numerosi e la situazione è diventata insostenibile..."

Questo ha scoraggiato anche i progettisti che non vedevano più nell'Agile un buon approccio allo sviluppo prodotto, bensì un adattamento forzato. Il malcontento ha causato una reazione a catena che ha coinvolto la maggior parte degli individui nell'area di ricerca e sviluppo. Così come dichiarato anche da un altro intervistato che ha lavorato nella stessa unità:

"... una grande difficoltà è che il management ci deve credere. Invece nell'azienda in cui lavoravo, la metà del management ci credeva tantissimo (americano) mentre l'altra metà no (italiano). L'organizzazione deve essere convinta altrimenti diventa davvero faticoso..."

In conclusione, l'Agile con i suoi adattamenti rispetto al contesto di applicazione può funzionare se c'è cambiamento di mentalità nelle persone. Ci sono alcune linee di business dove per la loro natura e per anzianità del middle management è più complesso apportare tale cambiamento. Questo può valere soprattutto per i prodotti più tradizionali, con un alto contenuto fisico, in cui il legame con le tradizioni è ancora dominante e il processo di evoluzione a un approccio di sviluppo più dinamico e innovativo richiede delle tempistiche maggiori rispetto a prodotti più digitali e all'avanguardia. Per questi ultimi, infatti, è normale sopravvivere in ambienti che richiedono una forte proattività e flessibilità rispetto al contesto in cui operano, dunque lo sforzo di adattamento a un processo di sviluppo così rivoluzionario è minore.

7. CONCLUSIONI

Le aziende sono sempre più coinvolte dalle esigenze di essere celeri e proattive per rispondere ai continui cambiamenti del mercato e cogliere le disruption che emergono da diversi fronti. Per questo motivo hanno dovuto ripensare i propri processi di sviluppo e l'Agile si è rilevato un valido approccio di gestione, che possiede tutte le peculiarità per soddisfare questi requisiti e contrastare la mancanza di reattività dei metodi più tradizionali.

Nonostante il paradigma sia nato per essere applicato al contesto software, per le ragioni sopracitate sta contaminando anche i contesti di sviluppo hardware, con tutte le implicazioni e le sfide che ne derivano. Infatti, i principi del Manifesto non possono essere semplicemente trasferiti da un perimetro all'altro, ma richiedono degli adattamenti specifici per rispondere a una serie di vincoli legati prevalentemente alle proprietà fisiche dei prodotti. Inoltre, le principali difficoltà riscontrate nell'applicazione dell'Agile per i domini non software risultano rincarate dalla complessità dei prodotti, in particolare di quelli meccatronici che richiedono la coesistenza di più ambiti di sviluppo: meccanico, elettronico e software.

Per procedere all'analisi del grado di agilità che le aziende riescono a implementare nei loro processi di sviluppo e degli adattamenti che le stesse mettono in atto, il lavoro è seguito tramite delle interviste semi-strutturate rivolte a dieci progettisti esperti nel campo, coinvolgendo l'analisi del processo di sviluppo di prodotti meccatronici. Le indagini si sono orientate per rispondere a due questioni principali: la prima per testare se la prevalenza di un'architettura integrale piuttosto che modulare possa facilitare od ostacolare l'implementazione dell'Agile, nell'ipotesi che un'architettura modulare sia preferibile per lavorare in modo iterativo e incrementale. La seconda questione riguarda il grado di digitalizzazione dei prodotti, sotto l'ipotesi che riducendo il contenuto digitale ci si allontana dal contesto di sviluppo software, perdendo di flessibilità e lasciando spazio alla prevalenza di pratiche più tradizionali. Al contrario, la vicinanza al mondo software e quindi a un ambito che per sua natura risulta essere più dinamico, consentirebbe di adattarsi con meno difficoltà.

In linea generale, dai risultati è emersa una mancanza di uniformità nell'implementazione dell'Agile: ognuno cerca di tradurre al meglio i principi del Manifesto e adattarli alla propria realtà. Infatti, le pratiche che prevedono l'impegno di team multidisciplinari e dedicati ai

singoli progetti e il feedback continuo da parte dei clienti non sono rispettate da tutti e nella stessa maniera.

