Politecnico di Torino

Collegio di Ingegneria Gestionale – LM/31

Corso di Laurea Magistrale in Ingegneria Gestionale

Identificazione ed analisi dei fattori impattanti la performance delle risorse nei progetti Agile tramite revisione sistematica della letteratura

Relatore:

Prof. Alberto De Marco

Co-relatore:

Ing. Filippo Maria Ottaviani

Candidato:

Claudia Zingarello

Matricola 292895

Anno Accademico 2024/2025

Sommario

1. Abstract

Negli ultimi anni, la metodologia Agile è diventata uno degli approcci più diffusi per la gestione dei progetti, in particolare nel contesto dello sviluppo software. Questo approccio pone un forte accento sull'importanza del team, poiché un'elevata produttività del gruppo costituisce un elemento chiave per il successo di un progetto. La performance dei team che adottano la metodologia Agile è influenzata da molteplici variabili, rendendo necessaria un'analisi approfondita per comprendere quali elementi contribuiscano al miglioramento delle prestazioni e quali, invece, possano costituire ostacoli. Attraverso un processo di revisione sistematica della letteratura, questa tesi si propone di identificare e categorizzare i principali fattori che incidono sulla produttività dei team Agile. La ricerca ha evidenziato quattro macro-dimensioni di influenza: team, individuo, organizzazione e processo. Per ciascuna di queste categorie sono stati individuati i fattori più rilevanti ed esaminato il loro impatto sulla produttività. Comprendere tali dinamiche permette di individuare con maggiore precisione le aree su cui intervenire per ottimizzare le prestazioni del team e incrementare l'efficacia delle metodologie Agile.

2. Introduzione

La metodologia Agile è oggi uno degli approcci più utilizzati nella gestione dei progetti, soprattutto nell'ambito dell sviluppo software. La produttività dei team che la utilizzano è determinata da numerosi fattori, il cui impatto non è stato ancora analizzato in modo sistematico. La letteratura esistente, infatti, non offre un sistema strutturato di metriche per valutare questi elementi, rendendo difficile individuare con precisione le dinamiche che influenzano le prestazioni dei team Agile. Questa ricerca si propone di identificare e categorizzare i principali fattori che incidono sulla produttività nei team Agile. Attraverso un processo di revisione sistematica della letteratura, la ricerca analizza la frequenza con cui tali fattori sono citati, raggruppandoli in quattro macro-dimensioni: team, individuo, organizzazione e processo.

Per comprendere l'evoluzione delle metodologie di gestione dei progetti e contestualizzare l'approccio Agile, viene svolta un'analisi della nascita del Project Management, ripercorrendone lo sviluppo dall'antichità fino all'era moderna. Successivamente, viene approfondito l'approccio Waterfall, illustrandone le cinque fasi principali e i limiti che hanno portato alla diffusione di metodologie alternative.

Il capitolo prosegue con un approfondimento dell'approccio Agile, soffermandosi su due delle sue implementazioni più note: Extreme Programming (XP) e SCRUM, descrivendone i principi fondamentali e le pratiche operative. Infine, viene presentato un confronto tra il modello Waterfall e Agile, evidenziandone le differenze principali.

Questa panoramica fornirà le basi teoriche necessarie per comprendere l'evoluzione della gestione dei progetti e il contesto nel quale si inserisce Agile, permettendo di analizzare in modo più approfondito i fattori che influenzano la produttività dei team che adottano questo approccio.

2.1. La nascita del Project Management

Il presente paragrafo mira a tracciare la naturale evoluzione che il Project Management ha sperimentato nel corso del tempo, permettendo di ripercorrere le peculiari caratteristiche della gestione dei progetti e delle tecniche adottate, dall'antichità fino ad oggi.

2.1.1. Il Project Management nella storia antica

Se si concorda con la definizione fornita al Project Management Institute, secondo cui un progetto è un'attività temporanea che ha come scopo quello di creare un prodotto, un servizio o un risultato unico e la gestione dei progetti implica l'applicazione di conoscenze, competenze, strumenti e tecniche per soddisfare i requisiti specifici di un progetto, allora si può affermare che, fin dagli albori della civiltà, l'umanità abbia lavorato sulla progettazione e realizzazione di progetti (Project Management Institute, 2017). Le antiche città della Mesopotamia, le piramidi d'Egitto, la Grande Muraglia cinese, il Colosseo, Stonehenge; la storia è piena di esempi di eccezionali imprese ingegneristiche, campagne militari e altre imprese incredibili che testimoniano la grandezza e l'ingegno dell'uomo nel superare sfide e realizzare opere di grande portata.

Sfortunatamente, nonostante la presenza di tutti questi risultati monumentali, la documentazione e i registri storici relativi ai progetti precedenti sono spesso scarsi. Questa mancanza può essere attribuita a diversi motivi. In primo luogo, l'élite istruita era maggiormente concentrata sul risultato finale del progetto anziché sulla metodologia impiegata per la sua realizzazione. Inoltre, la responsabilità dell'esecuzione di tali progetti spettava principalmente agli artigiani, i quali non erano necessariamente formati o interessati a condividere i propri metodi con gli altri. Al contrario, in diversi di questi progetti, i dettagli dell'esecuzione erano mantenuti segreti all'interno di specifiche tribù o famiglie specializzate nell'artigianato e trasmessi da una generazione all'altra. All'interno del suo libro, "An Introduction to the History of Project Management: From the Earliest Times to AD1900",

Y. Chiu afferma che la mancanza di documentazione sulla gestione dei progetti può essere attribuita al fatto che il termine “progetto” non era diffuso nei testi antichi (Chiu, 2010). Questa carenza ha reso il campo della gestione dei progetti più difficile da tracciare e documentare rispetto ad altre discipline, quali l’architettura, la medicina, l’economia o la matematica.

2.1.2. Il Project Management nell’era moderna

Sebbene non si possa stabilire con certezza il momento preciso in cui il lavoro per progetti, così come lo intendiamo oggi, abbia avuto origine, le opinioni divergenti di diversi autori offrono un quadro affascinante sulla genesi della moderna gestione dei progetti.

All’interno del suo libro, “An Introduction to the History of Project Management: From the Earliest Times to AD1900”, Y. Chiu sostiene che Henri Fayol e Henry Gantt siano da considerarsi i pionieri del Project Management (Chiu, 2010). Sebbene questa affermazione possa suscitare opinioni discordanti, è ampiamente riconosciuto che entrambi abbiano contribuito in modo significativo alla definizione dei principi fondamentali della gestione dei progetti, introducendo approcci e strumenti che ancora oggi costituiscono il nucleo delle pratiche di project management moderno.

Henri Fayol (1841–1925), ingegnere francese attivo in una rilevante azienda siderurgica, ebbe un ruolo fondamentale nel decennio che precedette la prima guerra mondiale, contribuendo al riarmo dell'esercito francese. Durante il suo periodo alla guida dell’azienda, si interessò sempre più alle problematiche gestionali e, attraverso un'attenta osservazione, identificò cinque funzioni fondamentali svolte da ogni manager a vari livelli nella propria attività quotidiana: (1) pianificazione, (2) organizzazione, (3) coordinamento, (4) controllo e (5) comando. Fayol formulò anche 14 principi per guidare i manager nell'esecuzione efficace di queste funzioni. Molti sostengono che le teorie di Fayol non rappresentino le vere complessità affrontate dai manager quotidianamente, tuttavia il suo lavoro ha dato un

contributo significativo alla gestione dei progetti.

Il secondo pioniere del project management moderno, secondo Y. Chiu, è Henry Gantt (1861-1919), ingegnere e successivamente consulente aziendale statunitense. È noto principalmente per aver sviluppato il diagramma di Gantt, una tecnica di pianificazione ancora oggi largamente utilizzata e considerata indispensabile per ogni project manager. Il diagramma di Gantt ha avuto un grande impatto nella storia della gestione dei progetti perché consente di suddividere i grandi progetti in attività più piccole e gestibili, evidenziando inoltre le interdipendenze tra le varie fasi, come mostrato nell'esempio in *Figura 1*.

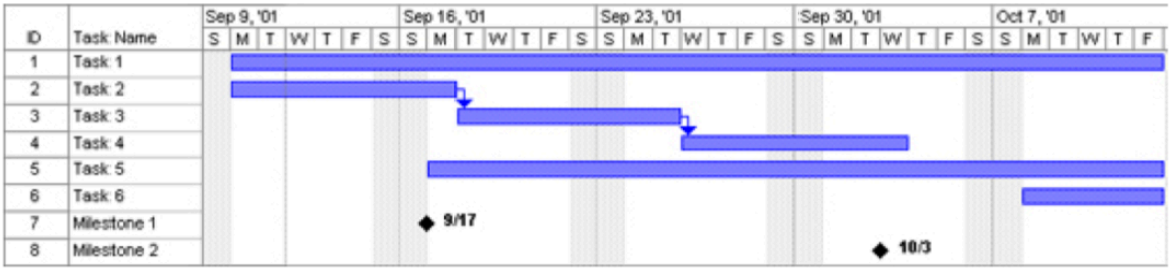

Figura 1: Un esempio di diagramma di Gantt che illustra la relazione esistente tra una serie di task

Sulla base del lavoro di Gantt sono nati successivamente molteplici concetti fondamentali ampiamente adottati nel project management, tra cui l'allocazione delle risorse e la struttura gerarchica delle attività di un progetto, nota come work breakdown structure (WBS).

Secondo quanto riportato da Snyder e Kline, l'avvio dell'era moderna del project management può essere fatto risalire al 1958, con l'introduzione dei metodi CPM e PERT (Kwak, 2005). Questi due metodi, sviluppati in modo indipendente, nacquero in contesti profondamente differenti: il PERT nell'ambito della Marina militare statunitense e il CPM nell'industria chimica. Nel 1958, infatti, la U.S. Navy avviò il progetto Polaris, volto alla realizzazione dei primi missili balistici lanciabili da sottomarini (SLBM) dotati di testate nucleari. La complessità e l'elevato grado di incertezza associati a questa iniziativa resero necessaria l'adozione di nuove tecniche di pianificazione, portando così allo sviluppo del Program Evaluation and Review Technique (PERT). Questo strumento si dimostrò

particolarmente utile per analizzare scenari alternativi e gestire le tempistiche progettuali in condizioni incerte, diventando una delle metodologie più diffuse per la valutazione di progetti complessi.

Parallelamente, il Critical Path Method (CPM) fu sviluppato dalla E.I. du Pont de Nemours Company per pianificare e gestire un articolato progetto di costruzione di un impianto chimico. Il CPM rispondeva all'esigenza di stimare con maggiore precisione i tempi e i costi del progetto, introducendo il concetto di percorso critico. Questo metodo permette di calcolare per ciascuna attività la prima data di inizio, la prima data di fine, l'ultima data di inizio e l'ultima data di fine per ogni attività, individuando il margine di flessibilità (float) e identificando come critiche le attività prive di margine temporale (float = 0). Il percorso critico rappresenta quindi la sequenza più lunga di attività interdipendenti che determina la durata minima complessiva del progetto. Grazie alla sua capacità di individuare le attività che, se ritardate, influenzerebbero l'intero progetto, il CPM si è rapidamente diffuso nel settore privato come strumento di pianificazione affidabile, fondato su solide basi matematiche.

2.1.3. PMBOK: Fondamenti di Project Management

Tra il 1969 e il 1972, la NASA (National Aeronautics and Space Administration) portò a termine sei missioni lunari di successo, dimostrando l'importanza di una gestione rigorosa e coordinata di progetti ad alta complessità. Sulla base dell'incredibile lavoro del governo degli Stati Uniti e dell'industria aerospaziale, nel 1987 il Project Management Institute (PMI) pubblicò un documento intitolato "The Project Management Body of Knowledge". Questo primo tentativo di standardizzazione fu successivamente ampliato e aggiornato nella prima edizione ufficiale della "A Guide to the Project Management Body of Knowledge (PMBOK® Guide)" del 1996 (Haugan, 2011).

Il PMBOK® Guide costituisce oggi uno dei riferimenti principali nella disciplina del project

management, offrendo un framework sistematico che include terminologie, processi e best practice riconosciute a livello globale.

Nel PMBOK® Guide, il processo di gestione del progetto si articola in cinque fasi, ognuna delle quali realizza obiettivi specifici del progetto (Project Management Institute, 2017). I cinque gruppi di processi sono:

1. *avvio*: in questo stadio avvengono tutte le attività necessarie per avviare un nuovo progetto o una fase dello stesso, incluso l'ottenimento dell'autorizzazione per avviare il progetto o la fase;
2. *pianificazione*: descrive i processi relativi alle modalità di sviluppo del progetto, la sua portata, la definizione degli obiettivi e il piano d'azione da seguire durante il progetto;
3. *esecuzione*: questa è la parte più importante di un progetto, in cui vengono attuati i processi necessari per portare a termine la consegna del prodotto e l'esecuzione delle attività specificate nel piano di progetto;
4. *monitoraggio e controllo*: attraverso la raccolta di dati sullo stato del progetto, vengono monitorati l'avanzamento e le prestazioni, i quali vengono confrontati con i piani stabiliti e, quando necessario, vengono effettuate delle modifiche correttive;
5. *chiusura*: processi per la finalizzazione del progetto, che permettono di chiudere tutti i compiti di tutti i gruppi di processo e di terminare il progetto in modo formale.

Il PMBOK include anche 10 Aree di Conoscenza, che rappresentano l'insieme di conoscenze e competenze necessarie per raggiungere specifici obiettivi (Project Management Institute, 2017). Di seguito sono elencate le Aree di Conoscenza descritte nella guida:

1. *Project Integration Management*: comprende i processi e le attività che assicurano una pianificazione, esecuzione e controllo adeguati del progetto, compresa la gestione formale delle modifiche al progetto. Come suggerisce il nome stesso, ogni

attività deve essere coordinata e integrata con le altre per raggiungere i risultati desiderati del progetto.