Da un'analisi di dettaglio sono state identificate tre diverse macro-configurazioni di sviluppo: un approccio più rigido, in cui l'Agile è integrato all'interno dei tradizionali modelli di Stage-Gate con la presenza di gate molto vincolanti, un approccio ibrido intermedio, in cui sono presenti degli istanti di verifica sotto forma di gate o milestones che consentono di indirizzare la corretta esecuzione dei lavori e introdurre momenti di feedback, e un approccio più flessibile, in cui si segue un Agile pressoché puro. Dunque, il paradigma è implementato nei processi in modo più o meno spinto, dando vita a diverse configurazioni di sviluppo che possono essere rispettivamente più o meno flessibili.

Per ciò che concerne gli sprint, si è registrato un lieve allungamento delle tempistiche rispetto alle due settimane che si è soliti usare per il software: questo è attribuibile al disallineamento tra i due mondi di sviluppo e alla necessità di dilatare le scadenze entro cui rilasciare delle porzioni incrementali di prodotto. Inoltre, a fronte del fatto che per prodotti tangibili è impossibile ottenere alla fine di ogni sprint una release dimostrabile e testabile, la Definition of Done ha subito una forte rivisitazione. Il risultato di uno sprint lato hardware assume quindi un carattere diverso in base allo stato di avanzamento del progetto, partendo da risultati molto teorici nelle fasi di ideazione iniziale, fino ad evolvere e arrivare alla realizzazione di prototipi o di porzioni di prodotto. Rimane comunque valida la regola di ottenere al termine di ogni sprint qualcosa che incrementi il valore del progetto.

L'architettura di prodotto è risultata una variabile determinante per stabilire sia quale scomposizione del progetto applicare per lavorare in termini Agile, quindi se suddividere il prodotto in moduli oppure se procedere allo sviluppo per diversi livelli di dettaglio, sia la facilità con cui si possono apportare modifiche sostanziali in fase di sviluppo. In particolare, per le architetture integrali sono state trovate alcune soluzioni al fine di applicare uno sviluppo in ottica incrementale, tuttavia, man mano che il prodotto evolve e diventa più maturo è complicato apportare delle modifiche sostanziali che potrebbero stravolgere l'intera struttura: il costo del cambiamento sarebbe troppo elevato, causando un blocco del design di prodotto.

Le dipendenze esterne permangono a prescindere dalla tipologia di prodotto sviluppato. Accordi e partnership strategiche sono rilevanti al fine di ridurre l'esposizione al rischio per le dipendenze da fornitori esterni. Tuttavia, non sono ancora state trovate delle soluzioni per

le dipendenze dovute agli enti certificatori, che continuano a vincolare il processo di sviluppo di determinati prodotti, soprattutto quelli medicali, e la possibilità di apportare modifiche ai manufatti una volta risultati idonei alle normative in vigore.

Un'analisi semiquantitativa ha permesso di dare evidenza alla relazione che lega il grado di digitalizzazione dei prodotti con la flessibilità dei processi implementati. In particolare, è risultato che i prodotti con un minor grado di digitalizzazione sono ancora vincolati da una metodologia di sviluppo piuttosto tradizionale, in cui l'Agile è introdotto nelle prime fasi quando ancora non è definito un design vero e proprio, ma procedendo con lo sviluppo prevale un flusso di lavoro Waterfall. Una flessibilità maggiore è attribuita allo sviluppo di prodotti che sono caratterizzati da configurazioni digitali intermedie, per i quali sono ancora presenti dei gate o delle linee guida da seguire. In ultimo, per prodotti altamente digitalizzati e molto vicini al contesto di sviluppo software, si segue uno sviluppo Agile senza particolari vincoli. Quindi si può affermare che le aziende più manifatturiere, che realizzano prodotti in cui prevale la componente tangibile, sono soggette ad alcuni vincoli di progettazione che limitano la trasformazione in toto del processo di sviluppo e si trovano ancora fortemente legate ad aspetti più tradizionali.

Infine, è emerso un ultimo aspetto che coinvolge la mera traduzione dei principi del Manifesto al contesto hardware messa in pratica dalle aziende. Infatti, per quanto un loro adattamento sia inevitabile non può essere estremizzato, altrimenti si tratterebbe di seguire semplicemente alcuni buoni principi di lavoro senza attuare un paradigma vero e proprio, perdendo la natura dell'Agile.

A tal proposito, è rilevante il concetto di "Mindset" con cui si indica la sfida di stabilire una mentalità Agile negli individui e che costituisce un vincolo al successo dell'implementazione di un approccio iterativo. Per quanto esuli dalle mere caratteristiche fisiche dei prodotti, è emerso che le inerzie interne, legate al cambiamento del metodo di lavoro da sempre utilizzato, sono più intense per i team che si occupano di sviluppare prodotti con un alto contenuto fisico. In questi casi il legame con le tradizioni è ancora dominante e il processo di evoluzione verso uno sviluppo più dinamico e innovativo richiede delle tempistiche maggiori. Al contrario, un atteggiamento favorevole ad accogliere l'Agile come paradigma di lavoro sarebbe dirompente per il suo successo.

Per concludere, l'Agile non consiste in una vera e propria sequenza di regole o metodi predefiniti da seguire alla lettera, ma si tratta di una filosofia organizzativa da implementare e coltivare internamente. Essere agili significa rispondere rapidamente ai cambiamenti, soddisfare dei requisiti che possono variare nel tempo, contemplare la collaborazione e la trasparenza interna e intraprendere un percorso di cambiamento sia a livello individuale che organizzativo. Nonostante la presenza di sfide legate alle peculiarità fisiche dei prodotti, con il tempo le organizzazioni stanno trovando il giusto fit di pratiche che consente loro di implementare un paradigma Agile ad hoc, costruito su misura. Questo può essere più o meno spinto, anche ibridato a modelli più tradizionali in modo da non azzerare del tutto il proprio patrimonio di conoscenze pregresse. Di fatto non è possibile rivoluzionare del tutto la natura delle aziende e dei prodotti sviluppati, ma, per quanto le sfide non siano trascurabili, con il giusto adattamento e la giusta mentalità è possibile diventare agili ed essere proattivi in un mondo che richiede sempre più celerità e innovazione.

BIBLIOGRAFIA

Abellàn, E. (2020). *Metodologia Agile: cos'è e che benefici apporta alle imprese?* Tratto da We are marketing - Global growth agents: https://www.wearemarketing.com/it/blog/metodologia-agile-scopri-cose-e-quali-benefici-puo-portare-alla-tua-azienda.html

Albers, A., Heimicke, J., Spadinger, M., Reiss, Breitschuh, J., N, . . . Marthalera, F. (2019). A systematic approach to situation-adequate mechatronic system development by ASD - Agile Systems Design. *Procedia CIRP, 84*, 1015 - 1022. doi:10.1016/j.procir.2019.03.312

Albers, A., Lohmeyer, Q., & Ebel, B. (2011). Dimensions of objectives in interdisciplinary product development projects. *ICED 11 - 18th International Conference on Engineering Design - Impacting Society Through Engineering Design*, 256-265.

Albers, A., Trost, S., Heimicke, J., & Spadinger, M. (2020). Alignment of the change to agile through method-supported evaluation of agile principles in physical product development. *Procedia CIRP, 91*, 600-614. doi:10.1016/j.procir.2020.02.218

Anteby, M., Lifshitz, H., & Eisenhardt, K. (2014). Primer: Qualitative Research in Strategic Management. *Strategic Management Journal*.

Atzberger, A., & Paetzold, K. (2019). Current Challenges of Agile Hardware Development: What are Still the Pain Points Nowadays? *Proceedings of the Design Society: International Conference on Engineering Design, 1*, 2209-2218. doi:10.1017/dsi.2019.227

Basias, N., & Yannis, P. (2018). Quantitative and Qualitative Research in Business & Technology: Justifying a Suitable Research Methodology. *Review of Integrative Business and Economics Research, 7*, p. 91-105.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). *Manifesto for Agile Software Development.* Tratto da http://agilemanifesto.org/

Böhmer, A., Beckmann, A., & Lindemann, U. (2015). *Open Innovation Ecosystem - Makerspaces within an Agile Innovation Process.*

Cantamessa, M., & Montagna, F. (2016). *Management of Innovation and Product Development: Integrating Business and Technological Perspectives.* Springer-Verlag London Ltd. doi:10.1007/978-1-4471-6723-5