2. *Project Scope Management*: racchiude i processi essenziali per garantire che il progetto includa tutto il lavoro necessario per raggiungere gli obiettivi stabiliti e il controllo dell'ambito del progetto.
3. *Project Schedule Management*: descrive i processi relativi al completamento puntuale del progetto.
4. *Project Cost Management*: comprende i processi necessari per garantire che il progetto si svolga e si completi all'interno del budget approvato.
5. *Project Quality Management*: racchiude i processi che permettono di garantire che il progetto soddisfi adeguatamente gli obiettivi per cui è stato intrapreso, tra cui le specifiche e le richieste del cliente.
6. *Project Resource Management*: comprende i processi necessari per garantire l'efficiente utilizzo delle risorse umane, materiali e finanziarie del progetto.
7. *Project Communications Management*: descrive i processi relativi ai meccanismi di comunicazione di un progetto e riguarda la generazione, la raccolta, la diffusione, l'archiviazione e la disposizione finale delle informazioni di progetto in modo tempestivo e appropriato.
8. *Project Risk Management*: comprende i processi necessari per l'identificazione e la valutazione dei rischi; prevede anche la risposta ai rischi, i quali, qualora non venissero gestiti, possono provocare ritardi o sforamenti dei costi.
9. *Project Procurement Management*: include tutti i processi che si occupano dell'acquisizione di beni e servizi al fine di raggiungere gli obiettivi del progetto, attraverso l'utilizzo di commesse e contratti stipulati con fornitori esterni.
10. *Project Stakeholder Management*: comprende i processi necessari per individuare

tutti gli stakeholder che potrebbero avere un'influenza sul progetto o essere influenzati da esso. Inoltre, comprende l'analisi delle aspettative delle parti interessate e il loro impatto sul progetto, lo sviluppo di strategie di gestione adeguate e il coinvolgimento efficace delle parti interessate nelle decisioni e nell'esecuzione del progetto.

2.2. L'approccio Waterfall

La metodologia Waterfall, nota anche come modello a cascata, descritta formalmente per la prima volta da Winston W. Royce in un articolo del 1970 (Royce, 1987), è considerata il modello tradizionale per eccellenza. Questa metodologia consiste in un processo di sviluppo sequenziale che si snoda come una cascata attraverso tutte le fasi di un progetto, in cui l'output di ogni fase diventa l'input della fase successiva. Si tratta di un modello iterativo in quanto una fase non può iniziare finché quella precedente non è stata conclusa. L'approccio Waterfall ha origine nelle industrie manifatturiere ed edili, dove si pone particolare attenzione nelle fasi di analisi e progettazione poichè eventuali modifiche al prodotto durante la fase di realizzazione comportano costi molto elevati.

2.2.1. Le cinque fasi del metodo Waterfall

Il numero di fasi può variare a seconda del progetto. La *Figura* 2 mostra uno dei possibili schemi di modello a cascata (Bassil, 2012), comprendente cinque fasi di seguito dettagliate. La fase di analisi consiste nella raccolta di tutti i requisiti, sia funzionali che non funzionali, del progetto. Relativamente ai requisiti funzionali, l'interazione tra l'utente e il software viene espressa attraverso dei casi d'uso. I requisiti funzionali includono l'obiettivo, il contesto d'uso, le caratteristiche del sistema e le funzionalità richieste, le specifiche per le interfacce e i database e le esigenze degli utenti finali.

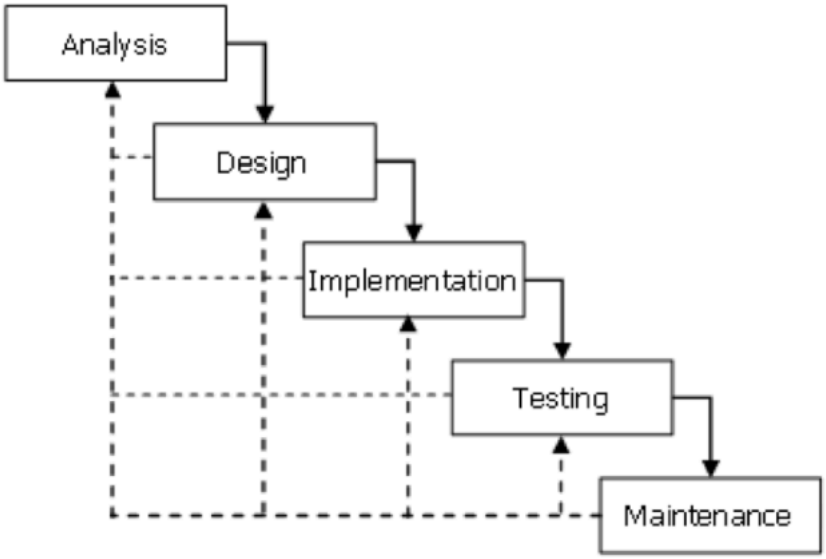

Figura 2: Le fasi del metodo Waterfall

Diversamente dai requisiti funzionali, quelli non funzionali definiscono i vincoli e le limitazioni del software e riguardano aspetti come scalabilità, disponibilità, affidabilità e prestazioni.

La fase di progettazione definisce il risultato finale e le modalità per realizzarlo sulla base dei requisiti precedentemente raccolti. Questa fase si basa su un piano dettagliato che solitamente subisce poche modifiche durante il corso del progetto, in quanto non è prevista una rielaborazione successiva. I progettisti e sviluppatori software si occupano di definire l'architettura, gli algoritmi, le strutture dati e le interfacce utente del sistema.

L'implementazione è la fase in cui i requisiti e le specifiche di progettazione vengono tradotti in un programma concreto all'interno di un ambiente di produzione. Durante la fase di implementazione, gli elementi del progetto sono realizzati uno dopo l'altro e ogni membro del team completa i compiti assegnati in sequenza prima di passarli al successivo individuo o gruppo.

Nella fase successiva vengono eseguiti i test su tutti i componenti del sistema al fine di verificare il corretto funzionamento di ogni parte in relazione alle altre e che il software sia

conforme alle specifiche. In questa fase è possibile individuare e risolvere eventuali bug.

L'ultima fase, ovvero quella di manutenzione, consiste nella modifica del prodotto dopo il suo rilascio con l'obiettivo di correggere errori e migliorare le prestazioni. Attraverso la manutenzione, è possibile adattare il software all'evoluzione dei requisiti al fine di garantire il suo corretto funzionamento nel tempo.

2.3. L'approccio Agile

Nei primi anni dello sviluppo software, in assenza di metodologie specifiche per il settore, si scelse di adottare il modello Waterfall, considerato affidabile e strutturato. Tuttavia, con il tempo, le sue limitazioni spinsero gli sviluppatori a ricercare alternative più flessibili che si adattassero meglio alle esigenze del settore. Questo portò alla nascita dei metodi Agile, caratterizzati da cicli iterativi brevi e dalla consegna progressiva di parti del progetto. Inizialmente, questi approcci non erano identificati con un nome unificato, fino a quando, nel febbraio 2001, diciassette esperti del settore, tra cui Kent Beck, Martin Fowler, Alistair Cockburn e Jeff Sutherland, si riunirono a Snowbird, nello Utah, per definire valori e principi comuni. Dopo aver valutato diverse opzioni, scelsero il termine "Agile", ritenendolo più rappresentativo e formularono il Manifesto Agile, una dichiarazione chiara e concisa dei principi fondamentali di questa metodologia (Fowler and Highsmith, 2001).

Il Manifesto enfatizza quattro valori fondamentali (Beck *et al.*, 2001):

1. Persone e comunicazione prima di procedure e strumenti
2. Software funzionante invece di documentazione dettagliata
3. Lavorare a stretto contatto con il cliente piuttosto che focalizzarsi sui contratti
4. Adattabilità ai cambiamenti anziché attenersi a piani fissi

L'approccio Agile affronta l'imprevedibilità dello sviluppo software adottando un metodo incrementale, riducendo al minimo attività come pianificazione e stime rigide. Il rilascio frequente di funzionalità, unito a un'interazione costante con il cliente, consente di

accelerare il time-to-market e garantire un adattamento rapido ai cambiamenti.

2.3.1. Extreme Programming (XP)

Extreme Programming (XP) è una metodologia Agile che si concentra principalmente sulle pratiche di sviluppo software piuttosto che sulla gestione del progetto (Kumar, Gupta and Singh, 2014). A differenza di altri approcci, XP può essere adottato integralmente oppure solo in parte, a seconda delle esigenze organizzative. Si tratta di una metodologia ampiamente riconosciuta e praticata, ma anche oggetto di opinioni contrastanti a causa delle sue pratiche rigorose. Alla base di XP vi sono quattro valori fondamentali: comunicazione, feedback, semplicità e coraggio, che si concretizzano in pratiche operative orientate a incrementare la produttività del team e a garantire il rilascio frequente di software funzionante, generalmente ogni una o tre settimane. In XP, il codice viene sempre scritto da almeno due programmatori che lavorano simultaneamente: tale pratica prende il nome di programmazione in coppia (pair programming). XP promuove pratiche come il refactoring continuo, ovvero la revisione incrementale del codice volta a migliorarne la struttura interna senza modificarne il comportamento esterno, e l'integrazione continua, che prevede la verifica e l'inserimento del nuovo codice all'interno della base. Uno degli aspetti centrali di questa metodologia è il coinvolgimento continuo e diretto del cliente, che partecipa attivamente al processo di sviluppo. In particolare, il cliente collabora con il team nella definizione delle "User Stories", brevi descrizioni dei requisiti del software espresse in linguaggio naturale e comprensibile che fungono da base per la pianificazione e la realizzazione del software. Il team di sviluppo valuta e organizza queste user stories per priorità, trasformandole iterativamente in funzionalità operative. Ogni iterazione prevede una fase di pianificazione, lo sviluppo vero e proprio e infine un test di accettazione nel quale si verifica se le funzionalità implementate soddisfano le richieste del cliente. Eventuali difetti emersi in fase di test vengono registrati e corretti nella successiva iterazione, garantendo un

controllo costante sulla qualità del software e una rapida reattività alle modifiche. L'intero processo prosegue finché tutte le user story sono state soddisfatte e non emergono nuove esigenze: a quel punto, il progetto può considerarsi concluso.

2.3.2. SCRUM

Tra i framework Agile, Scrum è attualmente il più popolare grazie al suo approccio intuitivo e facilmente implementabile. Jeff Sutherland, uno dei creatori del framework, definisce Scrum come "un processo di sviluppo software Agile progettato per aggiungere energia, attenzione, chiarezza e trasparenza ai team di progetto che sviluppano sistemi software" (Sutherland *et al.*, 2007). Nel framework Scrum, la volatilità dei requisiti viene vista come un'opportunità piuttosto che un ostacolo, riconoscendo che le esigenze del cliente possono evolvere nel tempo. Grazie a questo approccio adattivo, Scrum è particolarmente vantaggioso in quei progetti dove la pianificazione tradizionale risulterebbe inefficace. Inoltre, affinché il team possa prendere decisioni tempestive basate su eventi concreti e informazioni aggiornate, è fondamentale una stretta collaborazione tra i membri del team unita a una solida capacità di autogestione.

Il nucleo del framework Scrum è composto da 3 ruoli, 3 artefatti e 5 eventi.

Lo Scrum Core Team è rappresentato da un team compatto di professionisti, composto da uno Scrum Master, un Product Owner e dai Developer. Questa squadra lavora senza gerarchie o suddivisioni interne, focalizzandosi su un unico obiettivo per volta, ovvero il Product Goal. Grazie alla sua natura cross-funzionale, il team dispone di tutte le capacità e conoscenze necessarie per generare valore a ogni Sprint, mentre l'autogestione permette ai membri di organizzare autonomamente ruoli, compiti e tempistiche. Oltre a sviluppare il prodotto, il team si occupa di attività fondamentali come test, manutenzione, ricerca e interazione con gli stakeholder. L'organizzazione garantisce l'autonomia necessaria affinché il team possa gestire il proprio lavoro in modo efficiente. La suddivisione del lavoro per

Sprint consente di mantenere coerenza, ritmo sostenibile e concentrazione sugli obiettivi.

Di seguito sono descritti i tre ruoli principali in Scrum:

- *Developer*: si occupano della realizzazione dell'Increment a ogni Sprint. Devono organizzare il lavoro attraverso lo Sprint Backlog, monitorare lo Sprint Burndown Chart, garantire la qualità rispettando la Definition of Done. I Developer aggiornano quotidianamente la pianificazione per raggiungere gli obiettivi dello Sprint, collaborano per risolvere eventuali criticità riscontrate dai singoli membri e partecipano a revisioni e retrospettive per individuare i punti di miglioramento del processo.
- *Product Owner*: ha il compito di massimizzare il valore del prodotto sviluppato. Definisce il Product Goal e gestisce il Product Backlog, comunicando e organizzando gli elementi da sviluppare. Inoltre, si assicura che le esigenze e le priorità degli stakeholder siano tradotte in requisiti chiari e comprensibili. Può delegare alcune attività, ma resta il principale responsabile. Per garantire che il Product Owner operi efficacemente all'interno dell'organizzazione, le sue decisioni devono essere riconosciute e seguite da tutti. Tali decisioni si riflettono negli elementi e nelle priorità del Product Backlog, oltre a emergere chiaramente durante l'analisi dell'Increment nella Sprint Review.
- *Scrum Master*: ha il ruolo di supportare l'adozione di Scrum, assicurando che il team e l'organizzazione comprendano e applichino correttamente il framework. In qualità di coach, favorisce il miglioramento continuo delle pratiche di lavoro, aiuta i Developer a lavorare in modo autonomo e rimuove gli ostacoli che potrebbero rallentare il progresso del team, garantendo al contempo che tutti gli eventi Scrum si svolgano in modo efficace, costruttivo e nei tempi previsti. Collabora con il Product Owner per definire strategie efficaci di gestione del Backlog e di definizione del

Product Goal. Aiuta il team nella creazione di Increment di valore. Se necessario, facilita la comunicazione tra il team e gli stakeholder per migliorare la collaborazione e aiuta i dipendenti e gli stakeholder a sviluppare una mentalità empirica per affrontare le sfide complesse.

Nel framework Scrum, il tempo è considerato uno dei vincoli più importanti nella gestione di un progetto. Per affrontare questo vincolo, Scrum introduce un concetto noto come "Time-boxing", che consiste nell'assegnare un periodo di tempo specifico a ciascun processo e attività all'interno di un progetto Scrum. Gli eventi sono momenti chiave all'interno del framework Scrum poichè consentono l'ispezione e l'adattamento, promuovendo la trasparenza richiesta. L'organizzazione di questi eventi mira a creare regolarità ed evitare riunioni aggiuntive non previste dal framework.

Di seguito sono elencati i principali eventi Scrum:

- *Sprint*: fulcro del framework Scrum, è un ciclo di lavoro a durata fissa in cui vengono svolte tutte le attività necessarie al raggiungimento del Product Goal. Durante lo Sprint, la qualità deve essere preservata, il Product Backlog aggiornato a seconda delle necessità e l'ambito o "scope" può essere ridefinito con il Product Owner. Sprint più brevi favoriscono l'adattamento e riducono i rischi legati a costi e impegni. Il progresso può essere monitorato con strumenti specifici, come il Burndown Chart, che tuttavia non sostituiscono l'approccio empirico. Se lo Sprint Goal diventa obsoleto, solo il Product Owner ha l'autorità di annullarlo.
- *Sprint Planning*: avvia lo Sprint e definisce le attività da svolgere, coinvolgendo tutto lo Scrum Team. Il Product Owner si assicura che i membri del team siano preparati a discutere gli elementi prioritari del Product Backlog e il loro allineamento con il Product Goal. Il processo si articola in tre fasi: stabilire il valore dello Sprint, selezionare le attività e pianificare il loro svolgimento. I Developer suddividono il

lavoro in unità più gestibili, mantenendo piena autonomia nell'organizzazione. Lo Sprint Planning può durare fino a otto ore per Sprint mensili, riducendosi per Sprint più brevi.

- *Daily Scrum*: breve incontro quotidiano tra i Developer dello Scrum Team per monitorare i progressi dello Sprint e adattare lo Sprint Backlog in base alle necessità. Dura 15 minuti e si svolge sempre nello stesso momento e luogo. Lo Scrum Master o il Product Owner sono chiamati a partecipare come Developer nel caso in cui lavorino su elementi dello Sprint Backlog. I Developer hanno la libertà di adottare qualsiasi struttura o metodologia desiderino, purché il Daily Scrum rimanga incentrato sui progressi verso lo Sprint Goal e consenta di definire un piano concreto per il giorno successivo. Questo approccio favorisce la concentrazione e rafforza l'autogestione del team. Inoltre, il Daily Scrum facilita la comunicazione, aiuta a individuare e risolvere eventuali ostacoli, accelera il processo decisionale e riduce la necessità di riunioni aggiuntive. Il team può comunque rivedere il piano durante la giornata, se necessario, o discutere più dettagliatamente sull'adattamento del lavoro rimanente.
- *Sprint Review*: permette di valutare il lavoro svolto durante lo Sprint e discutere i passi successivi. Il team presenta i risultati agli stakeholder, analizza i progressi e, se necessario, aggiorna il Product Backlog. L'evento non andrebbe ridotto ad una mera presentazione, poiché è a tutti gli effetti una sessione di lavoro. Ha una durata massima di quattro ore per Sprint di un mese, ma è più breve per Sprint minori.
- *Sprint Retrospective*: è una riunione che si tiene alla fine di ogni iterazione per analizzare cosa ha funzionato e cosa no nello Sprint appena concluso e individuare le aree di miglioramento. Sulla base di questa analisi, si identificano i miglioramenti più importanti che vengono implementati il prima possibile, con la possibilità di

integrarli nello Sprint Backlog successivo. L’evento chiude lo Sprint e dura al massimo tre ore per Sprint di un mese.

Le metodologie agili si distinguono per la produzione limitata di documentazione. Nel framework Scrum, i documenti sono comunemente denominati "artefatti" per convenzione. Gli artefatti rappresentano il lavoro svolto o il valore generato e sono pensati per garantire la massima trasparenza delle informazioni chiave. Grazie a questa chiarezza, chi li analizza può prendere decisioni informate per eventuali adattamenti.

Gli artefatti sono elencati di seguito:

- *Product Backlog:* è una lista dinamica e ordinata di elementi necessari per sviluppare il prodotto finale. Rappresenta la sola fonte di attività per lo Scrum Team. Gli elementi considerati "pronti" per essere completati in uno Sprint vengono selezionati durante lo Sprint Planning, spesso dopo un processo di Refinement. Il Refinement è un’attività continua che suddivide e dettaglia gli elementi del Product Backlog, aggiungendo informazioni come descrizione, priorità e dimensione. La complessità di questi elementi varia a seconda del contesto lavorativo. I Developer, responsabili della stima delle dimensioni, possono essere supportati dal Product Owner, che li aiuta a comprendere meglio i requisiti e a individuare compromessi. Inoltre, il Product Owner può aggiungere nuovi elementi al Product Backlog o modificarne l’ordine a seconda delle esigenze. All’interno del Product Backlog si trova il Product Goal, ovvero l'obiettivo a lungo termine, che fornisce una direzione strategica per lo Scrum Team. Il Product Backlog si evolve per definire cosa è necessario per raggiungere tale obiettivo. Prima di affrontare un nuovo Product Goal, è necessario completare o eventualmente rivedere quello attuale, garantendo così un processo di sviluppo orientato al valore.
- *Sprint Backlog:* include lo Sprint Goal (l’obiettivo dello Sprint), l'elenco degli

elementi del Product Backlog selezionati (cosa sarà realizzato) e un piano dettagliato per completare l'Increment (come verrà sviluppato). È uno strumento pensato dai Developer per i Developer e fornisce una rappresentazione visibile e aggiornata in tempo reale del lavoro che dovrà essere svolto durante lo Sprint. Per tale motivo, lo Sprint Backlog viene affinato con l'evolversi della comprensione del lavoro, mantenendo un livello di dettaglio sufficiente per monitorare i progressi nel Daily Scrum.

- *Increment:* rappresenta un passo tangibile verso il Product Goal. Si aggiunge in modo cumulativo agli Increment precedenti, verificando attentamente che questi ultimi siano compatibili insieme. Ogni Increment deve essere funzionante e utilizzabile e più Increment possono essere sviluppati all'interno dello stesso Sprint. Tutti gli Increment vengono presentati durante la Sprint Review, tuttavia, questo passaggio non è obbligatorio per il rilascio di valore, poiché un Increment può essere fornito agli stakeholder anche prima del termine dello Sprint. Un lavoro è considerato parte di un Increment solo se rispetta la "Definition of Done", che rappresenta la descrizione formale dello stato in cui l'incremento deve trovarsi per essere considerato completo e soddisfare le metriche di qualità richieste.

2.4. Waterfall e Agile a confronto

Uno dei principali punti di forza della metodologia Waterfall risiede nella sua capacità di gestire e controllare in maniera efficace risorse come tempo, personale e budget. Questo modello si rivela particolarmente adatto nei contesti industriali in cui è necessario un flusso di lavoro strutturato rigidamente, in cui i cambiamenti successivi alla fase di codifica sono difficilmente sostenibili. Un altro aspetto favorevole è la produzione costante di documentazione in ogni fase del processo che facilita la comprensione del progetto e ne supporta la gestione.

Tuttavia, il modello Waterfall presenta anche numerosi limiti. Il suo carattere sequenziale impedisce di valutare i risultati di una fase prima di procedere con la successiva e non consente di ritornare a fasi precedenti per apportare modifiche, rendendo ogni passaggio definitivo. Questo può creare difficoltà quando il cliente non ha una visione chiara dei propri requisiti iniziali, rendendo complicata la gestione di eventuali cambiamenti in corso d'opera. Inoltre, scarsa integrazione tra i vari moduli del software e la netta separazione tra processi e dati limitano la flessibilità del modello, rendendolo inadatto alla programmazione orientata agli oggetti. L'elevata dipendenza tra i componenti implica che una modifica ai dati possa richiedere una completa rielaborazione del sistema, compromettendo la possibilità di riutilizzare il codice in contesti differenti.

In contrapposizione, l'approccio Agile si basa su cicli iterativi e incrementali che consentono una gestione più efficace dei cambiamenti nei requisiti utente. La natura flessibile di questo metodo permette di intervenire rapidamente sul codice esistente, rimuovendo le parti obsolete e adattando la progettazione alle nuove esigenze. L'approccio Agile consente al team di analizzare i risultati raggiunti attraverso le retrospettive di iterazione, promuovendo un apprendimento costante e il miglioramento continuo. Questa pratica favorisce anche l'integrazione flessibile di nuove funzionalità nel corso dello sviluppo. Un ulteriore vantaggio risiede nella modularizzazione del software: grazie all'incapsulamento, gli oggetti risultano indipendenti tra loro, semplificando la manutenzione, l'aggiornamento e la riusabilità del codice.

Tuttavia, anche questo approccio non è esente da svantaggi. Una delle principali criticità è la difficoltà nello stabilire un punto di chiusura del progetto, poiché il team può essere spinto a continuare le iterazioni nella ricerca di una soluzione ottimale. Inoltre, può rivelarsi complesso e richiede competenze specifiche che non sempre sono facilmente reperibili.

3. Revisione preliminare della letteratura

Si è scelto di utilizzare SCOPUS come fonte di ricerca principale per l'analisi preliminare della letteratura poiché offre un'ampia varietà di articoli peer-reviewed e riviste scientifiche di alta qualità. A differenza di altri database, Scopus è il database più utilizzato (Strozzi *et al.*, 2017) con oltre il 60% della copertura rispetto a Web of Science (Zhao and Strotmann, 2015) ed è considerato più affidabile di Google Scholar che include anche fonti accademiche non tradizionali (Noruzi, 2005).

La ricerca ha escluso tutti i tipi di documento diversi da articoli (ar), review (re), capitoli di libro (ch), conference paper (cp), libri (bk) o conference review (cr) scritti in inglese o italiano.

La stringa di ricerca usata è:

TITLE-ABS-KEY (("agile") AND ("team*") AND ("productivity" OR "performance") AND ("factor*" OR "aspect"))

È stato individuato un corpus iniziale di 395 articoli, selezionati in base alle parole chiave di ricerca. Attraverso l'analisi bibliometrica è stato possibile analizzare le pubblicazioni relative a questo argomento dal 1996 a oggi. Dall'analisi di tendenza delle pubblicazioni nel corso del tempo (*Figura 3*) è possibile notare che l'interesse per il tema dei fattori di produttività nei progetti Agile è emerso nel 2004, quando sono state pubblicate le prime pubblicazioni a riguardo. Tuttavia, il trend mostra un'intensificazione a partire dal 2011, quando il numero di pubblicazioni sull'argomento è aumentato notevolmente. Una delle principali ragioni che potrebbero spiegare tale aumento è la crescente popolarità della metodologia Agile e l'aumento della consapevolezza dell'importanza della produttività. Due importanti catalizzatori di questo cambiamento sono stati il rilascio della prima versione della Scrum Guide nel 2010, a opera di Schwaber e Sutherland e la presentazione della

certificazione PMI Agile nel 2011. Parallelamente, la diffusione degli strumenti per la misurazione della produttività del progetto Agile ha reso più facile raccogliere dati e aumentare la comprensione dei fattori che ne influenzano l'efficacia.

Si può osservare come il numero di pubblicazioni su questo tema sia ancora relativamente basso rispetto ad altri ambiti di ricerca legati alla gestione dei progetti, tuttavia è indice dell'ampio spazio disponibile per ulteriori studi e approfondimenti.

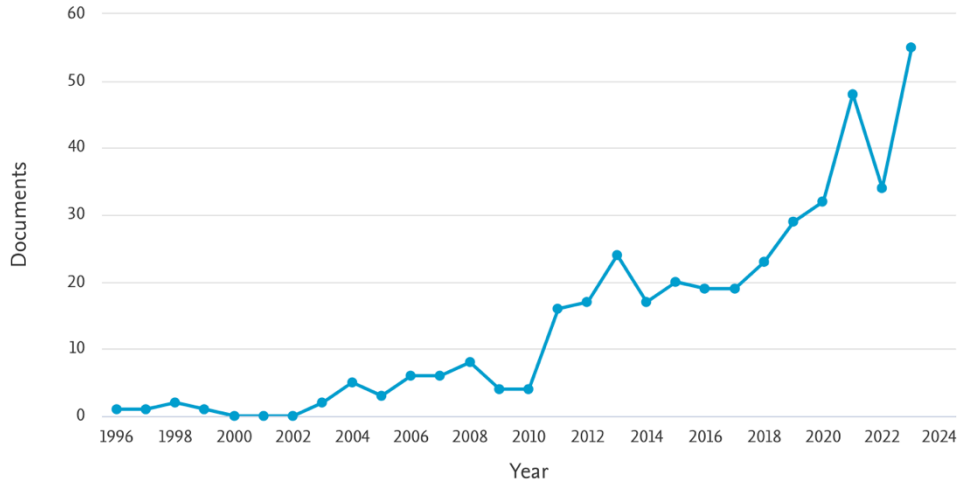

Figura 3: Documenti per anno in SCOPUS

Nella *Figura 4* è possibile vedere, in ordine decrescente, la lista dei primi dieci autori, per numero di pubblicazioni, riguardo l'argomento oggetto dello studio. Gli autori citati sono tutti rinomati ricercatori accademici il cui contributo alle metodologie Agile è ampiamente riconosciuto nel settore. In particolare, analizzando le biografie di questi autori, si evidenzia che Fabian Kortum e Kurt Schneider lavorano entrambi presso la Leibniz Universität Hannover in Germania e hanno pubblicato congiuntamente diversi articoli in cui esaminano come il feedback rapido e le dinamiche di squadra influenzino i risultati dei progetti Agile. Una situazione analoga si riscontra con Daniel C. Amaral e Edivandro C. Conforto, entrambi ricercatori presso l'Universidade de São Paulo, impegnati in ricerche sui fattori critici che influenzano la produttività dei team Agile.