Carbonell, P., & Rodriguez, A. (2006). Designing Teams for Speedy Product Development: The Moderating Effect of Technological Complexity. *Journal of Business Research, 59*(2), 225-232. doi:10.1016/j.jbusres.2005.08.002

Ciric, D., Lalic, B., Gracanin, D., Placic, I., & Zivlak, N. (2018). Agile Project Management in New Product Development and Innovation Processes: Challenges and Benefits Beyond Software Domain. In *2018 IEEE International Symposium on Innovation and Entrepreneurship (TEMS-ISIE)* (p. 1-9). doi:10.1109/TEMS-ISIE.2018.8478461

Conforto , E., & Amaral, D. (2015). Agile project management and stage-gate model - A hybrid framework for technology-based companies. *Journal of engineering and technology management, 40*, 1-14. doi:10.1016/j.jengtecman.2016.02.003

Conforto, E., Salum, F., Amaral, D., Silva, S., & Almeida, L. (2014). Can Agile Project Management Be Adopted by Industries Other than software development? *Project Management Journal, 45*. doi:10.1002/pmj.21410

Cooper, R. (1990). Stage-gate systems: a new tool for managing new products. *Business Horizons*. doi:10.1016/0007-6813(90)90040-I

Cooper, R. (2013). New products – What separates the winners from the losers and what drives success. In *The PDMA Handbook of New Product Development* (3 ed., p. 3-34). doi:10.1002/9780470172483.ch1

Cooper, R. (2016). Agile–Stage-Gate hybrids: The Next Stage for Product Development Blending Agile and Stage-Gate methods can provide flexibility, speed, and improved communication in new-product development. *Research technology management, 59*(1), 21. doi:10.1080/08956308.2016.1117317

Cooper, R., & Sommer, A. (2016). Agile-Stage-Gate: New idea-to-launch method for manufactured new products is faster, more responsive. *Industrial marketing management, 59*, 167-180. doi:10.1016/j.indmarman.2016.10.006

Cooper, R., & Sommer, A. (2018). Agile–Stage-Gate for Manufacturers: Changing the Way New Products Are DevelopedIntegrating Agile project. *Research Technology Management, 61*(2), 17-26. doi:10.1080/08956308.2018.1421380

Fiore, C. (2005). *Accelerated product development: combining lean and six sigma for peak performance.* New York: Productivity Press. doi:10.4324/9781482293807

Gartner, I. (s.d.). *Gartner Hype Cycle*. Tratto da Gartner: https://www.gartner.com/technology/research/methodologies/hype-cycle.jsp

Gioia, Dennis and Corley, Kevin and Hamilton, & Aimee. (2013). Seeking Qualitative Rigor in Inductive Research. *Organizational Research Methods, 16*, p. 15-31. doi:10.1177/1094428112452151

Hennink, M., Hutter, I., & Bailey, A. (2011). *Qualitative Research Methods.* London: SAGE Publications Inc. doi:10.1080/09581596.2011.565689

Ingrosso, A. (2020, Marzo 10). *I 4 valori del Manifesto Agile*. (A. Ingrosso, Produttore) Tratto da ALESSANDRO INGROSSO Agile Trainer & Coach: https://www.alessandroingrosso.com/2020/03/10/i-4-valori-del-manifesto-agile/

Janes, A., & Succi, G. (2012). The dark side of agile software development. *Proceedings of the ACM international symposium on New ideas, new paradigms, and reflections on programming and software*, (p. 215–228). Tucson, Arizona, USA. doi:10.1145/2384592.2384612

Lavecchia, V. (s.d.). *Caratteristiche, funzionamento e processi di Scrum*. Tratto da Informatica e Ingegneria Online: https://vitolavecchia.altervista.org/caratteristiche-funzionamento-e-processi-di-scrum/

Lavecchia, V. (s.d.). *Differenze tra la metodologia Waterfall e Scrum*. Tratto da Informatica e Ingegneria Online: https://vitolavecchia.altervista.org/differenze-tra-la-metodologia-waterfall-e-scrum/

Lehnen, J., Schmidt , T., & Herstatt, C. (2016). Bringing agile project management into lead user projects. *International Journal of Product Development, 213*(2), 212-232. doi:10.1504/IJPD.2016.078867

Oversen, N., & Dowlen, C. (2012). The challenges of becoming agile - experiences from new product development in industry and design education. In *Proceedings of the*

14th International Conference on Engineering and Product Design Education: Design Education for Future Wellbeing (p. 9-14). Glasgow, UK: EPDE 2012.