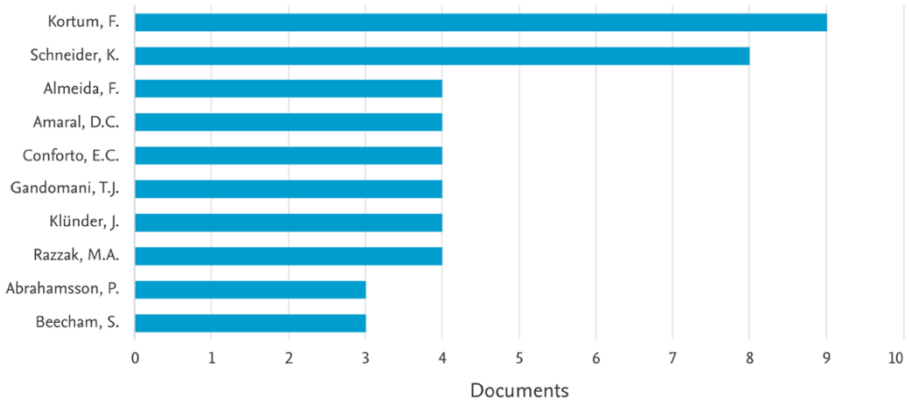

Figura 4: Documenti per autore in SCOPUS

Come si evince dal grafico riportato in *Figura 5*, gli Stati Uniti occupano la posizione di vertice per il numero di pubblicazioni nell'ambito in questione. Questa predominanza si deve al fatto che molte delle metodologie di gestione dei progetti più riconosciute a livello globale sono nate e si sono sviluppate negli Stati Uniti. Bisogna considerare, inoltre, che la metodologia Agile è stata adottata da molte aziende che sono state tra le prime a sperimentare questi approcci per affrontare le sfide complesse dei progetti e che, negli Stati Uniti, è presente una vasta rete di istituti accademici e centri di ricerca che hanno contribuito attivamente alla promozione della ricerca e alla pubblicazione di articoli in questo campo.

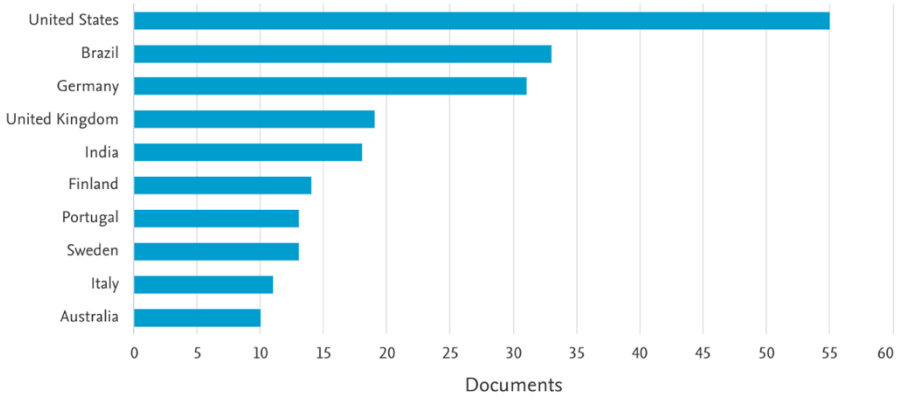

Figura 5: Documenti per Paese in SCOPUS

Il corpus analizzato è principalmente composto da articoli e conference paper, che insieme rappresentano circa l'80% dei documenti esaminati (*Figura 6*). Seguono le conference

review, che costituiscono una percentuale inferiore, pari al 13,1%. Anche se i libri e i capitoli di libri formano solo una piccola parte del totale, è fondamentale notare che possono conferire maggiore credibilità accademica, soprattutto per quanto riguarda concetti e metodologie consolidate e ampiamente accettate dalla comunità scientifica. Ad esempio, i libri pubblicati dal Project Management Institute hanno avuto un ruolo importante nella definizione di standard e pratiche riconosciute a livello globale in questo ambito di studio.

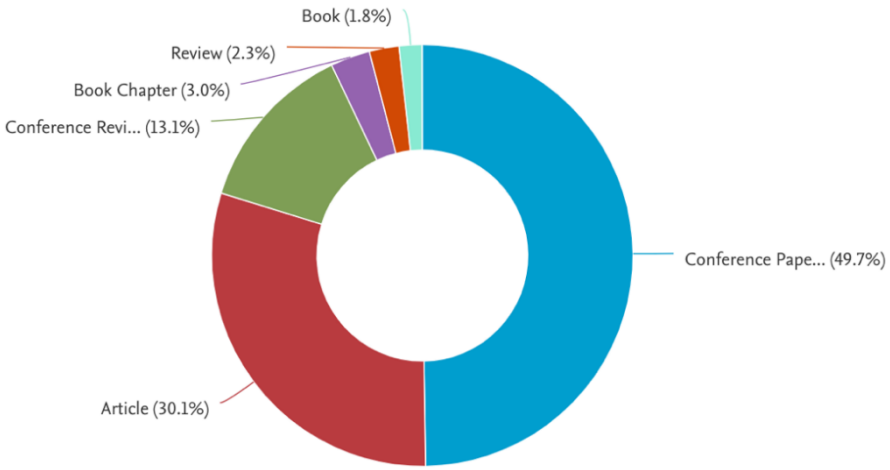

Figura 6: Documenti per tipo in SCOPUS

La distribuzione degli studi relativi ai fattori che influenzano la produttività dei team Agile mostra notevoli disparità tra i diversi settori industriali. Come si può evincere dal grafico a torta presente in *Figura 7*, il settore con il maggior impatto è quello informatico, con il 36,8% di pubblicazioni. Come ben noto, la metodologia Agile affonda le sue radici nei progetti di sviluppo software, dove nasce per rispondere alle sfide specifiche di questo settore come il bisogno di adattarsi rapidamente ai cambiamenti, la collaborazione tra team multidisciplinari e la consegna veloce di prodotti di qualità e per migliorare l'efficienza e l'efficacia dei team. Al secondo posto, con una quota del 13,6%, c'è il settore dell'ingegneria, il cui stretto legame con l'informatica si traduce in un aumento dei progetti di ingegneria che incorporano componenti tecnologici e si basano su metodi Agile per gestire complessità e rischi. Il campo

matematico, pur contribuendo in misura minore, è tra i più importanti perché fornisce le basi teoriche e concettuali per risolvere problemi complessi, ottimizzare algoritmi, modellare fenomeni reali e di progettare e implementare soluzioni robuste ed efficienti. Il settore economia, gestione e contabilità è, invece, strettamente legato al tema della produttività nei progetti Agile poiché la gestione efficiente delle risorse umane, insieme a una pianificazione accurata e una valutazione continua dei progressi, sono fondamentali per massimizzare i risultati e massimizzare il valore per l'azienda.

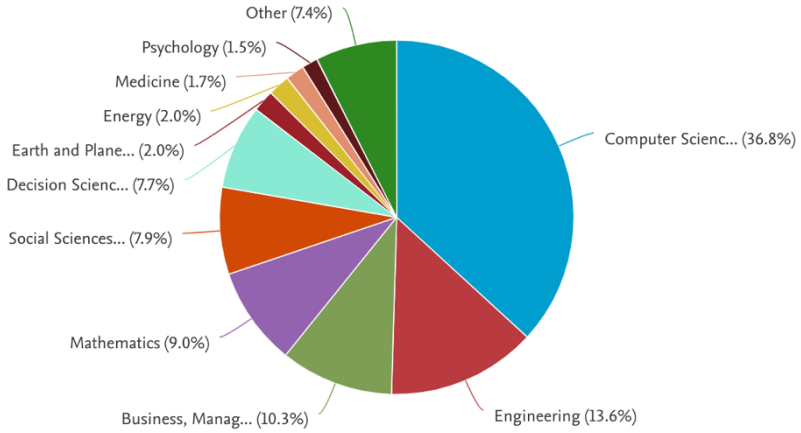

Figura 7: Documenti per area tematica in SCOPUS

4. Metodologia

In primo luogo, sono state formulate le domande di ricerca (RQ). Successivamente, per restringere il campo di ricerca e individuare gli studi rilevanti da includere, è stata eseguita una selezione basata su criteri di inclusione ed esclusione. Nella terza fase è stata effettuata un'accurata valutazione della qualità degli studi per identificare quelli più pertinenti. Nel quarto passaggio sono state estratte le informazioni utili a rispondere alle domande di ricerca. Infine, le informazioni sono state sintetizzate per presentare i risultati ottenuti.

4.1. Domande di ricerca

Nella prima fase dello studio sono state definite le domande di ricerca (RQ) seguendo il protocollo PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses).

RQ1: "*Quali fattori influiscono sulla produttività del team nello sviluppo software Agile?*"

RQ2: "*In che modo i fattori identificati influenzano la produttività del team nelle metodologie Agile?*"

Queste domande di ricerca hanno permesso di impostare un percorso chiaro e strutturato per l'intera revisione, guidando attentamente tutte le fasi di analisi e raccolta dei dati.

4.2. Identificazione degli studi

Sulla base di specifici criteri di inclusione ed esclusione, in questa fase sono stati scelti tutti gli articoli potenzialmente rilevanti per la revisione. La ricerca è stata sviluppata scaricando gli articoli da piattaforme come ResearchGate e ScienceDirect, i quali sono stati successivamente raccolti su Mendeley. A seguito dell'acquisizione degli articoli, sono stati rimossi i duplicati e gli articoli redatti in lingue diverse da inglese e italiano. È stato inoltre deciso di applicare un filtro per area tematica, includendo esclusivamente le seguenti discipline: informatica, ingegneria, matematica, economia, gestione e contabilità, scienze

decisionali, scienze sociali e psicologia. Completata questa selezione preliminare, si è passati alla fase di screening.

4.3. Selezione degli studi

La fase di screening è stata svolta in tre passaggi. È stata condotta una prima scrematura degli articoli considerando soltanto il titolo e l'abstract, per determinare se un articolo fosse rilevante per l'argomento di studio. Successivamente, è stato scaricato il testo completo degli articoli rimanenti e il numero di quelli non reperibili è stato annotato sotto "report non disponibili". Nell'ultima fase dello screening, sono stati esaminati integralmente gli articoli che avevano superato i precedenti passaggi, per valutarne l'idoneità all'inclusione nella revisione. Gli articoli inclusi nella revisione presentano le seguenti caratteristiche:

1. forniscono analisi originali o approfondimenti specifici, andando oltre la mera classificazione o la riproduzione di revisioni sistematiche;
2. affrontano in modo dettagliato il tema della produttività, identificando chiaramente i fattori che la influenzano;
3. analizzano fattori pertinenti alla produttività dei team Agile.

Questa scrematura ha consentito di determinare i 33 articoli da includere nello studio.

Il diagramma che illustra la strategia di selezione seguendo le linee guida PRISMA è riportato nella *Figura 8*.

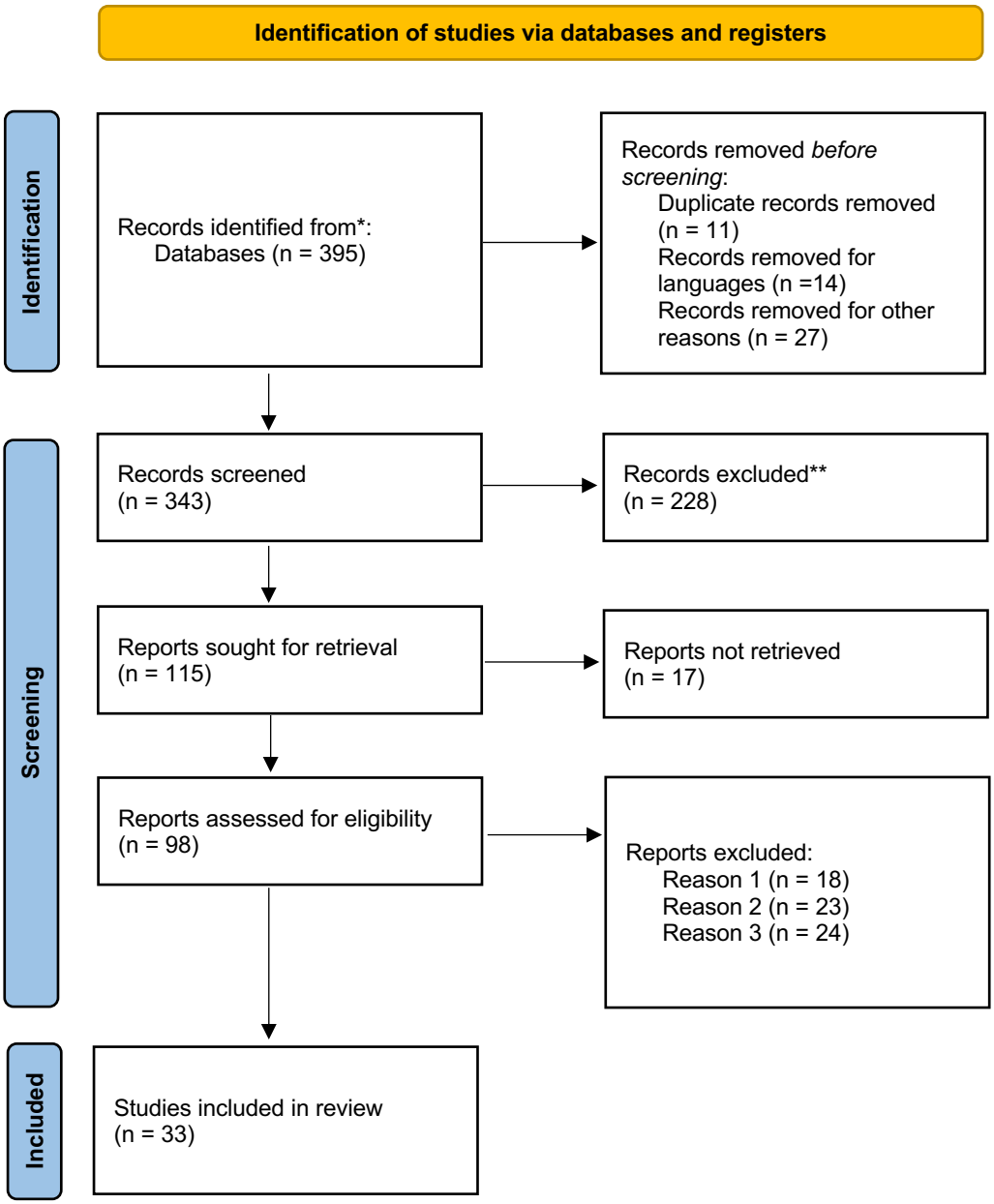

Figura 8: Identificazione degli articoli secondo le linee guida PRISMA

5. Interpretazione dei risultati

Questa sezione risponde alle domande di ricerca delineate precedentemente.

RQ1. Quali fattori influiscono sulla produttività del team nello sviluppo software Agile?

Per identificare i fattori che influiscono sulla produttività del team, abbiamo esaminato ogni articolo incluso nello studio ed elencato nella *Tabella 1*.

Autore	**Titolo**	**Anno**
Chen C.-Y.; Lee J.-C.	Exploring Teams' Temporal Factors for Determining Process Tailoring that Promotes the Evolution of Agilebased Software Development	2023
De Melo C.O.; S. Cruzes D.; Kon F.; Conradi R.	Interpretative case studies on agile team productivity and management	2013
Dingsøyr T.; Lindsjørn Y.	Team performance in agile development teams: Findings from 18 focus groups	2013
Drury-Grogan M.L.	Performance on agile teams: Relating iteration objectives and critical decisions to project management success factors	2014
Dutra E.; Santos G.	Organisational climate assessments of agile teams – a qualitative multiple case study	2021
Fagerholm F.; Ikonen M.; Kettunen P.; Münch J.; Roto V.; Abrahamsson P.	Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments	2016
Inayat I.; Salim S.S.	A framework to study requirements-driven collaboration among agile teams: Findings from two case studies	2015
Jiang J.; Yang X.; Zhou C.	The impact of social media usage on expertise coordination and team creative performance in distributed agile software development	2023
Kakar A.K.	Do Reflexive Software Development Teams Perform Better?	2017
Kaleemunnisa; Scharff C.; Bathula K.M.; Chen K.	Analyzing Scrum Team Impediments Using NLP	2023
Kortum F.; Karras O.; Klünder J.; Schneider K.	Towards a Better Understanding of Team-Driven Dynamics in Agile Software Projects: A Characterization and Visualization Support in JIRA	2019
Larusdottir M.K.; Kyas M.	Assessing the Performance of Agile Teams	2020
Meier A.; Kock A.	The human factor in agility: Exploring employee dedication in agile project organizations	2023
Parrish A.; Smith R.; Hale D.; Hale J.	A field study of developer pairs: Productivity impacts and implications	2004
Pirola-Merlo A.	Agile innovation: The role of team climate in rapid research and development	2011
Rahy S.; Bass J.M.	Implementation of Agile Methodology in Developing Countries: Case Study in Lebanon	2020
Ramírez-Mora S.L.; Oktaba H.; Patlán Pérez J.	Group maturity, team efficiency, and team effectiveness in software development: A case study in a CMMI-DEV Level 5 organization	2019
Ståhl D.	The dynamic versus the stable team: The unspoken question in large-scale agile development	2023
Tavakoli F.; Gandomani T.J.	A novel team productivity model for XP teams	2018
Thanthony S.; Marnewick A.; Marnewick C.	Communication patterns and team performance within agile software development project	2022
Vishnubhotla S.D.; Mendes E.; Lundberg L.	Understanding the perceived relevance of capability measures: A survey of Agile Software Development practitioners	2021
Vishnubhotla S.D.; Mendes E.; Lundberg L.	Investigating the relationship between personalities and agile team climate of software professionals in a telecom company	2020

Wood S.; Michaelides G.; Thomson C.	Successful extreme programming: Fidelity to the methodology or good teamworking?	2013
Topp J.; Hille J.H.; Neumann M.; Mötefindt D.	How a 4-Day Work Week and Remote Work Affect Agile Software Development Teams	2022
Iqbal J.; Omar M.; Yasin A.	An empirical analysis of the effect of agile teams on software productivity	2019
Fatema I.; Sakib K.	Factors Influencing Productivity of Agile Software Development Teamwork	2018
Kortum F.; Klunder J.; Schneider K.	Behavior-Driven Dynamics in Agile Development	2019
Alahyari H.; Horkoff J.; Matsson O.; Egenvall K.	What Do Agile Teams Find Important for their Success?	2018
Melo C.; Cruzes D.S.; Kon F.; Conradi R.	Agile Team Perceptions of Productivity Factors	2011
Scott E.; Charkie K.N.; Pfahl D.	Productivity, Turnover, and Team Stability of Agile Teams in Open-Source Software Projects	2020
Mashmool A.; Khosravi S.; Joloudari J.H.; Inayat I.; Gandomani T.J.; Mosavi A.	A Statistical Model to Assess the Team's Productivity in Agile Software Teams	2021
Barbosa I.F.; Oliveira M.P.; Reis P.B.S.; Gomes T.C.S.; Da Silva F.Q.B.	Towards understanding the relationships between interdependence and trust in software development: A qualitative research	2017
Moloto M.; Harmse A.; Zuva T.	Agile Methodology use factors that influence project performance in South African Banking sector-A case study	2021

Tabella 1: Studi selezionati

I fattori individuati sono stati suddivisi in quattro categorie per agevolarne la descrizione e la comprensione all'interno della *Tabella 2*:

- *team*: riguarda i fattori legati all'ambiente di lavoro, alle dinamiche interne al gruppo e alla distanza tra i membri;
- *organizzazione*: riguarda le caratteristiche della leadership, i tassi di turnover, la gestione del team e della formazione individuale;
- *processo*: coinvolge i fattori legati alle pratiche adottate per lo sviluppo del software, alle tecniche di gestione e agli strumenti volti a garantire la qualità del progetto;
- *individuo*: comprende i fattori personali come competenze tecniche, soft skills e motivazione.

Dimensione	Fattore
Team	Comunicazione Interazione tra i membri del team Progettazione del team Distanza tra i membri del team Ambiente di lavoro Sicurezza psicologica
Organizzazione	Gestione del team Leadership Turnover rate Gestione della formazione individuale
Processo	Pratiche per lo sviluppo del software Tecniche per la gestione dei progetti software Qualità
Individuo	Motivazione Capacità tecniche Soft skills

Tabella 2: Fattori suddivisi per categoria

RQ2: "In che modo i fattori identificati influenzano la produttività del team quando si utilizzano metodologie Agile?"

Per rispondere alla seconda domanda di ricerca, è stata condotta un'analisi per comprendere come i fattori individuati nella fase precedente influenzino le prestazioni dei team, utilizzando le informazioni estratte dagli studi selezionati.

5.1. Team

Come si evince dalla Tabella 3, il fattore più frequente per la dimensione del team è la comunicazione, considerata uno dei principi fondamentale della metodologia Agile (Larusdottir and Kyas, 2020). I problemi di comunicazione, come la mancanza di chiarezza e le incomprensioni, hanno un impatto negativo sullo sviluppo software perché ostacolano il progresso (Ramírez-Mora, Oktaba and Patlán Pérez, 2020).

	Comunicazione	Interazione tra i membri del team	Progettazione del team	Distanza tra i membri del team	Ambiente di lavoro	Sicurezza psicologica
Exploring Teams' Temporal Factors for Determining Process Tailoring that Promotes the Evolution of Agilebased Software Development		x		x		
Interpretative case studies on agile team productivity and management			x			
Team performance in agile development teams: Findings from 18 focus groups	x	x				
Performance on agile teams: Relating iteration objectives and critical decisions to project management success factors			x		x	
Organisational climate assessments of agile teams – a qualitative multiple case study		x				
Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments		x				
A framework to study requirements-driven collaboration among agile teams: Findings from two case studies	x	x		x		
The impact of social media usage on expertise coordination and team creative performance in distributed agile software development	x	x				
Analyzing Scrum Team Impediments Using NLP	x		x			
Towards a Better Understanding of Team-Driven Dynamics in Agile Software Projects: A Characterization and Visualization Support in JIRA	x	x	x		x	x
Assessing the Performance of Agile Teams	x					
The human factor in agility: Exploring employee dedication in agile project organizations		x				
A field study of developer pairs: Productivity impacts and implications		x				
Agile innovation: The role of team climate in rapid research and development					x	x
Implementation of Agile Methodology in Developing Countries: Case Study in Lebanon	x					
Group maturity, team efficiency, and team effectiveness in software development: A case study in a CMMI-DEV Level 5 organization	x		x			
The dynamic versus the stable team: The unspoken question in large-scale agile development						
A novel team productivity model for XP teams	x			x		
Communication patterns and team performance within agile software development project	x	x		x		
Understanding the perceived relevance of capability measures: A survey of Agile Software Development practitioners	x					
Investigating the relationship between personalities and agile team climate of software professionals in a telecom company						x
Successful extreme programming: Fidelity to the methodology or good teamworking?	x					
An empirical analysis of the effect of agile teams on software productivity		x				
Factors Influencing Productivity of Agile Software Development Teamwork	x					
Behavior-Driven Dynamics in Agile Development	x					
What Do Agile Teams Find Important for their Success?					x	
Agile Team Perceptions of Productivity Factors			x			
Productivity, Turnover, and Team Stability of Agile Teams in Open-Source Software Projects						
Towards understanding the relationships between interdependence and trust in software development: A qualitative research		x				
TOTALE	**14**	**12**	**6**	**4**	**4**	**3**

Tabella 3: Frequenza dei fattori per la dimensione del team

Una comunicazione efficace, invece, facilita la gestione dei conflitti e l'inserimento di nuovi membri nel team, permettendo loro di diventare produttivi più rapidamente e di abbassare la curva di apprendimento (Fatema and Sakib, 2017). Durante la pianificazione, la comunicazione tra i membri della squadra influisce direttamente sulla produttività del gruppo; nella fase di produzione, consente una migliore coordinazione e garantisce che le attività siano allineate con gli obiettivi di business; nella fase di manutenzione, la comunicazione continua con i clienti è necessaria per affrontare adeguatamente le richieste di modifica e i problemi tecnici (Tavakoli and Gandomani, 2018). La comunicazione tra i membri del team può essere formale, ad esempio tramite canali ufficiali, o informale, tramite interazioni spontanee ed entrambe le forme di comunicazione sono ugualmente importanti per il successo del team. In particolare, la comunicazione informale aiuta a sviluppare il rapporto di fiducia e la risoluzione dei problemi tra i membri del team (Inayat and Salim, 2015). Tuttavia, le differenze culturali e linguistiche possono rappresentare una barriera alla comunicazione, influenzando le dinamiche del team (Kaleemunnisa *et al.*, 2023). L'uso di lingue diverse nelle interazioni può creare difficoltà nella comprensione delle informazioni, portando a una riduzione della frequenza delle conversazioni.

La scelta del mezzo di comunicazione può variare in base alle esigenze informative, al contesto e alla fase del progetto ed ha un ruolo chiave all'interno delle dinamiche dei team. Analizzando le diverse fasi di un progetto, è emerso che la comunicazione faccia a faccia è generalmente preferita nelle fasi iniziali; al contrario, durante le fasi di produzione e chiusura del progetto, si tende a privilegiare la comunicazione asincrona, che consente una gestione più autonoma delle attività e riduce le interruzioni operative (Vishnubhotla, Mendes and Lundberg, 2021). La pandemia di COVID-19 ha messo in luce l'importanza di adottare un approccio comunicativo più flessibile e adattabile, favorendo una maggiore diffusione della

comunicazione asincrona rispetto ad altre modalità (Thanthony, Marnewick and Marnewick, 2022). Questo cambiamento è stato reso possibile grazie all'impiego di strumenti digitali come Slack e le piattaforme per videoconferenze (Rahy and Bass, 2020), che permettono ai team di mantenere la continuità operativa anche a distanza.

Tuttavia, i team percepiscono che un'eccessiva decentralizzazione della comunicazione possa ostacolare il trasferimento efficace delle informazioni, con il rischio di compromettere la collaborazione e la coesione del gruppo (Kortum *et al.*, 2019). Un elemento critico nella comunicazione asincrona è la tempestività, poiché i ritardi nelle risposte possono rallentare il flusso informativo e diminuire l'efficacia dello scambio. In tal senso, l'utilizzo dei social media si è rivelato utile per aumentare la rapidità e la fluidità nella condivisione di informazioni e conoscenze (Jiang, Yang and Zhou, 2024). Nonostante questi benefici, la comunicazione virtuale presenta delle limitazioni significative, tra cui la perdita dei segnali non verbali come il linguaggio del corpo e le espressioni facciali, che sono elementi essenziali per una comunicazione empatica ed efficace nelle interazioni faccia a faccia (Kaleemunnisa *et al.*, 2023).