Rubin, H., & Rubin, I. (2005). *Qualitative interviewing: The art of hearing data* (2nd ed.). Thousand Oaks, California. doi:10.4135/9781452226651

Sarantakos, S. (2005). *Social Research* (2nd ed.). Palgrave Macmillan Hampshire.

Schmidt, S., Atzberger, A., Gerling, C., Schrof, J., Weiss, S., & Paetzold, K. (2019). *Agile Development of Physical Products - An Empirical Study about Potentials, Transition and Applicability.* University of the German Federal Armed Forces Munich.

Schmidt, T., Weiss, S., & Paetzold, K. (2018b). EXPECTED VS. REAL EFFECTS OF AGILE DEVELOPMENT OF PHYSICAL PRODUCTS: APPORTIONING THE HYPE. *15th International Design Conference*, (p. 2121-2132). doi:10.21278/idc.2018.0198

Schmidt, T., Weiss, S., Paetzold, K., Stefan, & Kristin. (2018a). *Agile Development of Physical Products: An Empirical Study about Motivations, Potentials and Applicability.* University of the German Federal Armed Forces Munich.

Schrof, J., & Paetzold, K. (2019). Product modularization requirements in agile automotive product development. *30th Symposium Design for X*, (p. 171-182). doi:10.35199/dfx2019.15

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide.* Tratto da https://www.scrum.org/resources/scrum-guide

Sommer, A. F., Hedegaard, C., Dukovska-Popovska, I., & Steger-Jensen, K. (2015). Improved Product Development Performance through Agile/Stage-Gate Hybrids: The Next-Generation Stage-Gate Process? *Research-Technology Management, 58*, 34-45. doi:10.5437/08956308X5801236

Stare, A. (2014). Agile project management in product development projects. *Procedia - Social and Behavioral Sciences, 119*, 295 – 304. doi:10.1016/j.sbspro.2014.03.034

Thompson, K. (s.d.). *What is Agile Hardware Development?* Tratto da Agile for Hardware Product Development: https://www.agileforhardware.com/what-is-agile-hardware-development/

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research Policy, 24*(3), p. 419 - 440. doi:10.1016/0048-7333(94)00775-3

Wynn, D., & Clarkson, P. (2018). Process models in design and development. *Research in engineering design*. doi:10.17863/CAM.16827

RINGRAZIAMENTI

Vorrei dedicare questo spazio a chi ha contribuito alla realizzazione dell'elaborato e a chi mi ha accompagnata durante questo percorso universitario che sta volgendo al termine.

Un ringraziamento particolare va al mio relatore Marco Cantamessa, che mi ha seguita e indirizzata nella realizzazione della tesi di laurea, fin dalla scelta dell'argomento.

Ringrazio tutti coloro che si sono resi disponibili per condividere la propria esperienza lavorativa, dedicandomi un po' del loro tempo e fornendo un prezioso contributo senza il quale non sarebbe stato possibile realizzare questo lavoro.

Ringrazio infinitamente mia madre, mio padre e mio fratello per aver creduto in me e per avermi sempre sostenuta, restandomi accanto con l'infinita pazienza che li contraddistingue.

Un ringraziamento speciale va ai miei colleghi del Politecnico: Francesca, Valeria, Julia, Simona, Valentina, Michela, Elena e Alberto, le amicizie nate tra i banchi universitari, con cui ho condiviso molto e che mi hanno accompagnata in questo percorso. Ognuno a modo suo rimarrà un tassello importante. Grazie, perché non mi sono mai sentita sola.

Grazie a tutti i miei amici e alle persone a me care per avermi incoraggiata, ascoltata e per avermi regalato dei bellissimi momenti di spensieratezza.

Grazie di cuore a Simone, per essermi stato accanto, sempre.

Infine, l'ultimo ringraziamento va a me stessa, per i sacrifici e la determinazione che mi hanno permesso di arrivare fin qui, nella speranza di aver sempre la stessa forza e che questo traguardo possa essere l'inizio di una lunga e brillante carriera professionale e personale.