Un fattore strettamente collegato alla comunicazione è la distanza geografica tra i membri del team. Nei team distribuiti, l'assenza di contatto diretto può ostacolare la costruzione della fiducia e ridurre il coinvolgimento dei membri (Tavakoli and Gandomani, 2018). Inoltre, la distanza può generare una progressione instabile del progetto, con ritmi discontinui e imprevedibili (Chen and Lee, 2023) e decisioni frammentate che compromettono il coordinamento e l'efficacia del lavoro di squadra (Inayat and Salim, 2015).

Il secondo fattore più frequente in *Tabella 3* è l'interazione tra i membri del team. La collaborazione efficace tra i membri del team non è semplicemente auspicabile, ma rappresenta una vera e propria necessità per il raggiungimento degli obiettivi del progetto (Kaleemunnisa *et al.*, 2023). Una buona interazione permette di sviluppare modelli mentali

condivisi, grazie ai quali i membri hanno una comprensione comune delle dinamiche del lavoro, degli obiettivi e delle strategie (Dingsøyr and Lindsjørn, 2013). Alla base delle interazioni positive vi sono la fiducia e la coesione. La prima influisce sul modo in cui i membri gestiscono i compiti (Barbosa *et al.*, 2017) poiché un alto livello di fiducia facilita la delega dei compiti e riduce la necessità di controllo e supervisione, così che i membri del team possano lavorare in maniera più efficiente (Dingsøyr and Lindsjørn, 2013). La coesione, a sua volta, si riferisce al legame che si sviluppa tra i membri del team, rafforzato dalle esperienze condivise. Un team coeso sviluppa un forte senso di appartenenza e collaborazione (Dutra and Santos, 2020), fattori che incrementano la motivazione individuale e collettiva, contribuendo a una maggiore stabilità del gruppo. L'interazione tra i membri del team è essenziale anche per il monitoraggio reciproco delle prestazioni che permette ai membri del team di osservare e valutare le prestazioni degli altri. Questo processo permette di identificare rapidamente le aree di miglioramento e di apportare le modifiche necessarie per ottimizzare il lavoro, oltre ad aumentare sia la responsabilità individuale che quella collettiva (Dingsøyr and Lindsjørn, 2013). L'interazione continua tipica delle pratiche agili, nella programmazione in coppia è fondamentale. Tuttavia, i risultati di uno studio (Parrish *et al.*, 2004) hanno dimostrato che le coppie di programmatori che lavorano sullo stesso compito tendono a produrre codice di qualità superiore rispetto a quello sviluppato da programmatori che lavorano individualmente, ma sperimentano anche una perdita di produttività a causa di lavoro duplicato o di modifiche contrastanti che necessitano di tempo aggiuntivo per essere riconciliate. L'adozione di un protocollo per la collaborazione basato sui ruoli, in cui un programmatore assume il ruolo di "driver" (responsabile della scrittura del codice) e l'altro di "navigator" (che supervisiona e fornisce feedback) permette di mitigare il calo di produttività e ridurre le inefficienze (Parrish *et al.*, 2004). Quando l'interazione è positiva, anche i conflitti possono essere affrontati in modo

costruttivo perchè i membri del team sono più inclini a discutere delle loro differenze e cercare insieme delle soluzioni, piuttosto che lasciare che i conflitti si intensifichino (Kortum *et al.*, 2019). Inoltre, la costruzione di relazioni positive contribuisce ad aumentare il morale (Iqbal, Omar and Yasin, 2019). Un altro aspetto relativo alle interazioni tra i membri del team è la condivisione regolare di informazioni e al trasferimento delle conoscenze individuali a livello collettivo (Meier and Kock, 2023). Questo processo consente ai membri del team di apprendere gli uni dagli altri, favorendo la crescita della maturità cognitiva del gruppo (Thanthony, Marnewick and Marnewick, 2022). La collaborazione all'interno del team promuove il coordinamento delle competenze: quando ciascun individuo conosce a fondo le abilità degli altri, è possibile distribuire i compiti in modo più efficiente, sfruttando le competenze specifiche di ognuno e ottimizzando il lavoro di squadra (Jiang, Yang and Zhou, 2024).

Le scelte di progettazione del team sono un fattore importante per la produttività dei team Agile che si basano sul lavoro di squadra. Le dimensioni e l'identità del team, così come la diversità e le competenze dei membri, sono tutti elementi da tenere in considerazione quando si progettano i team Agile. Un team che include sia membri esperti che meno esperti consente di combinare conoscenze consolidate e flessibilità, sfruttando al meglio le capacità di ciascun individuo per rispondere efficacemente alle esigenze di progetto (De O. Melo *et al.*, 2013). In particolare, i membri esperti apportano un valore aggiunto in termini di conoscenza, mentre quelli più giovani contribuiscono con la loro flessibilità (Melo *et al.*, 2011). Inoltre, è fondamentale che i membri del team possiedano competenze complementari per favorire un'efficace collaborazione (Drury-Grogan, 2014). Nei team interfunzionali, i membri acquisiscono familiarità con aree che esulano dalle loro competenze specifiche e sono in grado di gestire compiti interdisciplinari. Questa versatilità consente di colmare eventuali lacune e di favorire il supporto reciproco tra i membri del

gruppo (Meier and Kock, 2023). Un altro aspetto rilevante è la maturità del team, poiché esiste una correlazione positiva tra la crescita del gruppo e la sua produttività. I team più maturi dimostrano una maggiore capacità di ottimizzare le risorse e di raggiungere gli obiettivi con minor sforzo (Ramírez-Mora, Oktaba and Patlán Pérez, 2020). Anche la dimensione del team è un fattore da tenere in conto quando si valuta la composizione del team. Gruppi più piccoli favoriscono un maggiore senso di responsabilità e una migliore visione d'insieme del prodotto; viceversa, all'aumentare del numero di membri, cresce anche la complessità della comunicazione e il rischio di conflitti interni, fattori che possono rallentare il progresso del progetto (De O. Melo *et al.*, 2013).

Un altro aspetto da tenere in considerazione quando si analizzano i fattori relativi alla produttività dei team Agile è l'ambiente di lavoro che influisce sulla produttività e si riferisce all'insieme di norme, atteggiamenti e aspettative che gli individui percepiscono in un contesto sociale specifico (Pirola-Merlo, 2010). È stato osservato che un ambiente di lavoro che promuove la fiducia e il supporto reciproco porta a un aumento della soddisfazione tra i membri del team (Drury-Grogan, 2014). Quando i dipendenti si sentono supportati e liberi di esprimere le proprie idee, sono più motivati a contribuire attivamente al progetto (Fagerholm *et al.*, 2015). Infatti, un clima positivo non solo migliora le prestazioni, ma accelera anche il completamento dei progetti (Pirola-Merlo, 2010). D'altra parte, un ambiente lavorativo negativo spesso porta ad un aumento dello stress che influisce sulla salute sia mentale che fisica di un individuo, facendo aumentare l'assenteismo e diminuire la produttività (Alahyari *et al.*, 2018).

La sicurezza psicologica è il fattore citato meno frequentemente negli studi selezionati, tuttavia è necessario per la creazione un ambiente lavorativo sano dove le persone possano sentirsi a loro agio ad esprimersi apertamente senza la paura di potenziali ripercussioni negative. Lavorare in un ambiente di questo tipo fa crescere il livello di produttività, in

quanto aumentano la creatività, l'impegno (Pirola-Merlo, 2010) e la comunicazione del team (Vishnubhotla, Mendes and Lundberg, 2020).

5.2. Organizzazione

	Gestione del team	Leadership	Turnover rate	Gestione della formazione individuale
Exploring Teams' Temporal Factors for Determining Process Tailoring that Promotes the Evolution of Agilebased Software Development		x		
Interpretative case studies on agile team productivity and management	x		x	
Team performance in agile development teams: Findings from 18 focus groups		x		
Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments	x			x
Analyzing Scrum Team Impediments Using NLP	x			
The human factor in agility: Exploring employee dedication in agile project organizations	x			
A field study of developer pairs: Productivity impacts and implications				x
Group maturity, team efficiency, and team effectiveness in software development: A case study in a CMMI-DEV Level 5 organization	x			
The dynamic versus the stable team: The unspoken question in large-scale agile development			x	
A novel team productivity model for XP teams	x		x	
Communication patterns and team performance within agile software development project		x		
How a 4-Day Work Week and Remote Work Affect Agile Software Development Teams	x			
An empirical analysis of the effect of agile teams on software productivity	x	x		
Factors Influencing Productivity of Agile Software Development Teamwork	x	x		x
Behavior-Driven Dynamics in Agile Development				
What Do Agile Teams Find Important for their Success?		x		x
Agile Team Perceptions of Productivity Factors			x	
Productivity, Turnover, and Team Stability of Agile Teams in Open-Source Software Projects			x	
A Statistical Model to Assess the Team's Productivity in Agile Software Teams		x		
TOTALE	**9**	**7**	**5**	**4**

Tabella 4: Frequenza dei fattori per la dimensione organizzativa

Come si evince dalla *Tabella 4*, la gestione del team a livello organizzativo emerge come fattore determinante per il successo della metodologia Agile, in cui il team ricopre un ruolo chiave. L'implementazione di politiche di gestione ottimali dei team porta ad un aumento della motivazione tra i membri che favorisce una migliore interazione (Tavakoli and Gandomani, 2018). Grazie a questa sinergia viene promossa la condivisione delle

conoscenze, elemento chiave per il raggiungimento di una maggiore produttività complessiva del team. Inoltre, assegnare i membri alle attività a tempo pieno garantisce maggiore concentrazione e coinvolgimento (De O. Melo et al., 2013). La distribuzione equa del carico di lavoro e la riduzione delle attività non autorizzate (Kortum et al., 2019) così come il monitoraggio reciproco delle prestazioni (Fatema and Sakib, 2017), migliorano la produttività. Un team ben organizzato ed efficace si basa sulla chiarezza di ruoli e obiettivi, poiché permette ai membri di comprendere le proprie responsabilità e ciò che gli altri si aspettano da loro (Ramírez-Mora, Oktaba and Patlán Pérez, 2020). Quando i membri del team hanno la libertà di scegliere il metodo più adatto per eseguire un compito e definire come lavorare insieme, si sviluppa maggiore autonomia (Meier and Kock, 2023). Inoltre, i team che stabiliscono autonomamente i propri obiettivi tendono a essere più produttivi rispetto a quelli in cui le direttive provengono dal leader grazie al maggiore coinvolgimento nei processi aziendali (Iqbal, Omar and Yasin, 2019). La partecipazione del team alla selezione di nuovi membri contribuisce a mantenere l'identità del gruppo (Fagerholm et al., 2015). L'introduzione di strategie innovative come la settimana lavorativa corta non solo favoriscono un migliore equilibrio tra lavoro e vita personale, ma migliorano anche l'efficienza e la concentrazione durante il lavoro (Topp et al., 2022).

Il secondo fattore più discusso a livello organizzativo è la leadership. Quest'ultima consiste nell'abilità di guidare e coordinare efficacemente le attività dei membri del team, monitorare le prestazioni, assegnare compiti e favorire lo sviluppo delle competenze e abilità all'interno del gruppo (Mashmool et al., 2021). Un buon leader motiva i membri, pianifica e organizza il lavoro e crea un ambiente positivo che stimola la collaborazione e l'impegno (Dingsøyr and Lindsjørn, 2013). Lo stile di leadership più apprezzato nei team Agile è la "servant leadership", ovvero un approccio focalizzato sul team piuttosto che sull'individuo (Thanthony, Marnewick and Marnewick, 2022). Questo stile supporta la condivisione dei

compiti della leadership tra tutti i membri del team, promuovendo una maggiore autonomia e condivisione delle informazioni (Iqbal, Omar and Yasin, 2019). Tuttavia, uno studio (Chen and Lee, 2023) afferma l'importanza della leadership temporale per gestire le attività legate al tempo, come la pianificazione, e per mantenere il team allineato sui obiettivi e scadenze. Ciò è particolarmente importante nei progetti Agile, dove le scadenze sono serrate e le modifiche ai piani possono avvenire rapidamente.

Dalle ricerche selezionate emerge che il turnover influisce in maniera ambivalente sulla produttività dei team di sviluppo software Agile. Un basso livello di turnover consente di avere team stabili (Ståhl, 2023), i cui membri conoscono bene le competenze e lo stile di comunicazione degli altri, facilitando l'interazione e lo sviluppo di pratiche e processi consolidati nel tempo. Quando i membri del team cambiano frequentemente, invece, risulta più difficile preservare la coesione del gruppo e si può verificare un aumento dello stress e del carico di lavoro dovuto al processo di integrazione di nuovi membri all'interno del team, il quale prevede sessioni di formazione e la riorganizzazione della routine (Scott, Charkie and Pfahl, 2020). La fase di sviluppo del team e i processi di gestione dei conflitti congiuntamente alla personalità dei membri del team sono le cause individuate in relazione ad un maggiore turnover (De O. Melo *et al.*, 2013). Nonostante l'adozione di pratiche come la programmazione in coppia e le riunioni quotidiane, le interviste e le retrospettive hanno rivelato che i team hanno riscontrato notevoli difficoltà a consegnare le storie nelle iterazioni successive al cambio di personale (De O. Melo *et al.*, 2013). Tuttavia, la stabilità del team può portare a stagnazione (Melo *et al.*, 2011), la quale limita l'introduzione di nuove idee e soluzioni che inducono un miglioramento nei processi di coordinamento del team (De O. Melo *et al.*, 2013). I team dinamici favoriscono l'upskilling e la creazione di un "pool di competenze" grazie alla diverse esperienze e competenze di membri diversi del team; inoltre tendono ad essere più flessibili e maggiormente pronti di fronte a nuove richieste di mercato

o del cliente (Ståhl, 2023). Quanto detto serve a dimostrare che anche un fattore di disturbo come il turnover può in realtà portare ad un aumento della produttività grazie all'integrazione di nuovi membri, i quali condivideranno le competenze acquisite nel tempo con i membri esistenti.

Nell'ambito della dimensione organizzativa è emerso un ulteriore fattore, ovvero la gestione della formazione individuale. Le opportunità di apprendimento e sviluppo personale sono fondamentali per attrarre e mantenere membri di alto livello nei team (Fagerholm et al., 2015). Infatti, le aziende che offrono possibilità di crescita professionale tendono a essere più attraenti per i talenti e motivano i membri del team a partecipare a progetti che consentono loro di risolvere problemi apprendendo nuove competenze. Inoltre, investire nella formazione tecnica e nello sviluppo delle competenze consente ai membri di essere preparati di fronte alle sfide dei progetti (Alahyari et al., 2018) e migliorare l'efficacia del team (Fatema and Sakib, 2017). Anche per i membri con più anni di esperienza, il supporto formativo è essenziale per adattarsi all'evoluzione delle pratiche di lavoro nel tempo (Parrish et al., 2004). Un ambiente che incoraggia l'apprendimento continuo e il miglioramento personale contribuisce a creare una cultura di innovazione e collaborazione all'interno del team (Alahyari et al., 2018). Inoltre, l'inserimento di nuovi membri nel team può essere facilitato attraverso un buon processo di formazione, che riduce la curva di apprendimento e aumenta la produttività complessiva (Fatema and Sakib, 2017).

5.3. Processo

Come evidenziato nella *Tabella 5*, il fattore più ricorrente all'interno della dimensione del processo è rappresentato dalle pratiche per lo sviluppo del software. L'adozione di pratiche specifiche per la gestione del lavoro, come gli stand-up meetings, le pianificazioni iterative e le retrospettive, svolge un ruolo chiave nel mantenere il team focalizzato e motivato, facilitando la risoluzione tempestiva dei problemi e migliorando la produttività complessiva

(Vishnubhotla, Mendes and Lundberg, 2021). Una pianificazione iniziale più flessibile consente di dedicare maggiore attenzione allo sviluppo, facilitando l'adattamento ai cambiamenti e diminuendo il rischio di realizzare un prodotto non in linea con le necessità del mercato e degli utenti.

	Pratiche per lo sviluppo del software	Tecniche per la gestione dei progetti software	Qualità
Performance on agile teams: Relating iteration objectives and critical decisions to project management success factors	x		x
Do Reflexive Software Development Teams Perform Better?	x		
Assessing the Performance of Agile Teams			x
Implementation of Agile Methodology in Developing Countries: Case Study in Lebanon	x		
A novel team productivity model for XP teams		x	
Understanding the perceived relevance of capability measures: A survey of Agile Software Development practitioners	x		
Successful extreme programming: Fidelity to the methodology or good teamworking?	x	x	
An empirical analysis of the effect of agile teams on software productivity	x	x	x
Behavior-Driven Dynamics in Agile Development		x	
What Do Agile Teams Find Important for their Success?	x		
A Statistical Model to Assess the Team's Productivity in Agile Software Teams			x
Agile Methodology use factors that influence project performance in South African Banking sector-A case study	x		
TOTALE	**8**	**4**	**4**

Tabella 5: Frequenza dei fattori per la dimensione di processo

Questo metodo favorisce iterazioni più frequenti, promuovendo la collaborazione tra i membri del team e ottimizzando le performance complessive (Moloto, Harmse and Zuva, 2021). In merito ai fattori che influiscono sul successo di un progetto, uno studio ha scoperto che l'applicazione rigorosa delle pratiche stabilite migliora le prestazioni del team (Wood, 2013). Tuttavia, affinché queste pratiche siano efficaci, è fondamentale che le cerimonie Agile vengano comprese e applicate correttamente. Ad esempio, una gestione inefficace delle retrospettive può ostacolare il miglioramento continuo, mentre una pianificazione ben strutturata degli sprint consente di stabilire obiettivi chiari e di mantenere l'allineamento del team sulle attività da svolgere, rendendo il processo di sviluppo più fluido e la gestione del tempo più efficiente (Rahy and Bass, 2020). Inoltre, il rilascio prioritario delle funzionalità più rilevanti si conferma come un fattore determinante per il successo dei progetti Agile

(Alahyari *et al.*, 2018). Si evidenzia anche l'importanza del testing continuo, che inizia con il rilascio della prima iterazione e prosegue lungo l'intero ciclo di vita del progetto, al fine di identificare e risolvere tempestivamente eventuali problemi di qualità (Iqbal, Omar and Yasin, 2019). Il coinvolgimento attivo degli utenti finali nel processo di sviluppo è ritenuto fondamentale in quanto l'assenza di un'interazione costante con gli utenti può rallentare l'avanzamento del lavoro e compromettere l'efficacia delle soluzioni sviluppate (Rahy and Bass, 2020). Ad esempio, decisioni non coordinate con il cliente possono generare lavoro inutile e privo di valore, per cui è necessario un focus sulle sole attività che aggiungono effettivamente valore al prodotto finale (Drury-Grogan, 2014). Tra le pratiche emerse dagli studi analizzati figura anche la "reflexivity", un'attività che consiste nell'analisi continua delle dinamiche interne e delle decisioni prese, con l'obiettivo di introdurre miglioramenti nel processo di lavoro. L'efficacia di questa pratica risulta particolarmente elevata quando è accompagnata da un'elevata interdipendenza dei risultati tra i membri del team. In progetti caratterizzati da un alto grado di innovazione, la combinazione di reflexivity e interdipendenza contribuisce ad ottimizzare le prestazioni, mentre in contesti più standardizzati, un uso eccessivo della reflexivity può avere effetti contrari, introducendo complessità non necessarie e rallentando il progresso del lavoro (Kakar, 2017). Pratiche come la scrittura di codice facilmente testabile permettono di rilevare tempestivamente errori durante lo sviluppo, aumentando l'affidabilità del prodotto finale; in parallelo, l'adozione di un design semplice e lineare facilita la comprensione e la manutenzione del codice, riducendo la complessità delle soluzioni e rendendo più agevole l'implementazione di modifiche (Alahyari *et al.*, 2018).

Relativamente alle tecniche per la gestione dei progetti Agile, è stato osservato che l'adozione di standard di codifica collettivi, in cui tutti gli sviluppatori scrivono e mantengono il codice in un formato uniforme, ha un impatto diretto sulle prestazioni del

progetto (Wood, 2013). Questo approccio favorisce una collaborazione più efficace, consentendo a ciascun membro del team di intervenire su qualsiasi parte del codice in qualsiasi momento, con una conseguente riduzione significativa dei tempi di revisione e correzione (Tavakoli and Gandomani, 2018). Inoltre, la proprietà collettiva del codice facilita il trasferimento di conoscenze tra i membri del team, migliorando la coesione e la continuità del lavoro (Wood, 2013). L'impiego di test e build automatizzati permette di mantenere elevati standard qualitativi, evitando che le modifiche al codice introducano nuovi difetti. Tuttavia, è stata individuata una correlazione negativa tra l'uso dell'automazione nei test e la produttività dei team Agile poichè la loro implementazione e gestione possono risultare complesse e dispendiose in termini di tempo e risorse (Iqbal, Omar and Yasin, 2019). Scrivere codice facilmente testabile consente di individuare tempestivamente eventuali errori durante il ciclo di sviluppo, migliorando l'affidabilità del software; allo stesso tempo, adottare un design chiaro e lineare semplifica la comprensione e la manutenzione del codice, riducendo la complessità delle soluzioni e rendendo più agevole l'implementazione di modifiche (Alahyari *et al.*, 2018). Infine, la reversibilità delle modifiche al codice, che garantisce flessibilità, viene percepita dagli sviluppatori come vantaggiosa per la produttività (Iqbal, Omar and Yasin, 2019). A supporto del miglioramento continuo, l'integrazione di strumenti per il feedback proattivo, come il plugin ProDynamics in JIRA, si è rivelata efficace nell'incrementare la produttività (Kortum, Klunder and Schneider, 2019). I team che hanno avuto accesso a questo tipo di feedback hanno dimostrato una maggiore capacità di pianificare e gestire le attività, mantenendo prestazioni elevate, mentre i gruppi che ne erano privi hanno incontrato maggiori difficoltà nel raggiungere gli obiettivi prefissati.

Un ulteriore fattore che incide sulla produttività del team è la qualità. In particolare, la conformità agli standard qualitativi, misurata attraverso il numero di difetti riscontrati

durante le fasi di testing, rappresenta un elemento chiave per garantire che il software soddisfi le aspettative del cliente (Iqbal, Omar and Yasin, 2019). Utilizzando l'analisi di regressione, uno studio ha rilevato una correlazione positiva tra il rispetto di tali standard e la produttività del team: un aumento di un'unità nella conformità alla qualità comporta, infatti, un incremento medio di 0,21 unità nella produttività (Mashmool *et al.*, 2021). Inoltre, gli sviluppatori percepiscono che la produzione di software di alta qualità si traduca in prestazioni migliori del team e sono disposti a tollerare eventuali ritardi se questi garantiscono un prodotto finale eccellente poiché ciò riduce problemi futuri e libera risorse per nuove funzionalità (Drury-Grogan, 2014). Al contrario, un'architettura del codice carente può rallentare il progresso del progetto e aumentare il rischio di errori, compromettendo l'efficienza complessiva del team (Larusdottir and Kyas, 2020).

5.4. Individuo

	Motivazione	Capacità tecniche	Soft skill
Team performance in agile development teams: Findings from 18 focus groups			x
Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments	x	x	x
Analyzing Scrum Team Impediments Using NLP		x	
Towards a Better Understanding of Team-Driven Dynamics in Agile Software Projects: A Characterization and Visualization Support in JIRA	x		
The human factor in agility: Exploring employee dedication in agile project organizations	x		
A novel team productivity model for XP teams	x		
Understanding the perceived relevance of capability measures: A survey of Agile Software Development practitioners	x	x	x
Factors Influencing Productivity of Agile Software Development Teamwork	x		
What Do Agile Teams Find Important for their Success?	x	x	
Agile Team Perceptions of Productivity Factors		x	
TOTALE	**7**	**5**	**3**

Tabella 6: Frequenza dei fattori per la dimensione individuale

Come si può notare nella Tabella 6, relativa alla sfera individuale, la motivazione è il fattore citato con più frequenza ed è ritenuta fondamentale durante tutte le fasi del progetto, dalla

pianificazione iniziale alla consegna finale (Tavakoli and Gandomani, 2018). Un team motivato tende a essere maggiormente coinvolto nei processi di apprendimento, il che può, nel tempo, contribuire a migliorarne l'efficacia complessiva (Fatema and Sakib, 2017). La motivazione alimenta il desiderio di crescita personale e favorisce la partecipazione attiva a team ad alte prestazioni, con l'obiettivo di sviluppare ed esprimere le proprie competenze (Fagerholm *et al.*, 2015). Inoltre, rappresenta una forza trainante che orienta costantemente l'energia e gli interessi del gruppo verso il raggiungimento degli obiettivi (Vishnubhotla, Mendes and Lundberg, 2021). È stato dimostrato che il riconoscimento del raggiungimento degli obiettivi incrementa ulteriormente la motivazione; in particolare, le ricompense collettive, come attività sociali condivise, risultano più efficaci rispetto a premi di natura economica (Fagerholm *et al.*, 2015). In prospettiva, uno studio suggerisce l'utilizzo di indicatori oggettivi per misurare il livello di dedizione dei dipendenti, come ad esempio i dati anonimi relativi all'assenteismo o agli incentivi ricevuti (Meier and Kock, 2023).

Le competenze tecniche, come la padronanza di diversi linguaggi di programmazione e degli strumenti di sviluppo (Vishnubhotla, Mendes and Lundberg, 2021), nonché l'esperienza maturata da membri che hanno partecipato a più generazioni di progetti (Melo *et al.*, 2011), risultano fondamentali per il buon andamento di un team. Al contrario, un livello di competenza insufficiente può compromettere negativamente le prestazioni complessive (Kaleemunnisa *et al.*, 2023). È quindi essenziale che i membri del team possiedano conoscenze adeguate per affrontare le sfide tecniche e contribuire efficacemente agli obiettivi progettuali (Fagerholm *et al.*, 2015). In tal senso, la selezione di professionisti con un'elevata preparazione e un background diversificato rappresenta un elemento determinante per il successo di un progetto (Alahyari *et al.*, 2018).

Le soft skill, pur essendo meno menzionate rispetto ad altri fattori nella dimensione individuale, ricoprono un ruolo fondamentale nell'ottimizzare l'efficacia del lavoro. Se

adeguatamente sviluppate e combinate, possono risultare persino più determinanti delle competenze tecniche o delle qualifiche formali per il successo personale e professionale (Vishnubhotla, Mendes and Lundberg, 2021). Tra queste, l'adattabilità, intesa come la capacità di modificare le strategie in risposta a cambiamenti interni o esterni e all'analisi del contesto, è strettamente collegata alla performance del team, anche se nei progetti Agile tende spesso a essere data per scontata (Dingsøyr and Lindsjørn, 2013). Le abilità sociali, infine, sono essenziali per favorire la costruzione e il mantenimento di una solida identità di gruppo, poiché consentono agli individui di comunicare e collaborare efficacemente. Per questo motivo, dovrebbero essere considerate un criterio rilevante nella selezione dei membri di un team (Fagerholm *et al.*, 2015).

6. Conclusioni

L'obiettivo di questa ricerca è stato quello di analizzare in modo approfondito il tema della produttività nei team Agile, evidenziando come l'efficacia nella gestione dei progetti di sviluppo software non dipenda da un singolo elemento, bensì da una complessa interazione di fattori. A tal fine, la ricerca è riuscita ad individuare e categorizzare le principali variabili influenti, raggruppandole in quattro macro-dimensioni: team, organizzazione, processo e individuo. Per ciascuna di queste categorie è stata analizzata la frequenza con cui i fattori venivano citati e discussi negli articoli esaminati.