APPENDICE

Appendice A - Traccia dell'intervista

Azienda e intervistato: Informazioni generali sull'azienda e sulla mansione svolta dall'intervistato

1. *Di cosa si occupa la sua azienda o, più nel dettaglio, la business unit in cui lavora?*
2. *Qual è esattamente la sua mansione?*

Metodologia di sviluppo

3. *Ha già sentito parlare dell'Agile e ha mai applicato alcune delle sue pratiche?*
   SI: *quali pratiche?*
   NO: *quali metodologie di sviluppo applica all'interno della sua azienda?*
4. *Da quante risorse è composto il team? Quanti team lavorano in parallelo per un singolo progetto?*

Prodotto: informazioni sul prodotto oggetto dell'intervista. Lo scopo è quello di comprendere le tipologia di architettura e il grado di digitalizzazione dello stesso.

5. *Quali sono i prodotti di cui lei si occupa nello specifico?*
6. *In linea di massima, da quanti componenti è composto il prodotto (limitandosi a quelli che la sua azienda deve progettare, omettendo quindi la BOM dei componenti la cui progettazione è curata esternamente)?*
7. *Definirebbe l'architettura di prodotto modulare o integrale? Intendendo la prima come un'architettura nella quale i singoli moduli sono fortemente indipendenti tra loro (dal punto di vista delle funzionalità, della geometria e della tecnologia di riferimento), così che la loro progettazione può essere condotta in modo relativamente indipendente, e la seconda ovviamente il contrario (sottosistemi interdipendenti, che richiedono forti interazioni tra i rispettivi progettisti).*
8. *All'interno del prodotto sono presenti dei componenti elettronici e/o digitali? Quali? Qual è la loro funzione all'interno del prodotto?*

9. *Considerando il prodotto nel suo complesso e volendo suddividerlo fisicamente in componenti, qual è il contributo percentuale in termini funzionali fornito rispettivamente dai componenti meccanici, elettronici e software?*

Pianificazione dei progetti

10. *Solitamente i progetti sono pianificati fin dall'inizio in modo accurato e rigido, definendo fin da subito nel dettaglio le funzionalità del prodotto (A)? Oppure all'inizio sono definite le funzioni più importanti e poi il resto è dettagliato in corso d'opera con regolari aggiornamenti (B)?*
11. *(A) Come gestite i cambiamenti? Pensa che sarebbe possibile attuare uno sviluppo iterativo, suddividendo il prodotto in task da deliberare?*
12. *(B) Dunque, utilizzate un processo iterativo, concentrandovi poco per volta su un insieme di funzioni di prodotto. A quanto ammonta la durata degli sprint? È simile per tutti i prodotti oppure cambia?*
13. *Come monitorate lo stato di avanzamento dei lavori?*

Partecipazione del cliente

14. *C'è una partecipazione attiva da parte del cliente nella definizione dei requisiti e delle specifiche di prodotto? Quanto impatta il suo giudizio?*

Definition of Done e dipendenze esterne

Si Agile

15. *Nell'ambito di sviluppo software è molto semplice realizzare degli incrementi di prodotto potenzialmente spendibili, dopo tutto si tratta di consegnare stringhe di codice. Come adattate questo principio al contesto di sviluppo hardware? E quindi, come dividete il prodotto in diversi item da deliberare e testare progressivamente? Come definite uno "sprint fatto"?*
16. *Il fatto di dover acquistare dei materiali o dei componenti da fornitori esterni costituisce un vincolo? Vi ostacola nella realizzazione di item da testare?*
17. *Qualche suo fornitore o cliente applica questo approccio nello sviluppo prodotto? Questo condiziona i vostri rapporti? In che modo?*

No Agile

18. *I test sul prodotto avvengono solo alla fine della fase di sviluppo oppure sono effettuati anche durante le fasi precedenti?*

Difficoltà riscontrate

Si Agile

19. *Secondo lei quali sono le principali difficoltà legate all'implementazione dell'Agile nello sviluppo di prodotto fisici?*

No Agile

20. *Secondo lei è concretamente possibile implementare l'Agile nello sviluppo di prodotto fisici?*

21. *Ha qualche ultima considerazione da fare?*