Dai risultati è emerso che i fattori più rilevanti per la performance del team sono quelli legati alle dinamiche relazionali e comunicative, in linea con i principi dell'Agile che pongono al centro le interazioni tra gli individui. Una gestione inefficace di questi aspetti può compromettere significativamente la produttività. Parallelamente, per la dimensione organizzativa, la gestione delle risorse e lo stile di leadership sono risultati i fattori più citati. Sul piano del processo, l'adozione di pratiche per lo sviluppo software si è rivelata l'elemento più influente sulla performance del team. Infine, nella sfera individuale, la motivazione è emersa come il fattore più rilevante.

I risultati ottenuti migliorano la comprensione delle dinamiche interne che determinano il successo del team, consentendo di individuare le aree critiche su cui intervenire e permettendo ai manager di adottare strategie più mirate. Inoltre, la conoscenza dettagliata di questi fattori favorisce la personalizzazione dei processi operativi, contribuendo a un miglioramento complessivo delle performance.

Sebbene sia stato possibile individuare con quale frequenza i fattori venivano citati negli articoli esaminati, non è stato possibile determinarne il peso effettivo in termini di impatto sulla produttività.

A partire dalla mappatura dei fattori individuata, studi futuri potrebbero approfondire il peso relativo di ciascun elemento attraverso l'analisi di casi aziendali concreti, utilizzando metriche specifiche per misurare la produttività e valutare l'efficacia delle strategie di ottimizzazione.

Inoltre, un ulteriore sviluppo potrebbe concentrarsi sulla costruzione di un modello dinamico in grado di prevedere l'impatto dei vari fattori sulla produttività del team. Un sistema di questo tipo fornirebbe un supporto decisionale efficace, permettendo di intervenire in modo proattivo sulle criticità e adottare misure mirate per migliorare la gestione dei progetti Agile.

7. Bibliografia

Alahyari, H. *et al.* (2018) 'What Do Agile Teams Find Important for Their Success?', in *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*. *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, Nara, Japan: IEEE, pp. 474–483. Available at: https://doi.org/10.1109/APSEC.2018.00062.

Barbosa, I.F. *et al.* (2017) 'Towards Understanding the Relationships between Interdependence and Trust in Software Development: A Qualitative Research', in *2017 IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. *2017 IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, Buenos Aires, Argentina: IEEE, pp. 66–69. Available at: https://doi.org/10.1109/CHASE.2017.12.

Bassil, Y. (2012) 'A Simulation Model for the Waterfall Software Development Life Cycle', *International Journal of Engineering*, 2(5).

Beck, K. *et al.* (2001) 'Manifesto for agile software development'.

Chen, C.-Y. and Lee, J.-C. (2023) 'Exploring Teams' Temporal Factors for Determining Process Tailoring that Promotes the Evolution of Agilebased Software Development', *ACM SIGMIS Database: the DATABASE for Advances in Information Systems*, 54(1), pp. 46–65. Available at: https://doi.org/10.1145/3583581.3583585.

Chiu, Y. (2010) *An Introduction to the History of Project Management: From the earliest times to AD 1900*. Eburon Uitgeverij BV.

De O. Melo, C. *et al.* (2013) 'Interpretative case studies on agile team productivity and management', *Information and Software Technology*, 55(2), pp. 412–427. Available at: https://doi.org/10.1016/j.infsof.2012.09.004.

Dingsøyr, T. and Lindsjørn, Y. (2013) 'Team Performance in Agile Development Teams: Findings from 18 Focus Groups', in H. Baumeister and B. Weber (eds) *Agile Processes in Software Engineering and Extreme Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Business Information Processing), pp. 46–60. Available at: https://doi.org/10.1007/978-3-642-38314-4_4.

Drury-Grogan, M.L. (2014) ‘Performance on agile teams: Relating iteration objectives and critical decisions to project management success factors’, *Information and Software Technology*, 56(5), pp. 506–515. Available at: https://doi.org/10.1016/j.infsof.2013.11.003.

Dutra, E. and Santos, G. (2020) ‘Organisational climate assessments of Agile teams – a qualitative multiple case study’, *IET Software*, 14(7), pp. 861–870. Available at: https://doi.org/10.1049/iet-sen.2020.0048.

Fagerholm, F. *et al.* (2015) ‘Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments’, *Information and Software Technology*, 64, pp. 132–147. Available at: https://doi.org/10.1016/j.infsof.2015.01.010.

Fatema, I. and Sakib, K. (2017) ‘Factors Influencing Productivity of Agile Software Development Teamwork: A Qualitative System Dynamics Approach’, in *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*. *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*, Nanjing: IEEE, pp. 737–742. Available at: https://doi.org/10.1109/APSEC.2017.95.

Fowler, M. and Highsmith, J. (2001) ‘The agile manifesto’, *Software development*, 9(8), pp. 28–35.

Haugan, G.T. (2011) ‘Project Management Fundamentals - Key Concepts and Methodology (2nd Edition)’, in. Management Concepts, Inc. Available at: https://app.knovel.com/hotlink/pdf/id:kt00UBFKL2/project-management-fundamentals/key-concepts-project.

Inayat, I. and Salim, S.S. (2015) ‘A framework to study requirements-driven collaboration among agile teams: Findings from two case studies’, *Computers in Human Behavior*, 51, pp. 1367–1379. Available at: https://doi.org/10.1016/j.chb.2014.10.040.

Iqbal, J., Omar, M. and Yasin, A. (2019) ‘An Empirical Analysis of the Effect of Agile Teams on Software Productivity’, in *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Sukkur, Pakistan: IEEE, pp. 1–8. Available at: https://doi.org/10.1109/ICOMET.2019.8673413.

Jiang, J., Yang, X. and Zhou, C. (2024) 'The impact of social media usage on expertise coordination and team creative performance in distributed agile software development', *Kybernetes*, 53(7), pp. 2414–2436. Available at: https://doi.org/10.1108/K-08-2022-1171.

Kakar, A.K. (2017) 'Do Reflexive Software Development Teams Perform Better?', *Business & Information Systems Engineering*, 59(5), pp. 347–359. Available at: https://doi.org/10.1007/s12599-017-0481-5.

Kaleemunnisa *et al.* (2023) 'Analyzing Scrum Team Impediments Using NLP', in A. Capozucca et al. (eds) *Frontiers in Software Engineering Education*. Cham: Springer Nature Switzerland (Lecture Notes in Computer Science), pp. 42–55. Available at: https://doi.org/10.1007/978-3-031-48639-5_4.

Kortum, F. *et al.* (2019) 'Towards a Better Understanding of Team-Driven Dynamics in Agile Software Projects: A Characterization and Visualization Support in JIRA', in X. Franch, T. Männistö, and S. Martínez-Fernández (eds) *Product-Focused Software Process Improvement*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 725–740. Available at: https://doi.org/10.1007/978-3-030-35333-9_56.

Kortum, F., Klunder, J. and Schneider, K. (2019) 'Behavior-Driven Dynamics in Agile Development: The Effect of Fast Feedback on Teams', in *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*. *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, Montreal, QC, Canada: IEEE, pp. 34–43. Available at: https://doi.org/10.1109/ICSSP.2019.00015.

Kumar, R., Gupta, A. and Singh, H. (2014) 'Agile Methodologies: Working Mechanism with Pros and Cons', in. Available at: https://api.semanticscholar.org/CorpusID:110097831.

Kwak, Y.-H. (2005) 'Brief history of project management', *The story of managing projects*, 9.

Larusdottir, M.K. and Kyas, M. (2020) 'Assessing the Performance of Agile Teams', in J. Abdelnour Nocera et al. (eds) *Beyond Interactions*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 71–83. Available at: https://doi.org/10.1007/978-3-030-46540-7_8.

Mashmool, A. *et al.* (2021) ‘A Statistical Model to Assess the Team’s Productivity in Agile Software Teams’, in *2021 IEEE 4th International Conference and Workshop Óbuda on Electrical and Power Engineering (CANDO-EPE)*. *2021 IEEE 4th International Conference and Workshop Óbuda on Electrical and Power Engineering (CANDO-EPE)*, Budapest, Hungary: IEEE, pp. 11–18. Available at: https://doi.org/10.1109/CANDO-EPE54223.2021.9667902.

Meier, A. and Kock, A. (2023) ‘The human factor in agility: Exploring employee dedication in agile project organizations’, *International Journal of Project Management*, 41(7), p. 102527. Available at: https://doi.org/10.1016/j.ijproman.2023.102527.

Melo, C. *et al.* (2011) ‘Agile Team Perceptions of Productivity Factors’, in *2011 AGILE Conference*. *2011 AGILE Conference*, Salt Lake City, UT, USA: IEEE, pp. 57–66. Available at: https://doi.org/10.1109/AGILE.2011.35.

Moloto, M., Harmse, A. and Zuva, T. (2021) ‘Agile Methodology use factors that influence project performance in South African Banking sector-A case study’, in *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, Mauritius, Mauritius: IEEE, pp. 1–6. Available at: https://doi.org/10.1109/ICECCME52200.2021.9590856.

Noruzi, A. (2005) ‘Google Scholar: The New Generation of Citation Indexes’, *LIBRI*, 55. Available at: https://doi.org/10.1515/LIBR.2005.170.

Parrish, A. *et al.* (2004) ‘A Field Study of Developer Pairs: Productivity Impacts and Implications’, *IEEE Software*, 21(05), pp. 76–79. Available at: https://doi.org/10.1109/MS.2004.1331306.

Pirola-Merlo, A. (2010) ‘Agile innovation: The role of team climate in rapid research and development’, *Journal of Occupational and Organizational Psychology*, 83(4), pp. 1075–1084. Available at: https://doi.org/10.1348/096317909X480653.

Project Management Institute (2017) *A guide to the project management body of knowledge (PMBOK guide)*. Sixth edition. Project Management Institute.

Rahy, S. and Bass, J.M. (2020) 'Implementation of Agile Methodology in Developing Countries: Case Study in Lebanon', in J.M. Bass and P.J. Wall (eds) *Information and Communication Technologies for Development*. Cham: Springer International Publishing (IFIP Advances in Information and Communication Technology), pp. 217–228. Available at: https://doi.org/10.1007/978-3-030-65828-1_18.

Ramírez-Mora, S.L., Oktaba, H. and Patlán Pérez, J. (2020) 'Group maturity, team efficiency, and team effectiveness in software development: A case study in a CMMI-DEV Level 5 organization', *Journal of Software: Evolution and Process*, 32(4), p. e2232. Available at: https://doi.org/10.1002/smr.2232.

Royce, W.W. (1987) 'Managing the development of large software systems: concepts and techniques', in *Proceedings of the 9th International Conference on Software Engineering*. Washington, DC, USA: IEEE Computer Society Press (ICSE '87), pp. 328–338.

Scott, E., Charkie, K.N. and Pfahl, D. (2020) 'Productivity, Turnover, and Team Stability of Agile Teams in Open-Source Software Projects', in *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Portoroz, Slovenia: IEEE, pp. 124–131. Available at: https://doi.org/10.1109/SEAA51224.2020.00029.

Ståhl, D. (2023) 'The dynamic versus the stable team: The unspoken question in large-scale agile development', *Journal of Software: Evolution and Process*, 35(12), p. e2589. Available at: https://doi.org/10.1002/smr.2589.

Strozzi, F. *et al.* (2017) 'Literature review on the "Smart Factory" concept using bibliometric tools', *International Journal of Production Research*, 55, pp. 1–20. Available at: https://doi.org/10.1080/00207543.2017.1326643.

Sutherland, J. *et al.* (2007) 'Distributed Scrum: Agile Project Management with Outsourced Development Teams', in *40th Annual Hawaii International Conference on System Sciences*, p. 274. Available at: https://doi.org/10.1109/HICSS.2007.180.

Tavakoli, F. and Gandomani, T.J. (2018) 'A Novel Team Productivity Model for XP Teams':, *Journal of Cases on Information Technology*, 20(4), pp. 93–109. Available at: https://doi.org/10.4018/JCIT.2018100106.

Thanthony, S., Marnewick, A. and Marnewick, C. (2022) 'Communication patterns and team performance within agile software development project', *International Journal of Agile Systems and Management*, 15(1), p. 118. Available at: https://doi.org/10.1504/IJASM.2022.124173.

Topp, J. *et al.* (2022) 'How a 4-Day Work Week and Remote Work Affect Agile Software Development Teams', in A. Przybyłek et al. (eds) *Lean and Agile Software Development*. Cham: Springer International Publishing (Lecture Notes in Business Information Processing), pp. 61–77. Available at: https://doi.org/10.1007/978-3-030-94238-0_4.

Vishnubhotla, S.D., Mendes, E. and Lundberg, L. (2020) 'Investigating the relationship between personalities and agile team climate of software professionals in a telecom company', *Information and Software Technology*, 126, p. 106335. Available at: https://doi.org/10.1016/j.infsof.2020.106335.

Vishnubhotla, S.D., Mendes, E. and Lundberg, L. (2021) 'Understanding the perceived relevance of capability measures: A survey of Agile Software Development practitioners', *Journal of Systems and Software*, 180, p. 111013. Available at: https://doi.org/10.1016/j.jss.2021.111013.

Wood, S. (2013) 'Successful extreme programming: Fidelity to the methodology or good teamworking?', *Information and Software Technology* [Preprint].

Zhao, D. and Strotmann, A. (2015) *Analysis and Visualization of Citation Networks*, *Synthesis Lectures on Information Concepts, Retrieval, and Services*. Available at: https://doi.org/10.2200/S00624ED1V01Y201501ICR039.